Received: 19 July 2018 | Revised: 28 April 2019 | Accepted: 12 June 2019

DOI: 10.1002/widm.1330

ADVANCED REVIEW

Method evaluation, parameterization, and result validation in unsupervised data mining: A critical survey

Albrecht Zimmermann

UNICAEN, ENSICAEN, CNRS, GREYC, Normandie Université, Caen, France

Correspondence
Albrecht Zimmermann, UNICAEN, ENSICAEN, CNRS, GREYC, Normandie Université, 14000 Caen, France.
Email: albrecht.zimmermann@unicaen.fr

Abstract

Machine Learning (ML) and Data Mining (DM) build tools intended to help users solve data-related problems that are infeasible for "unaugmented" humans. Tools need manuals, however, and in the case of ML/DM methods, this means guidance with respect to which technique to choose, how to parameterize it, and how to interpret derived results to arrive at knowledge about the phenomena underlying the data. While such information is available in the literature, it has not yet been collected in one place. We survey three types of work for clustering and pattern mining: (1) comparisons of existing techniques, (2) evaluations of different parameterization options and studies providing guidance for setting parameter values, and (3) work comparing mining results with the ground truth. We find that although interesting results exist, as a whole the body of work on these questions is too limited. In addition, we survey recent studies in the field of community detection, as a contrasting example. We argue that an objective obstacle for performing needed studies is a lack of data and survey the state of available data, pointing out certain limitations. As a solution, we propose to augment existing data by artificially generated data, review the state-of-the-art in data generation in unsupervised mