Advances in Data Analysis and Classification
https://doi.org/10.1007/s11634-022-00496-5

REGULAR ARTICLE

Over-optimistic evaluation and reporting of novel cluster algorithms: an illustrative study

Theresa Ullmann[1] · Anna Beer[2] · Maximilian Hünemörder[3] · Thomas Seidl[2] · Anne-Laure Boulesteix[1]

Received: 12 August 2021 / Revised: 18 December 2021 / Accepted: 14 February 2022

Abstract

When researchers publish new cluster algorithms, they usually demonstrate the strengths of their novel approaches by comparing the algorithms' performance with existing competitors. However, such studies are likely to be optimistically biased towards the new algorithms, as the authors have a vested interest in presenting their method as favorably as possible in order to increase their chances of getting published. Therefore, the superior performance of newly introduced cluster algorithms is over-optimistic and might not be confirmed in independent benchmark studies performed by neutral and unbiased authors. This problem is known among many researchers, but so far, the different mechanisms leading to over-optimism in cluster algorithm evaluation have never been systematically studied and discussed. Researchers are thus often not aware of the full extent of the problem. We present an illustrative study to illuminate the mechanisms by which authors—consciously or unconsciously—paint their cluster algorithm's performance in an over-optimistic light. Using the recently published cluster algorithm Rock as an example, we demonstrate how optimization of the used datasets or data characteristics, of the algorithm's parameters and of the choice of the competing cluster algorithms leads to Rock's performance appearing better than it actually is. Our study is thus a cautionary tale that illustrates how easy it can be for researchers to claim apparent "superiority" of a new cluster algorithm. This illuminates the vital importance of strategies for avoiding the problems of over-optimism (such as, e.g., neutral benchmark studies), which we also discuss in the article.

✉ Theresa Ullmann
tullmann@ibe.med.uni-muenchen.de

1 Institute for Medical Information Processing, Biometry, and Epidemiology, LMU Munich, Marchioninistr. 15, 81377 München, Germany

2 Institute for Informatics, LMU Munich, München, Germany

3 Department of Computer Science, CAU Kiel, Kiel, Germany

Published online: 17 March 2022

Keywords Over-optimism · Clustering · Cluster analysis · Evaluation

Mathematics Subject Classification 62H30 · 68W40

1 Introduction

Cluster analysis refers to grouping similar objects in data, while separating dissimilar ones. While there already are a huge number of cluster algorithms (see e.g., Xu and Wunsch (2010) for an overview), researchers continue to propose novel algorithms every year. Researchers who introduce a new cluster algorithm typically publish it together with a demonstration of the strengths of their approach and its superiority over alternative methods.

However, the results of such studies should be regarded with caution. *Publication bias* (Boulesteix et al. 2015) constitutes a considerable external incentive for researchers to demonstrate the superiority of their new approach: journals and conferences are much more likely to accept a paper about a novel computational method if this method shows good performance and is "better" than pre-existing approaches. This may tempt researchers to present their method's performance in an *over-optimistic* fashion, a mechanism that is also called the "self-assessment trap" (Norel et al. 2011). Such scenarios can not only appear in the research field of clustering but can also be found in all types of *methodological research*, i.e., the development and evaluation of data analytic techniques and algorithms (Boulesteix et al. 2020).

Over-optimization is not necessarily performed in a malicious or even intentional manner, but it is problematic because the new method may turn out to have a worse performance than initially claimed when it is later investigated in a neutral comparison study, i.e., a study whose authors do not have a vested interest in one of the competing methods, see Boulesteix et al. (2013). In other words, the good performance result is not replicable (Boulesteix et al. 2020). Anecdotal evidence for this lack of replicability is presented by Buchka et al. (2021) for a specific data analysis problem related to the pre-processing of a special type of high-throughput molecular data. The over-optimistic presentation of computational methods may lead to the usage of flawed methods in applications, which could ultimately hinder research progress or even lead to questionable results in applied research.

But how exactly may researchers present their new methods in an over-optimistic fashion? For *supervised* classification, an illustrative case has already been presented in the field of bioinformatics by Jelizarow et al. (2010). They considered a "promising" novel classification method, which in reality was not superior to other classifiers. Yet the authors were able to demonstrate that different mechanisms allow over-optimistic presentation of this new method's performance, namely choosing specific datasets, optimizing the method's settings and characteristics to these datasets while burying the other in the file drawer, and choosing suboptimal competing classifiers.

However, to the best of our knowledge, such a study has not yet been conducted for cluster analysis, i.e., the unsupervised scenario. While over-optimistic (selective) reporting is well understood in the context of statistical testing and supervised learning, where its impact can be easily measured, it is much less so in the field of cluster analysis, which is characterized by the difficulty to properly evaluate methods. We thus aim at filling this gap by demonstrating how a novel cluster algorithm's performance can be presented in an (overly) favorable light.

The problem of over-optimism is in fact as important in unsupervised clustering as it is in supervised classification, and is probably even exacerbated because the performance evaluation of cluster algorithms has not been studied as systematically as the evaluation of supervised classifiers in the methodological literature. Guidance for proper benchmarking of cluster algorithms has only recently emerged (Van Mechelen et al. 2018). Even though the "true" cluster labels are unknown in clustering applications, researchers typically use datasets with known labels to evaluate their novel cluster algorithms. To some extent, the performance evaluation of cluster algorithms thus appears similar to the evaluation of classifiers. Yet for cluster analysis, the role of test data is not as clear-cut as in supervised classification (Ullmann et al. 2021), which entails that researchers are less aware that "overfitting" can not only happen in supervised classification, but also in cluster analysis. Moreover, optimizing hyperparameters such as the number of clusters based on the "ground truth", as is frequently done in cluster algorithm evaluation, does not take into account that other researchers who eventually want to use the algorithm in applications do not know the "true" cluster labels of their datasets, and will thus likely obtain worse results than the performances reported in the original evaluation of the novel algorithm. To evaluate their new method, researchers might also use performance evaluation measures which do not require a fixed "ground truth", such as internal validation indices which measure internal properties of the data (e.g., homogeneity and/or separateness of the clusters). However, over-optimism can still be an issue when using these indices.

In the present study, we use the "Rock" algorithm (Beer et al. 2019) as an illustrative example. Beer et al. (2019) agreed to the usage of their algorithm in our paper. Rock was originally introduced as a "promising" new algorithm and was presented as being able to outperform competitors. In subsequent studies, it turned out that Rock does not generally perform better than its competitors. In the present paper, we show that Rock outperforms competing algorithms in very specific scenarios and that these scenarios can be obtained by three different mechanisms: (1.) optimization of datasets and data characteristics, (2.) optimization of parameters of the Rock algorithm and (3.) the choice of the competing clustering approaches. We demonstrate that if the optimized scenarios are selectively reported and the settings in which Rock performs worse are omitted, the algorithm then appears to outperform its competitors—as a result of an over-optimistic presentation.

Rock is used only as an example—demonstrating the specific characteristics of the Rock algorithm is not the main interest of our work. Rather, we use Rock to illustrate more general mechanisms of over-optimization. We suspect that many studies which introduce new cluster algorithms are affected by these mechanisms. However, given that over-optimization can happen quite subtly and/or unintentionally, we do not cite any published papers here which probably presented their results in an over-optimistic fashion. Neither do we try to quantify the actual optimistic bias that currently exists in the literature on cluster algorithms. Rather, our study is intended as a cautionary tale to raise awareness of the over-optimism problem, and to illuminate the importance of using strategies to avoid over-optimism (e.g., avoiding selective reporting, using independent test data and conducting neutral benchmark studies, as discussed in detail in Sect. 6).

We first give an overview of related work in Sect. 2. Section 3 explains how we performed optimization of Rock's performance. The corresponding results are presented in Sect. 4 and further discussed in Sect. 5. Possible solutions for the problem of over-optimism are outlined in Sect. 6. We conclude the paper in Sect. 7.

2 Related work

In this section we discuss studies that are related to our work. After presenting studies which directly look at the over-optimistic bias of new computational methods, we address aspects in the field of data mining that are connected to over-optimistic presentation of cluster algorithms.

2.1 Previous work about over-optimistic bias of new computational methods

There appears to be a lack of literature about over-optimism in the introduction of new cluster algorithms. For computational methods other than clustering, there exist some studies, to our knowledge mostly in the field of bioinformatics.

As mentioned above, a study similar to ours was previously reported by Jelizarow et al. (2010), but for supervised classification. Moreover, while this study illustrated over-optimism with a classification method for gene expression data and used real cancer gene expression datasets for this purpose, our example is not application specific. For performance evaluation we choose simulated and real datasets which are frequently used for the evaluation of cluster algorithms in computational research (e.g., the synthetic "Two Moons" dataset, the Iris dataset etc., see Sect. 3).

Broadly speaking, the three categories of optimization mechanisms that we analyze are similar to the categories previously considered in Jelizarow et al. (2010), i.e., optimization of the data, optimization of the algorithm's characteristics, and the choice of competing approaches. However, the use of simulated data allows us to systematically consider data characteristics such as noise or dimensionality, which was not done for the real datasets used in Jelizarow et al. (2010).

In a similar application context, Yousefi et al. (2010) also addressed over-optimism when reporting the performance of newly proposed classifiers. They focused on classification on high-dimensional data with low sample size, such as gene expression data. The authors specifically considered the optimization of the datasets, i.e., they analyzed the optimistic bias that results from reporting only the datasets with the best (or second best) performance of the new classifier. They estimated this bias in a simulation study, by repeatedly sampling sets of datasets, and recording the best (or second best) performing dataset of each set. The aim of their study thus was to *quantify* the optimistic bias with specific focus on the choice of datasets, whereas we model different over-optimization mechanisms of a (hypothetical) researcher in an illustrative way. The results of Yousefi et al. (2010) show that in the high-dimensional data setting, there is indeed a large optimistic bias when reporting only the best or second best performing dataset.

Finally, again in the context of bioinformatics, a recent study aimed to estimate the optimistic bias in the reported performance of new computational methods to preprocess a special type of raw high-throughput molecular data (Buchka et al. 2021). The approach was to perform a literature search and compare the reported performance of newly introduced methods against their performance in later neutral comparison studies. As expected, novel methods were ranked better than competitors in most of the papers introducing them, but outperformed competitors at a lesser rate in neutral studies. Yet the new methods still outperformed more than 50% of their paired competitors in neutral studies, showing that while there is optimistic bias, there is also some level of genuine scientific progress.

Outside of bioinformatics, Ferrari Dacrema et al. (2021) assessed optimistic bias in research about recommender systems. Recommender algorithms can be used, for example, to propose new movies to a media streaming user based on previously watched movies. Many new recommendation algorithms based on deep learning were published in recent years, which usually claimed superiority over previous approaches. Ferrari Dacrema et al. (2021) repeated the evaluations of the original authors, but with additional baseline algorithms. Their analysis showed that most of the new methods did not actually outperform simple and long-known baseline algorithms, provided strong-performing baselines were chosen and their hyperparameters were tuned as carefully as those of the new algorithms. This highlights that not including strong competitors or not treating the competing methods fairly might lead to optimistic bias.

2.2 Information visualization

Over-optimistic presentation of results can also be obtained by visualization methods, i.e., not only by a biased selection of *which* data to show, but also by *how* the selected data is shown. Studies on *information visualization* address the latter aspect. For example, visualization methods with a high lie factor (the ratio between "size of effect shown in graphic" and "size of effect in data", see Tufte (1983)), or misleading labeling and scaling of axes, could be used by a researcher to let their algorithm appear in a more favorable light.

We do not focus on such mechanisms in our study, and instead illustrate that over-optimistic reporting of results is also possible if all rules regarding "correct" information visualization are observed.

2.3 Robustness

Robust clustering algorithms yield a similar quality of results for similar input. Thus, it is unlikely that there are experimental setups which yield notably better results than similar experiments and could thus be selectively presented in an over-optimistic fashion. We do not systematically evaluate the robustness of any of the tested cluster algorithms in Sect. 4, but rather show how the lack of robustness can be exploited in order to over-optimistically present the results of the exemplary algorithm. Out of the diverse types of robustness, we focus on the lack of robustness regarding different properties of the data as well as hyperparameter settings. For example, we consider robustness w.r.t. noise. "Noise" can mean either background noise, i.e., uniformly distributed points across the data space which do not belong to the original distribution, or jitter, i.e., small deviations or perturbations in the original distribution. We regard only the latter in our experiments.

That robustness is crucial for clustering algorithms was already stated by Davé and Krishnapuram (1997). In recent literature on cluster algorithms, the robustness regarding different properties of the data is often presented, e.g., the size of the dataset, number of clusters, dimensionality, and structure of the data. Usually there is a base case for which one property at a time is changed to regard the effects on the clustering result. However, it is often left unclear how and why this base case was obtained, and how the settings which are not regarded in the respective experiment are chosen.

Even though the robustness regarding the choice of hyperparameters seems similarly important, authors often refer to "expert knowledge" for finding the "best" setting, and omit a robustness analysis. This can lead to enormous disagreements in the evaluation of an algorithm, see, e.g., the controversy about DBSCAN (Ester et al. 1996; Gan and Tao 2015; Schubert et al. 2017). Even easily interpretable hyperparameters, such as the number of clusters k (e.g., for k-Means, Lloyd 1982), which at first sight do not seem to require a robustness analysis, might show better performance w.r.t. the evaluation measure when set at a value different from the "ground truth".

To summarize, robustness regarding different aspects is not only important to guarantee a predictable quality of clustering for users, but also reduces the potential for over-optimism.

2.4 Adversarial attacks

An adversarial attacker may corrupt the results of an algorithm by only performing small changes or additions in a dataset, leading to a wrong but more favorable outcome for the attacker (Goodfellow et al. 2018). Even though adversarial attacks are most often regarded in context of supervised machine learning, they can also influence results of unsupervised machine learning: recently, Chhabra et al. (2020) showed that adversarial attacks are also possible for clustering, even without knowing important details of the cluster algorithm. Algorithms which tend to return results of highly varying quality, also for only small perturbations in the data, are easy victims not only for adversarial attacks, but also for over-optimism. However, where adversarial attackers aim at changing only certain results, over-optimistic researchers would try to change the impression of an algorithm's overall quality. By knowing the details of their novel algorithm as well as deciding on all hyperparameters and competitive methods, the influence over-optimistic researchers can have on the presentation of their results is massive, especially compared to an adversarial attacker.

3 Over-optimization methods

In this section we outline the concept and the experimental design of our study. We first explain the three different categories of over-optimization mechanisms that we illustrate in our study. We then detail our concrete implementation, e.g., the clustering algorithms, datasets, evaluation measure and optimization method.

3.1 Three categories of over-optimization

Imagine a researcher who wishes to present his/her cluster algorithm in a favorable light. We model the work process of this researcher as an "optimization task": the characteristics of the study in which the new algorithm is compared to existing ones are optimized such that the researcher's algorithm scores well, in particular better than the best performing competing algorithm. This optimization can refer to (1.) finding datasets or data characteristics for which the new algorithm works particularly well, (2.) finding optimal parameters of the algorithm (and vice versa, neglecting the search for optimal parameters for the competitors) or (3.) choosing specific competing algorithms.

Optimizing datasets or data characteristics. A new cluster algorithm might perform well for specific types of datasets, but not for other types. Researchers might decide to report only the best-performing types of datasets. Additionally, for synthetic datasets, there is potential for over-optimism when varying specific characteristics (e.g., the amount of noise, the sample size, or the number of dimensions), and reporting only the optimal settings. Moreover, simulated datasets depend on the random seed, such that in turn, the performance of the cluster algorithm might also vary over different random seeds. Researchers might actively look for a "good" random seed or simply stumble across a particular "good" random seed by chance, neglecting to try other random seeds to check for robustness.

Optimizing the algorithm's parameters or characteristics. Hyperparameters of the cluster algorithm, or characteristics of the algorithm designed during the development phase, could be varied by researchers to look for the best result. Hyperparameter optimization (HPO) is per se a legitimate procedure in performance evaluation. However, there is less awareness for proper evaluation of cluster algorithms combined with HPO, compared to the more extensive methodological literature on correct evaluation of supervised classifiers with HPO (Boulesteix et al. 2008; Bischl et al. 2021). In cluster analysis, over-optimism in relation to HPO may result from (1.) optimizing hyperparameters based on the "true" cluster labels known to the researchers, and (2.) not splitting the data into training and test sets. Both aspects will be discussed in more detail in Sects. 4 and 5. Moreover, over-optimism might also result when researchers neglect to set optimal parameters for the *competing* algorithms, e.g., when choosing suboptimal hyperparameter defaults for the competitors while finetuning their own algorithm.

Optimizing the choice of competing algorithms. Finally, researchers might pick specific competing clustering methods that let their own algorithm appear in a better light. They could neglect to look for the best state-of-the-art competitor, instead opting for less optimal comparison algorithms. Even if the researchers are aware of state-of-the-art competitors, they might not include them because the codes are not openly available, or implemented in a programming language which they are not familiar with. Researchers could also think of different groups of competing cluster algorithms, and then pick the group that is most favorable for comparison with their own algorithm. A new density-based cluster algorithm could for example be compared either with a group of other density-based algorithms, or with a group of some well-known, not necessarily density-based cluster algorithms. While both choices could in principle be sensible, it is over-optimistic if researchers either deliberately exclude a class of competitors a priori because they expect their novel algorithm to perform worse than this class, or if they choose the competitor group a posteriori after having seen the results (Jelizarow et al. 2010).

Apart from these three categories of optimization, there are some further optimization possibilities (e.g., optimizing the evaluation measure) that we do not analyze here in detail, but briefly discuss in Sect. 5.

We assume that usually, researchers do not consciously perform the three classes of optimization tasks in a malicious and systematic manner. Nevertheless, in the course of a longer research process during which researchers try different datasets, algorithm parameters/configurations and competing algorithms, researchers might optimize the settings in an unsystematic and (probably) unintentional manner. Even if researchers start their analysis with the best intentions, they might post-hoc rationalize their (over-optimistic) choices as perfectly reasonable decisions, given that "[h]umans are remarkably good at self-deception" and scientists often "fool themselves" (Nuzzo 2015).

One might argue that the optimizations outlined above are not actually *over*-optimizations and that it is perfectly fine to look for scenarios in which a novel algorithm performs well. We would agree that it is not a priori wrong to search for and report such scenarios, as a new cluster algorithm can never be expected to outperform every other cluster algorithm in every situation. However, it should also be transparently reported how the presented "successful" scenarios were obtained, and how the algorithm performs in other settings. Over-optimism ultimately appears when performance results are *selectively* reported. We will illustrate this with our results in Section 4.

3.2 Experimental setup

We now present the exemplary cluster algorithm and its settings, the competing algorithms, the datasets and the evaluation measure. Our fully reproducible code is available at https://github.com/thullmann/overoptimism-clust-algo.

In accordance with the authors, we used the already published algorithm Rock (Beer et al. 2019) as a novel and promising algorithm. Rock is an iterative approach similar to Mean Shift (Fukunaga and Hostetler 1975), but based on the k nearest neighbors (kNN) instead of the bandwidth. In each step, points "roam" to the mean of their respective k nearest neighbors. Points with a similar final position are assigned to a common cluster. The algorithm involves the hyperparameter t_{max}, which gives the maximum number of iterations. As the maximum meaningful value for k is fixed ($k > \frac{n}{2}$ would lead to an assignment of all points to the same cluster), and the increase of k in every step is linear, t_{max} also determines the number k of nearest neighbors regarded in each iteration. The larger t_{max} is chosen, the closer values for k are in consecutive steps. Lower values for t_{max} thus lead to larger gaps between consecutive values for k, which may cause volatile merges of different clusters. On the other hand, higher values for t_{max} lead to more iterations, which increases runtime.

As typical for short papers, only a limited number of experiments is presented in Beer et al. (2019), illustrating that the underlying idea is promising. The results for Rock looked good compared to k-Means (Lloyd 1982), DBSCAN (Ester et al. 1996) and Mean Shift, which are typical competitors in the field and representatives for algorithms finding different types of clusters. As examples for competing algorithms, we thus chose k-means, DBSCAN, Mean Shift and additionally Spectral Clustering (Ng et al. 2001).

As the clustering performance measure we use the Adjusted Mutual Information Score (AMI, Vinh et al. 2010), a version of the Mutual Information (MI) Score adjusted for chance agreement of random partitions. For each dataset and cluster algorithm, the known "true" clustering (as given either by the simulation design for the synthetic datasets or by additional label information for the real datasets) was compared via the AMI with the clustering found by the algorithm. The higher the AMI, the more similar the two clusterings are. The AMI attains its maximum value of 1 if the two clusterings are identical, and equals 0 if the MI between the two clusterings is equal to the MI value expected for two random partitions. We give the detailed mathematical definition of the AMI in the appendix A.

While we only use the AMI in our illustration for the sake of conciseness, a similar analysis could be performed for alternative indices which measure the agreement of the calculated clusterings with the "ground truth", or even for internal validation indices which evaluate clusterings based on internal properties of the data alone and do not require the "ground truth" (see also the discussion in Sect. 5.2).

The choice of exemplary datasets is linked to the three different optimization tasks outlined in Sect. 3.1. We thus give the datasets for each task in turn and explain how the optimization was performed. Note that we performed the three optimization tasks sequentially, building on the results of each previous task. Of course, in reality, a researcher will likely not perform the optimizations in such a perfectly sequential matter, and might jump between different tasks of optimization or try to optimize different aspects simultaneously. Again, our sequential procedure merely serves illustrative purposes.

For some specific details of the implementation, we refer to the appendix A.

Optimizing datasets and data characteristics. For this part of the analysis, we chose three commonly used different synthetic datasets from scikit-learn (Pedregosa et al. 2011), see Fig. 2: Two Moons[1], Blobs[2] (for details on this dataset, see the appendix A), and Rings[3].

[1] https://scikit-learn.org/stable/modules/generated/sklearn.datasets.make_moons.html, visited: 05/31/2021.

[2] https://scikit-learn.org/stable/modules/generated/sklearn.datasets.make_blobs.html, visited: 05/31/2021.

[3] https://scikit-learn.org/stable/modules/generated/sklearn.datasets.make_circles.html, visited: 05/31/2021.

First, we performed optimization by varying the following data characteristics: a) for Two Moons, the sample size and the jitter values (where "jitter" denotes small random perturbations to the original data points in the clusters), b) for Blobs, the sample size, the number of dimensions and the number of generated clusters ("blobs"), and c) for Rings, the sample size and the jitter values. The goal of the optimization was to find the parameter configuration (e.g., for Two Moons, the configuration (n, j) of sample size and jitter value) that yields the largest performance difference between Rock and the best of the competitors – which is not necessarily the parameter configuration that yields the best *absolute* performance of Rock.

That is, for each of the three types of synthetic datasets in turn, we performed the following formal optimization task:

$$\text{argmax}_{D \in \mathcal{D}} \left\{ \frac{1}{10} \sum_{i=1}^{10} \left(AMI\left(Rock(D^i), y_{D^i}\right) - \max_{C \in \mathcal{C}} AMI\left(C(D^i), y_{D^i}\right)\right)\right\} \tag{1}$$

where $D \in \mathcal{D}$ denotes the different variants of the dataset. For example, for the Two Moons data, each dataset D is a version of Two Moons with a specific jitter value and sample size. Each D has a cluster label ground truth y_D. For each $D \in \mathcal{D}$, ten different versions of D, namely $D^i, i = 1, \ldots, 10$ resulting from ten different random seeds were generated. Put differently, we performed ten simulation iterations per setting, i.e., we sampled ten datasets from each data distribution with a specific data parameter setting. The AMI difference is then averaged over these ten versions. This is supposed to reduce the influence of the random seed. Only at a later point in the analysis did we look at the effect of picking specific random seeds (see below). $Rock(D^i)$ denotes the application of Rock to the data D^i, returning a partition of the objects. Analogously, the competing algorithms $C \in \mathcal{C}$ return a partition of D^i, with $\mathcal{C} = \{k\text{-means, DBSCAN, Mean Shift, Spectral Clustering}\}$.

For each of the three types of datasets in turn, we performed the optimization task (1) by using the Tree-structured Parzen Estimator (TPE, Bergstra et al. 2011), as implemented in the Optuna framework (Akiba et al. 2019) in Python[4]. TPE is a Bayesian optimization (BO) method. BO approaches sequentially propose new parameter configurations based on a library of previous evaluations of the objective function (for more details on BO methods and the TPE, see the appendix A). The TPE is often used for hyperparameter optimization of machine learning models, but in our case, we use it to optimize the *data* parameters. The TPE optimization can be considered as a very simplified model of the researcher's optimization procedure. Of course, a researcher's behavior does not exactly correspond to the mathematical procedure of the TPE. However, if researchers perform *intentional* (over-)optimization, then they might indeed use an optimization method such as the TPE to find the best data settings. The Bayesian optimization mimics the researcher's (unintentional) over-optimization in the following sense: as mentioned above, a researcher developing a new cluster algorithm might sequentially look for data settings in which the new algorithm performs well, taking

[4] https://optuna.readthedocs.io/en/stable/reference/generated/optuna.samplers.TPESampler.html, visited: 05/31/2021.

into account performance information from previously tried data parameters. This is the reason why we chose the TPE over a simple grid search or random search, because the latter do not use previously obtained performance information. To make the TPE process more "realistic", we supplied a grid of limited discrete values to the TPE, given that a researcher presumably would not try arbitrary real numbers. We performed this experiment with only 100 optimization steps for each of the three types of datasets, in order to fairly represent a researcher trying different data parameters by hand.

After determining the optimal values for the data parameters (which we will later report in Table 1 in Sect. 4.1), we analyzed the performance of Rock for non-optimal parameter values. That is, for each dataset and single data parameter in turn, the parameter was varied over a list of values, while the other data parameters were kept fixed at their optimal values. For example, for the Two Moons dataset we tried different jitter values and plotted the corresponding performance as measured by the mean AMI over ten random seeds against the jitter, keeping the sample size at the optimal value determined by the TPE. These analyses show the effects of selectively reporting only the best data parameters versus the performance of the algorithm over a broader range of each data parameter.

In the experiments given so far, we always considered the AMI averaged over ten random seeds. In the final step of the analysis for this section, we specifically study the influence of individual random seeds. We take the Two Moons dataset as an example, with a data parameter setting which is not optimal for Rock, but for which DBSCAN performs very well. We generate 100 datasets with these characteristics by setting 100 different random seeds, to check whether there exist particular seeds for which Rock does perform well, leading to over-optimization potential.

For all experiments described so far, we applied reasonable parameter choices (defaults or heuristics) for the cluster algorithms. For Rock we chose $t_{max} = 15$, as done for all experiments in the original paper (Beer et al. 2019), and for the competing algorithms see the appendix A.

Optimizing the algorithm's parameters or characteristics. For this example we varied Rock's hyperparameter t_{max} (maximum number of iterations). As t_{max} is discrete with a reasonable range of $\{1, \dots, 30\}$, a researcher could easily try every value by hand. Thus we did not perform optimization with the TPE, but with a full grid search, i.e., we calculated the AMI performance of Rock for each value of t_{max} and for each dataset. For this illustration, we considered the *absolute* performance of Rock, given researchers would also strive to maximize the absolute performance of their novel algorithm.

As exemplary datasets, we again considered Two Moons, Blobs and Rings, and additionally four real datasets frequently used for performance evaluation: Digits, Wine, Iris and Breast Cancer as provided by scikit-learn[5] (see also the UCI Machine Learning Repository, Dua and Graff 2017). The data parameter settings for the three synthetic datasets (sample size, amount of jitter etc.) corresponded to the optimal settings from the TPE optimization of (1). We used a single random seed to generate the illustrative synthetic datasets.

5 https://scikit-learn.org/stable/datasets/toy_dataset.html, visited: 05/31/2021.

In a next step, using the Two Moons dataset as an example, we compared the AMI performances of Rock and DBSCAN over ten random seeds, first without, then with hyperparameter optimization for Rock and DBSCAN. We used the TPE for HPO of DBSCAN. Here, the TPE was not intended to model a researcher's behavior, but was used as a classical HPO method. The comparison illustrates the effect of neglecting parameter optimization for competing algorithms.

Optimizing the choice of competing algorithms. We did not perform new experiments here. Rather, we looked at the results from the two previous optimization tasks to derive the potential for optimization of the choice of competing cluster algorithms.

4 Results

We present our results for the three optimization tasks outlined above, starting with the optimization of datasets and data characteristics.

4.1 Optimizing datasets and data characteristics

In this subsection we examine how strongly the choice of the "best" properties of a dataset, along with the type of dataset, can influence the performance estimation of Rock.

4.1.1 Optimization of the data parameters with TPE

Table 1 reports the optimal data parameters for the three synthetic datasets as determined by the TPE optimization. The search space for each parameter is given in parentheses and consists of discrete values. The column "AMI diff." shows the difference of the AMI obtained by Rock to the AMI obtained by the best competitor (averaged over ten random seeds). Recall that the AMI difference was used as the optimization criterion by the TPE to find the "optimal" parameter configuration. The column "Abs. AMI" denotes the absolute performance of Rock as measured by the AMI averaged over ten random seeds. The standard deviation over the seeds is also displayed.

Table 1 Optimal data parameters as determined by the TPE optimization

Dataset	Sample size	Jitter	# of dim.	# of clusters	AMI diff.	Abs. AMI
Two Moons	**1000**	**0.15**	2	2	+0.3581	0.7881
	([1, 16] · 100)	([1, 20] · 0.01)	(default)	(default)		±0.1583
Blobs	**300**	–	**3**	**2**	+0.0475	0.8881
	([1, 16] · 100)		([2,20])	([2,10])		±0.1573
Rings	**1600**	**0.02**	2	2	+0.1789	0.1789
	([1, 16] · 100)	([1, 20] · 0.01)	(default)	(default)		±0.0026

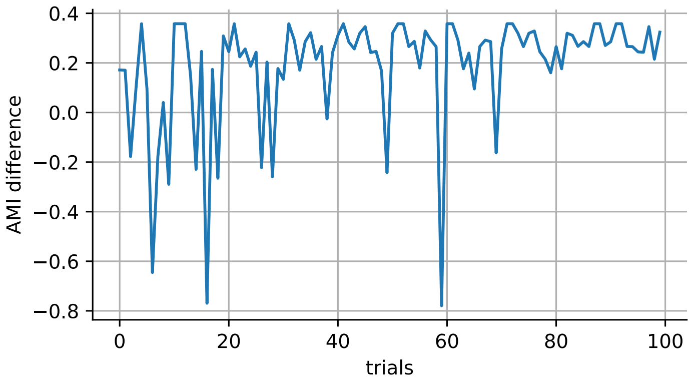

Fig. 1 Optimization progression for the Two Moons dataset, with the AMI difference averaged over ten random seeds

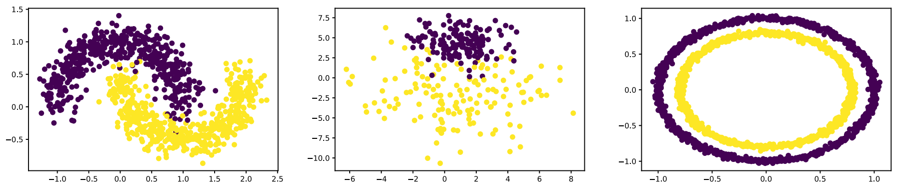

Fig. 2 Example datasets (Two Moons, Blobs, Rings) with the optimal data parameters. For the Blobs example we only show the first and second dimensions

For the example of the Two Moons dataset, Fig. 1 shows a graphical representation of the TPE process over 100 optimization steps. The final “optimal” result is given by the best trial out of the 100 trials. The datasets with the optimal settings are pictured in Fig. 2, using a single illustrative seed of 0.

Judging from the results in Table 1, Rock appears to show better performance than its competitors. A researcher could use the results to claim Rock’s “superiority”. However, the *absolute* performance of Rock for the Rings dataset is not very good with a mean AMI of only 0.1789. Rock is only the best algorithm here because the competing methods completely fail to detect the clustering. A researcher who tries to optimize the data types might thus decide to let the Rings dataset disappear in the “file drawer”, particularly if he/she must omit some results due to page limits, and only present the Two Moons and Blobs datasets, for which Rock performs well, both in absolute and in relative (compared to competitors) terms. But would this presentation for Two Moons and Blobs be over-optimistic? To obtain a more realistic picture of Rock’s abilities, we analyze the results when the data parameters are not set at the optimal values, but varied over a grid.

4.1.2 Varying the data parameters

We consider the influence of the sample size, the number of dimensions and the amount of jitter. For each data parameter, we pick one data type for illustrative purposes (either Two Moons or Blobs). The data parameters that are not currently considered are set

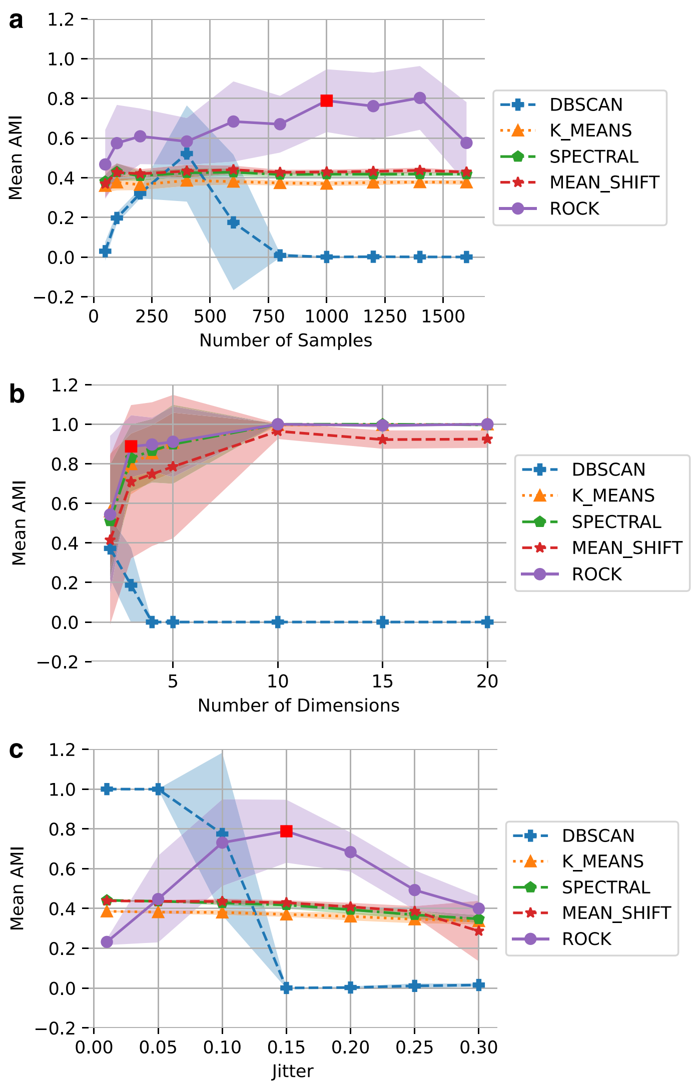

Fig. 3 **a** Varying the sample size for the Two Moons dataset (jitter = 0.15), **b** varying the number of dimensions for the Blobs dataset (sample size = 300, number of blobs = 2), **c** varying the jitter amount for the Two Moons dataset (sample size = 1000)

to their optimal values from Table 1. Figure 3a–c show the performance of Rock and its competitors measured by the AMI over ten random seeds, depending on the varied data parameters. The border around each line shows the standard deviation over the seeds. Red squares indicate the optimal setting from Table 1.

Sample size. Here we consider the Two Moons dataset in Fig. 3a. We tried the following sample sizes: 50, 100, 200, 400, 600, 800, 1000, 1200, 1400, 1600. The jitter value is set at its optimal value 0.15 from Table 1. Rock indeed appears to perform better here than its competitors over a broader range of numbers of samples, not just for the optimal setting. However, at smaller sample sizes, the difference to k-means, spectral clustering and Mean Shift is less impressive than at Rock's optimal setting of $n = 1000$.

Dimensionality. The Blobs dataset is analyzed in Fig. 3b, varying the number of dimensions over {2, 3, 4, 5, 10, 15, 20}. The sample size is set at 300 and the number of generated blobs is 2, according to Table 1. Rock performs better than competitors mainly for small dimensions. Once the number of dimensions exceeds 5, Rock cannot outperform k-means and Spectral Clustering.

Jitter. The amount of jitter is varied for the Two Moons dataset, see Fig. 3c. We tried the following jitter amounts: 0.01, 0.05, 0.10, 0.15, 0.20, 0.25, 0.30. The sample size is set to the optimal value of 1000 according to Table 1. Rock performs better than its competitors for the jitter set at 0.15 and above. However, for lower jitter values, Rock cannot outperform DBSCAN. Moreover, for jitter values of 0.25 and 0.30, the difference from Rock to k-means, spectral clustering and Mean Shift is quite low and not as impressive as at the optimal setting of 0.15.

To summarize, the performance of Rock is not robust with respect to variation of the data parameters, which leads to potential for over-optimization. While Rock is indeed better than its competitors for certain ranges of the data parameters, there are also settings for which Rock either does not perform better than the competitors, or the performance advantage is small. Thus the apparent "superiority" of Rock is generally less impressive than indicated by the results found from the TPE optimization in Table 1.

4.1.3 Influence of the random seed

For the analyses mentioned so far, the mean AMI over ten random seeds was considered. However, it is also possible that a researcher chooses a particular random seed for which Rock performs well. As seen in Fig. 3c, Rock is outperformed by DBSCAN on the Two Moons dataset for a jitter value of 0.05 and 1000 samples. This statement is based on the AMI averaged over 10 random seeds. But could there also be particular

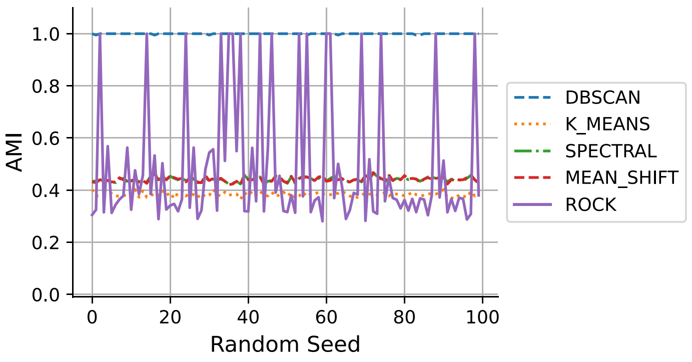

Fig. 4 Performance of the cluster algorithms on the Two Moons dataset (sample size = 1000, jitter = 0.05) over 100 random seeds

random seeds for which Rock does perform well? In Fig. 4, we display the behavior of Rock and its competitors over 100 different random seeds. Since Rock performs as well as DBSCAN for some particular seeds, there is potential for an over-optimizing researcher to pick such a seed. While deliberately trying multiple seeds and presenting only the best one can be considered as malicious behavior, it is also possible that the seed set by the researcher is by chance a "good one", and that the researcher does not consider a dependence of the performance on the random seed. To avoid such unintentional over-optimism, it is advisable to account for sampling variability and average over multiple random seeds, even when the cluster algorithm itself is deterministic. While the practice of sampling multiple datasets from a data distribution is well-known in statistics, this is sometimes neglected when evaluating data mining tasks like clustering.

4.2 Optimizing the algorithm's parameters

We analyze how the hyperparameter t_{max} of Rock can be optimized. In contrast to the previous sections, we now consider the absolute performance of Rock, given that a researcher would presumably not only try to outperform competitors, but also strive to obtain AMI values for Rock which are close to 1.

Additionally to Two Moons, Blobs and Rings, we consider the four real datasets mentioned in Sect. 3.2: Digits, Wine, Iris, Breast Cancer. For the Two Moons, Rings and Blobs datasets, we used the optimal data parameters from Table 1 and only generated a single illustrative dataset for each type by using 42 as a random seed. In accordance with typical evaluation of cluster algorithms, we do not split the datasets into training and test sets (see, however, the discussion in Sect. 6.2).

Figure 5 shows the performance of Rock as measured by the AMI, over t_{max} ranging from 1 to 30.

It can be seen that for different datasets, different t_{max} values are optimal. An optimistic researcher could report (only) the best t_{max} and the corresponding performance for each dataset. Optimizing hyperparameters of a cluster algorithm based on the "ground truth" of datasets (here via the AMI) is frequently seen in the literature.

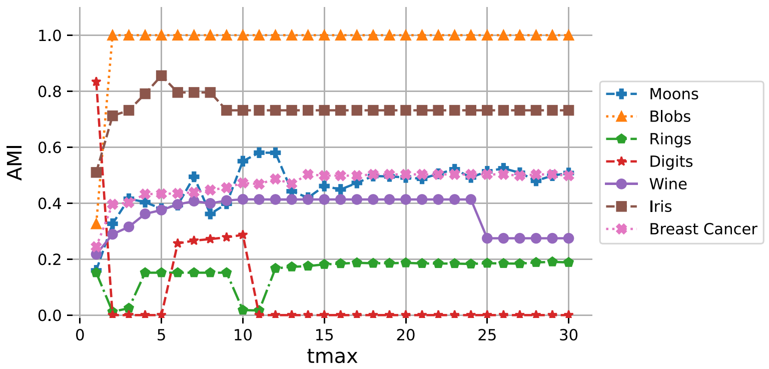

Fig. 5 Varying the hyperparameter t_{max} of Rock for different datasets

But as mentioned above, this could be over-optimistic with regards to the future performance of the algorithm: the evaluation of a novel algorithm is ultimately supposed to give hints about how well the algorithm will perform in future applications. But applied researchers usually do not know the "true" cluster labels of their datasets, as otherwise there would be no need for clustering. Thus the applied researchers cannot use a "ground truth" to determine a good t_{max} value for their specific datasets, and will thus obtain worse results for their datasets than the performances reported in the original paper which introduced the cluster algorithm. We will further discuss this issue in Sect. 5.1.

An alternative to reporting the best t_{max} for each dataset individually is to look for a t_{max} value that leads to good performance for multiple datasets. For example, $t_{max} = 12$ yields reasonable performance values for Blobs, Two Moons and Iris. Thus, optimistic researchers might only report these three datasets with $t_{max} = 12$ and claim that this choice of t_{max} will perform well for future datasets. However, such a statement would likely be over-optimistic as $t_{max} = 12$ was chosen on only a few datasets, and considering the varied behavior of the different datasets for different t_{max} in Fig. 5.

Over-optimism can not only result from optimizing the hyperparameters of the novel algorithm, but also from simultaneously neglecting to optimize the hyperparameters of the *competing* algorithms. As an example, we compare Rock with DBSCAN on the Two Moons dataset, with the data parameters optimized for Rock from Table 1. Recall that in Sect. 4.1, we did not perform hyperparameter optimization, and instead used hyperparameter defaults or heuristics for the algorithms which could be reasonably justified (see also the appendix A): for Rock, $t_{max} = 15$ as in the original paper of Beer et al. (2019), and for DBSCAN, $minPts = 2 \cdot$#of dimensions, leading to $minPts = 4$ for Two Moons, and $eps = 0.2$. The AMI for Rock for this case is 0.7881 ± 0.1583 (mean and standard deviation over ten random seeds), see also Table 1. This mean value is different from the AMI value in Fig. 5 at $t_{max} = 15$, because a single seed was used for the latter. The AMI performance of DBSCAN was only 0.0007 ± 0.0024.

We then performed hyperparameter optimization for both cluster algorithms (with regards to the absolute AMI performance over ten random seeds). For Rock, we performed a simple grid search over $t_{max} \in \{1, 2, \ldots, 30\}$. The optimal performance is at the previously used default $t_{max} = 15$, thus again yielding a mean AMI of 0.7881 ± 0.1583. This is not surprising, given that $t_{max} = 15$ was used in Sect. 4.1 to optimize the data parameters of Two Moons such that Rock obtains superior performance (although the performance *difference* was used as the optimization criterion in that section). For DBSCAN, we performed hyperparameter optimization with the TPE, and obtained optimal parameters of $minPts = 41$ and $eps = 0.4$, leading to a performance of 0.8300 ± 0.0244, which is a major improvement over the previous performance of DBSCAN. Thus DBSCAN outperforms Rock after hyperparameter optimization. This demonstrates that if researchers decide to perform hyperparameter optimization for the cluster algorithms to be compared, they should conduct the optimization not only for their own algorithm, but also equally carefully for all competing methods.

Returning to the topic of data type optimization (Sect. 4.1), Fig. 5 also shows the potential for picking specific datasets for which Rock performs reasonably well (e.g. Blobs, Iris, Two Moons) and discarding the ones with worse performance (Digits, Rings). Again, *over*-optimization is marked by selective reporting: while no cluster algorithm can be expected to perform well on all types of data, it is still important to report data types for which a novel algorithm fails to detect clusters, to illuminate the limitations of the new method.

4.3 Optimizing the choice of competing algorithms

Here we revisit the results from Sect. 4.1 to analyze whether there is potential for picking specific competing cluster algorithms such that Rock appears better. For example, Fig. 3a–c show that Rock often performs better than DBSCAN, which was also due to neglecting hyperparameter optimization for DBSCAN, cf. Sect. 4.2. By picking suitable data parameter ranges, an over-optimistic researcher could praise the drastic performance improvement from Rock over DBSCAN. The same figures show that Rock is often better than Mean Shift. Thus, there is the potential for the following narrative: "Rock is an improvement of Mean Shift". As the figures show, this claim would sweep some caveats under the carpet. For example, the other competitors, k-means and spectral clustering, are (almost) as good as Rock for the Blobs dataset in Fig. 3b.

5 Discussion

We have illustrated that selective presentation of performance results can lead to over-optimistic assessment of a novel cluster algorithm. Neglecting to show limitations of a new algorithm can lead to users applying it in inappropriate settings for the algorithm, which leads to unusable results. In this section, we discuss potential further aspects of over-optimism that we did not focus on, but would be interesting to study in future work.

5.1 Hyperparameter tuning and development of the algorithm

As explained in Sect. 4.2, the current standard of reporting the performance of a novel algorithm with hyperparameters optimized to the clustering "ground truth" (e.g., with a grid search) is likely over-optimistic. Using the ground truth of datasets for performance evaluation of a novel algorithm has a further drawback: as the number of datasets labeled by experts is limited, researchers using these datasets optimize their algorithm's characteristics on these few labeled real world datasets, or alternatively use (unrealistic) synthetic datasets. Datasets such as Two Moons and Blobs are frequently used, but provide very limited information about how the cluster algorithm will perform in much more complex applied settings.

The optimization to a few datasets might not only concern the hyperparameters of the algorithm, but also the characteristics of the algorithm which are explored in the development phase. For example, Rock contains some "hidden hyperparameters" such as the growth rate of the number of neighbors considered in each iteration, or the weighting of the different nearest neighbours (Beer et al. 2019). These characteristics are not intended to be changed by the user, but were decided on by the researchers during the development of the algorithm. However, if such characteristics are optimized according to the performance on just a few selected datasets, then this might result in an over-optimistic "overfitting" effect.

5.2 Evaluation measure

For all our experiments in this paper we used the Adjusted Mutual Information (AMI) as measure for the quality of clustering. Other partition similarity indices such as the Normalized Mutual Information (NMI, Strehl and Ghosh 2002), Adjusted Rand Index (ARI, Hubert and Arabie (1985)), Accuracy and F1-measure are often used in the field (see also Albatineh et al. (2006), for an overview). They all range in $[-1, 1]$ resp. $[0, 1]$ and describe how well the clustering results correspond to a ground truth, but have slightly different behaviors (Pfitzner et al. 2009). These indices are also called *external validation* indices, because they require an externally known partition (the ground truth) for evaluation. Yet evaluating a clustering based on the given "ground truth" might not always be the best choice. There could be interesting cluster structures in the data which differ from the given "true" labels, particularly because there is no unique definition of what a "good" clustering is (Hennig 2015). Moreover, as pointed out above, many real world datasets do not come with given labels. Thus researchers might also use internal validation indices (Halkidi et al. 2015) which do not require knowledge of the "true" labels, but evaluate a clustering based on internal properties of the data alone. Popular internal indices which measure within-cluster homogeneity and between-cluster heterogeneity/separateness include the Average Silhouette Width index (Kaufman and Rousseeuw 2009), the Caliński-Harabasz index (Caliński and Harabasz 1974), and the Davies-Bouldin index (Davies and Bouldin 1979). Such indices can also be used for performance evaluation of novel clustering algorithms, yet they might be susceptible to the over-optimism mechanisms outlined above. For example, researchers could optimize datasets and data characteristics with respect to an internal index, such that this index indicates a good performance for the new cluster algorithm, analogous to the optimization with the AMI discussed in Sect. 4.1.

The multitude of possible evaluation criteria—external or internal – gives rise to another potential source of over-optimism: Researchers could try different measures and pick the one that is most favorable to their novel algorithm. While researchers might be understandably uncertain about which evaluation measure to choose, they should not try different measures and then pick only the most favorable one *after* having seen the results. Researchers should carefully consider before starting the experimental evaluation which performance criterion is of particular interest in the considered context. If multiple measures are tried, then these should all be reported.

5.3 Preprocessing

Preprocessing the data can significantly influence the results of clustering. In our study, we scaled all the datasets. There are different normalizations that may be applied to the data, as well as methods to remove outliers or noise to improve the clustering results.

To avoid over-optimism, researchers should refrain from trying different preprocessing methods and reporting only the one most favorable to their new algorithm. Moreover, the same preprocessing steps should be applied to all datasets and *for all compared cluster algorithms*. Otherwise, if only the new algorithm is combined with suitable preprocessing, it might have an unfair advantage. A clear distinction should be made between preprocessing steps and steps belonging to the new cluster algorithm.

5.4 Theoretical evaluation

While we focus on the experimental evaluation of cluster algorithms with simulated or real-world datasets, it would also be interesting to study over-optimism in the context of theoretical analyses of algorithms. For example, researchers often make claims about their novel algorithms which they prove mathematically. But they could use very specific assumptions to yield the desired results. It might not always be easy for readers to judge how unrealistic these assumptions are, i.e., to which extent the assumptions restrict the use of the algorithm in real-world applications. Authors should thus always make their theoretical assumptions very clear, and thoroughly discuss how restrictive they are.

While theoretical analyses can, in principle, be affected by over-optimism, they are often a vital part of the evaluation of novel cluster algorithms. Theoretical results, if carefully deduced, can give a more complete picture of the algorithm's capabilities. Authors who thoroughly analyze their novel algorithm from a theoretical perspective might also use this background knowledge to choose a suitable and clearly defined experimental study design, such that unintentional over-optimization in the experimental part of the analysis could sometimes be partially avoided.

6 Possible solutions

As we have illustrated, there might be a strong over-optimistic bias when introducing a new cluster algorithm. How can such a bias be avoided or corrected? We discuss three options that all researchers can consider using in their research: (1.) avoiding selective reporting and analyzing robustness, (2.) evaluating the new method on independent data, and (3.) performing neutral benchmark studies. Moreover, we discuss (4.) how changing incentives in research culture and the publication system (that are beyond the control of individual researchers) might help to reduce over-optimism.

6.1 Avoiding selective reporting and analyzing the robustness of the algorithm

Our results have shown that over-optimistic presentation ultimately requires a certain amount of *selective reporting*, i.e., reporting only specific scenarios in which the new algorithm performs well. This might happen if many different scenarios are tried and only the "best" ones are reported, while the others are buried in the file drawer. Researchers might also omit the analysis of certain scenarios a priori, for example, when only considering data simulated according to a specific model. Such constraints should be clearly explained, and the performance of the algorithm should not be oversold.

In the context of *model-based* cluster algorithms (see McLachlan et al. (2019) for an overview), selective reporting might be easier to detect. For example, if mainly datasets generated by the model of the newly developed algorithm are chosen, and/or the novel algorithm is compared with competing methods that were developed for the detection of clusters generated by other models, then the novel algorithm immediately has an advantage, which can be easily spotted. Nevertheless, there is still potential for an over-optimistic selection of datasets and comparative methods among all "reasonable" possibilities. Moreover, other potential sources of over-optimism discussed above, such as (hyper)parameter optimization, are also existent for model-based clustering. Readers and reviewers of articles about novel model-based cluster algorithms should keep this in mind, and the authors themselves must be careful to avoid over-optimistic choices.

Ideally, researchers should report scenarios in which their algorithm performed worse, to give a more realistic picture of the limitations of the novel approach. This may also require researchers to check the robustness of their algorithm (cf. Sect. 2.3): if the cluster algorithm is not robust with respect to certain data parameters, this should be honestly reported. Discussing the evaluation results for various parameter choices could also be beneficial as there is often not a single "best" choice and different parameters could be useful in different applications (Cerioli et al. 2018).

6.2 Validation on independent data

It is advisable to evaluate a new algorithm's performance on fresh data that was *not* used for developing the algorithm and assessing its performance (Jelizarow et al. 2010). As we have demonstrated in Sects. 4.1 and 4.2, looking for specific data parameters or tweaking the algorithm's hyperparameters might cause unintentional overfitting to the datasets used during the research process. As discussed in Sect. 5.1, overfitting to the used datasets could also concern the algorithm's characteristics that were engineered in the development phase. The algorithm might not perform quite as well on new data, which would constitute a more realistic assessment of its performance.

More realistic performance values might also be obtained by taking inspiration from supervised classification and splitting the used datasets into "training" and "test" sets (Ullmann et al. 2021). Then hyperparameters such as t_{max} are optimized on the training set, and the chosen t_{max} is evaluated on the test set to assess performance. This could partially avoid "overfitting" of the hyperparameters to the data. However, a) this splitting procedure does not say anything about the performance on genuinely new data/data from different distributions, and b) when using the ground truth for optimization on the training set, this does not solve the problem that applied researchers who wish to use the new cluster algorithm in practice usually do not know the ground truth of their datasets, and thus cannot use the hyperparameter optimization procedure of the original authors. Therefore, it is advisable for authors who introduce a new algorithm to discuss and evaluate criteria for hyperparameter choice that do not require the ground truth, for example internal validation indices. Such indices could be used to choose hyperparameters on the training set, and to evaluate the chosen hyperparameters on the test set to ensure that potential overfitting effects are detected.

6.3 Neutral benchmark studies

Awareness about the dangers of selective reporting and the importance of evaluation on fresh data might help to alleviate the problem of over-optimism. Academic teaching/training and illustrative studies such as ours can contribute to creating such awareness. Moreover, following guidelines for methodological computational research can help researchers avoid over-optimism (Boulesteix 2015). Ultimately, this will probably not solve the problem completely. Researchers are incentivized by the publication system to present their new algorithm favorably, which is unlikely to change in the short term (see 6.4). They are also more competent with respect to their own methods—and thus more likely to use them optimally than competing methods when conducting the evaluation. Thus, neutral benchmark studies are additionally required.

A neutral benchmark study is characterized by the comparison of existing algorithms (instead of the introduction of a new method), and neutrality of the authors, i.e., the authors do not have a vested interest in a particular method showing better performance than the others and are as a group approximately equally familiar with all considered methods, see Boulesteix et al. (2013, 2017) for an extensive discussion of these concepts. As mentioned in the introduction, neutral benchmark studies are less likely to suffer from over-optimism and usually offer a more realistic performance evaluation than studies presenting new methods.

In the field of clustering methodology, neutral benchmark studies are rarer than for supervised classification. Lately, however, there have been some advances: guidelines for performing benchmark studies for cluster algorithms were published in Van Mechelen et al. (2018). Following these guidelines, Hennig (2021) compared nine popular cluster algorithms, mainly with respect to various internal validation indices, but also regarding the recovery of the “true” clusterings. For an overview of previous cluster benchmark studies, see Van Mechelen et al. (2018) and Hennig (2021). In principle, the guidelines of Van Mechelen et al. (2018) could and should also be followed by non-neutral researchers who evaluate their new algorithm.

6.4 Changing incentives in the culture of research and the publication system

The three possible solutions presented so far are in principle accessible to individual researchers or teams of researchers. Ultimately, however, each researcher is subject to the constraints of the research and publication system. For example, researchers might hesitate to report limitations of their novel algorithm, because this could reduce their chances of getting published. Moreover, it can still be difficult to publish a neutral comparison study as many journals and conferences—stressing the importance of “novelty”—prefer studies introducing new methods (Boulesteix et al. 2018). In our view, changes in this attitude are necessary to further reduce over-optimism. Accepting neutral benchmark studies for publication should become more widespread. Furthermore, reporting limitations of novel algorithms should not be considered a “failure” and instead an integral part of a healthy research culture. Journals and conferences should actively encourage authors to report scenarios in which their new algorithm does not perform optimally, or at least should not consider such reporting to be a cause for rejection. At the same time, editors and reviewers play an important role in filtering manuscripts in which authors do not carefully justify their experimental choices and only present very specific settings, which may be a hint that the results could potentially be over-optimistic. It should be taken into account, however, that even when a persuasive justification is given, the authors might still have arrived at these choices by (intentional or unintentional) over-optimization.

7 Conclusion

We have shown that studies which introduce new cluster algorithms might be affected by over-optimistic presentation of the results. For illustrative purposes, we have demonstrated different over-optimism mechanisms using the recently developed Rock algorithm as an example. While this is a specific example, we believe that these mechanisms might similarly apply to other novel clustering algorithms. We have also given some recommendations for avoiding over-optimism. It is our hope that going forwards, these guidelines will be taken into account. After all, overselling of novel methods does not contribute to genuine scientific progress.

Acknowledgements We thank Oliver Langselius and Anna Jacob for making valuable language corrections.

Author Contributions Conceptualization: TU (Lead), AB (Supporting), MH (Supporting), A-LB (Supporting)
Methodology: TU (Lead), AB (Supporting), MH (Supporting)
Software: MH (Lead), TU (Supporting), AB (Supporting) Validation: TU (Lead)
Writing—original draft preparation: TU (Lead), AB (Supporting), MH (Supporting), A-LB (Supporting), TS (Supporting)
Writing—review and editing: TU (Lead), AB (Supporting), MH (Supporting), A-LB (Supporting), TS (Supporting)
Funding acquisition: TS (Lead), A-LB (Supporting) Supervision: A-LB (Lead), TS (Supporting).

Funding Open Access funding enabled and organized by Projekt DEAL. This work has been funded by the German Federal Ministry of Education and Research (BMBF) under Grant No. 01IS18036A. The authors of this work take full responsibilities for its content.

Availability of data and material All used datasets are publicly available in scikit-learn for Python (Pedregosa et al. 2011).

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

Code availability Our fully reproducible code is available at: https://github.com/thullmann/overoptimism-clust-algo.

A Appendix

In this appendix we give some details about the implementation outlined in Sect. 3.2. More information can be found in our fully reproducible code which is available at https://github.com/anonresearcher461/over-optimism. All experiment were performed with Python[6], version 3.9.5.

A.1 Adjusted mutual information (AMI)

Here we give the mathematical definition of the Adjusted Mutual Information Score (AMI, Vinh et al. 2010) which we use to compare the calculated clusterings with the "true" cluster labels. To define the AMI, we first discuss the entropy H of a single clustering and the Mutual Information (MI) of two clusterings. See Vinh et al. (2010) and Meila (2015) for more detailed explanations.

Let C and C' be two clusterings with k respectively l clusters. Let $n_{ij}, i = 1, \dots, k, j = 1, \dots, l$ the number of data points which are in cluster i of C and cluster j of C'. Let $n_{i\bullet}$ and $n_{\bullet j}$ be the respective marginal sums, and n the overall number of data points.

The entropy H of clustering C is defined as

$$H(C) = -\sum_{i=1}^{k} \frac{n_{i\bullet}}{n} log\left(\frac{n_{i\bullet}}{n}\right).$$

The entropy can be interpreted as the level of uncertainty associated with the clustering C. The Mutual Information (MI) of the clusterings C, C' is defined as

$$MI(C, C') = \sum_{i=1}^{k}\sum_{j=1}^{l} \frac{n_{ij}}{n} \log\left(\frac{n_{ij}/n}{n_{i\bullet}n_{\bullet j}/n^2}\right).$$

The MI measures to which extent knowledge of the clustering C reduces uncertainty about the clustering C'. The MI is a symmetric measure, and it holds that

$$0 \leq MI(C, C') = MI(C', C) \leq min(H(C), H(C')).$$

[6] https://www.python.org, visited: 05/31/21.

The MI can be normalized to ensure the measure ranges in [0, 1], yielding the Normalized Mutual Information (NMI):

$$NMI(C, C') = \frac{MI(C, C')}{avg(H(C), H(C'))}.$$

Different choices for the "average" avg are possible, e.g., the arithmetic mean, the geometric mean, the minimum or maximum. We use the arithmetic mean (Kvalseth 1987), which is the scikit-learn default.[7]

Both the MI and NMI tend to increase with an increasing number of clusters, even if the information shared mutually between the clusterings does not actually increase. To account for this effect, the MI can be adjusted for chance: the MI of C, C' is compared with the expected MI for two random clusterings drawn from a permutation model (see Vinh et al. (2010) for details). The Adjusted Mutual Information Score (AMI) is thus calculated as follows:

$$AMI(C, C') = \frac{MI(C, C') - E[MI(C, C')]}{avg(H(C), H(C')) - E[MI(C, C')]}. \tag{2}$$

The AMI attains its maximum value of 1 if the two clusterings are identical, and equals 0 if the MI between the two clusterings is equal to the MI value expected for two random partitions. Negative values occur if the agreement between C and C' is "worse" than chance.

A.2 Scaling of the datasets

All datasets used in our study were scaled with the scikit-learn standard scaler[8], by subtracting the mean and dividing by the standard deviation of each variable. That is, for each dataset $D = (x_{ij})_{i=1,\dots,n, j=1,\dots,d}$, with n samples and d dimensions, each entry x_{ij} is scaled according to

$$\frac{x_{ij} - \frac{1}{n}\sum_{i=1}^{n} x_{ij}}{\sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(x_{ij} - \frac{1}{n}\sum_{i=1}^{n} x_{ij}\right)^2}}$$

A.3 Details about the blobs dataset

The Blobs dataset[9] consists of isotropic Gaussian clusters, i.e., each cluster $k \in \{1, \dots, K\}$ (with K the number of generated clusters) corresponds to a Gaussian distribution with covariance matrix $\sigma_k^2 I_d$, where $\sigma_k^2 \geq 0$ and I_d is the d-dimensional

[7] https://scikit-learn.org/stable/modules/generated/sklearn.metrics.adjusted_mutual_info_score.html, visited: 05/31/2021.

[8] https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html, visited: 05/31/2021.

[9] https://scikit-learn.org/stable/modules/generated/sklearn.datasets.make_blobs.html, visited: 05/31/2021.

identity matrix. We chose a standard deviation of $\sigma_k = 3\frac{k}{K}$ for each cluster k. This generates different variances for the clusters, making some clusters more compact and thus easier to detect, and others more scattered and harder to find.

A.4 Bayesian optimization (BO) and the tree-structured parzen estimator (TPE)

BO approaches (see Shahriari et al. (2016) for an introduction) are popular for optimization problems of the type $\text{argmax}_{x \in \mathcal{X}} f(x)$, where $f : \mathcal{X} \mapsto \mathbb{R}$ is expensive to evaluate. In each step of a BO procedure, f is modelled with a *surrogate model*, based on a library of evaluations of f from previous steps: $((x^{(1)}, f(x^{(1)}), \ldots, (x^{(k-1)}, f(x^{(k-1)}))$. The surrogate model is used to construct an acquisition function, which is cheaper to evaluate and easier to optimize than f, yielding the optimal argument $x^{(k)}$. Then $(x^{(k)}, f(x^k))$ is added to the library, and the process is repeated by updating the surrogate model. The concrete surrogate model and the acquisition function of the TPE were chosen by Bergstra et al. (2011) such that optimization of the acquisition function ultimately leads to optimization of $x \mapsto l(x)/g(x)$, where $l(x)$, $g(x)$ are two Gaussian Mixture Models. $l(x)$ is fitted to the observations $(x^{(i)})_i$ that performed well so far, i.e., for which $f(x^{(i)}) > y^*$ for some threshold value y^*. $g(x)$ is fitted to the remaining observations. The threshold y^* is chosen as a quantile of the observed $y^{(i)} = f(x^{(i)})$ values, such that $p(y > y^*) = \gamma$ for a suitable $\gamma \in (0, 1)$. For more details on the TPE, see the original paper of Bergstra et al. (2011), the Optuna documentation[10], and our reproducible code.

A.5 Default settings for the hyperparameters of the cluster algorithms

For the analysis in Sect. 4.1 (optimizing datasets and data characteristics), we used defaults or heuristics for the hyperparameters of the cluster algorithms which a researcher could justify as "reasonable choices". For Rock, we chose $t_{max} = 15$, as in the original paper of Beer et al. (2019). For k-Means and Spectral Clustering we used the number of ground truth clusters for the parameter k and the default settings from scikit-learn. For DBSCAN, we followed Schubert et al. (2017) to set $minPts = 2d$ with d being the number of dimensions. Moreover, we set $eps = 0.2$, which can be seen as a sensible value, given that the samples were scaled to unit variance. For estimation of the bandwidth for Mean Shift we use the scikit-learn function estimate_bandwidth[11].

References

Akiba T, Sano S, Yanase T, Ohta T, Koyama M (2019) Optuna: A next-generation hyperparameter optimization framework. In: Proceedings of the 25rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp 2623–2631

[10] https://optuna.readthedocs.io/en/stable/reference/generated/optuna.samplers.TPESampler.html, visited: 05/31/2021.

[11] https://scikit-learn.org/stable/modules/generated/sklearn.cluster.estimate_bandwidth.html, visited: 05/31/2021.

Albatineh AN, Niewiadomska-Bugaj M, Mihalko D (2006) On similarity indices and correction for chance agreement. J Classif 23(2):301–313

Beer A, Kazempour D, Seidl T (2019) Rock-let the points roam to their clusters themselves. In: Proceedings of the 22nd International Conference on Extending Database Technology (EDBT), pp 630–633

Bergstra J, Bardenet R, Bengio Y, Kégl B (2011) Algorithms for hyper-parameter optimization. Adv Neural Inf Process Syst NIPS 24:2546–2554

Bischl B, Binder M, Lang M, Pielok T, Richter J, Coors S, Thomas J, Ullmann T, Becker M, Boulesteix AL, Deng D, Lindauer M (2021) Hyperparameter optimization: Foundations, algorithms, best practices and open challenges. arXiv preprint arXiv:2107.05847

Boulesteix AL (2015) Ten simple rules for reducing overoptimistic reporting in methodological computational research. PLoS Comput Biol 11(4):e1004191

Boulesteix AL, Strobl C, Augustin T, Daumer M (2008) Evaluating microarray-based classifiers: an overview. Cancer Inf 6:77–97

Boulesteix AL, Lauer S, Eugster MJ (2013) A plea for neutral comparison studies in computational sciences. PLoS ONE 8(4):e61562

Boulesteix AL, Stierle V, Hapfelmeier A (2015) Publication bias in methodological computational research. Cancer Informatics 14(S5):11–19

Boulesteix AL, Wilson R, Hapfelmeier A (2017) Towards evidence-based computational statistics: lessons from clinical research on the role and design of real-data benchmark studies. BMC Med Res Methodol 17:138

Boulesteix AL, Binder H, Abrahamowicz M, Sauerbrei W (2018) On the necessity and design of studies comparing statistical methods. Biometr J 60(1):216–218

Boulesteix AL, Hoffmann S, Charlton A, Seibold H (2020) A replication crisis in methodological research? Significance 17(5):18–21

Buchka S, Hapfelmeier A, Gardner PP, Wilson R, Boulesteix AL (2021) On the optimistic performance evaluation of newly introduced bioinformatic methods. Genome Biol 22:152

Caliński T, Harabasz J (1974) A dendrite method for cluster analysis. Commun Stat 3(1):1–27

Cerioli A, García-Escudero LA, Mayo-Iscar A, Riani M (2018) Finding the number of normal groups in model-based clustering via constrained likelihoods. J Comput Graph Stat 27(2):404–416

Chhabra A, Roy A, Mohapatra P (2020) Suspicion-free adversarial attacks on clustering algorithms. Proc AAAI Conf Artif Intell 34:3625–3632

Davé RN, Krishnapuram R (1997) Robust clustering methods: a unified view. IEEE Trans Fuzzy Syst 5(2):270–293

Davies DL, Bouldin DW (1979) A cluster separation measure. IEEE Trans Pattern Anal Mach Intell. PAMI-1(2):224–227

Dua D, Graff C (2017) UCI machine learning repository. http://archive.ics.uci.edu/ml

Ester M, Kriegel HP, Sander J, Xu X (1996) A density-based algorithm for discovering clusters in large spatial databases with noise. In: KDD'96: Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, pp 226–231

Ferrari Dacrema M, Boglio S, Cremonesi P, Jannach D (2021) A troubling analysis of reproducibility and progress in recommender systems research. ACM Trans Inf Syst 39(2):1–49

Fukunaga K, Hostetler L (1975) The estimation of the gradient of a density function, with applications in pattern recognition. IEEE Trans Inf Theory 21(1):32–40

Gan J, Tao Y (2015) DBSCAN revisited: mis-claim, un-fixability, and approximation. In: Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data, pp 519–530

Goodfellow I, McDaniel P, Papernot N (2018) Making machine learning robust against adversarial inputs. Commun ACM 61(7):56–66

Halkidi M, Vazirgiannis M, Hennig C (2015) Method-independent indices for cluster validation and estimating the number of clusters. In: Hennig C, Meila M, Murtagh F, Rocci R (eds) Handbook of cluster analysis. Chapman and Hall/CRC, Boca Raton, pp 616–639

Hennig C (2015) What are the true clusters? Pattern Recogn Lett 64:53–62

Hennig C (2021) An empirical comparison and characterisation of nine popular clustering methods. Adv Data Anal Classif. https://doi.org/10.1007/s11634-021-00478-z

Hubert L, Arabie P (1985) Comparing partitions. J Classif 2(1):193–218

Jelizarow M, Guillemot V, Tenenhaus A, Strimmer K, Boulesteix AL (2010) Over-optimism in bioinformatics: an illustration. Bioinformatics 26(16):1990–1998

Kaufman L, Rousseeuw PJ (2009) Finding groups in data: an introduction to cluster analysis. John Wiley & Sons, Hoboken, NJ

Kvalseth TO (1987) Entropy and correlation: some comments. IEEE Trans Syst Man Cybern 17(3):517–519

Lloyd S (1982) Least squares quantization in PCM. IEEE Trans Inf Theory 28(2):129–137

McLachlan GJ, Lee SX, Rathnayake SI (2019) Finite mixture models. Ann Rev Stat Appl 6:355–378

Meila M (2015) Criteria for comparing clusterings. In: Hennig C, Meila M, Murtagh F, Rocci R (eds) Handbook of cluster analysis. Chapman and Hall/CRC, London, pp 640–657

Ng AY, Jordan MI, Weiss Y (2001) On spectral clustering: Analysis and an algorithm. In: Proceedings of the 14th International Conference on Neural Information Processing Systems: Natural and Synthetic, pp 849–856

Norel R, Rice JJ, Stolovitzky G (2011) The self-assessment trap: can we all be better than average? Mol Syst Biol 7(1):537

Nuzzo R (2015) How scientists fool themselves-and how they can stop. Nat News 526:182–185

Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V, Vanderplas J, Passos A, Cournapeau D, Brucher M, Perrot M, Duchesnay E (2011) Scikit-learn: machine learning in Python. J Mach Learn Res 12:2825–2830

Pfitzner D, Leibbrandt R, Powers D (2009) Characterization and evaluation of similarity measures for pairs of clusterings. Knowl Inf Syst 19(3):361–394

Schubert E, Sander J, Ester M, Kriegel HP, Xu X (2017) DBSCAN revisited, revisited: why and how you should (still) use DBSCAN. ACM Trans Database Syst 42(3):1–21

Shahriari B, Swersky K, Wang Z, Adams RP, de Freitas N (2016) Taking the human out of the loop: a review of Bayesian optimization. Proc IEEE 104(1):148–175

Strehl A, Ghosh J (2002) Cluster ensembles–a knowledge reuse framework for combining multiple partitions. J Mach Learn Res 3:583–617

Tufte E (1983) The visual display of quantitative information. Graphics Press, Cheshire, CT

Ullmann T, Hennig C, Boulesteix AL (2021) Validation of cluster analysis results on validation data: a systematic framework. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery e1444

Van Mechelen I, Boulesteix AL, Dangl R, Dean N, Guyon I, Hennig C, Leisch F, Steinley D (2018) Benchmarking in cluster analysis: a white paper. arXiv preprint arXiv:1809.10496

Vinh NX, Epps J, Bailey J (2010) Information theoretic measures for clusterings comparison: variants, properties, normalization and correction for chance. J Mach Learn Res 11:2837–2854

Xu R, Wunsch DC (2010) Clustering algorithms in biomedical research: a review. IEEE Rev Biomed Eng 3:120–154

Yousefi MR, Hua J, Sima C, Dougherty ER (2010) Reporting bias when using real data sets to analyze classification performance. Bioinformatics 26(1):68–76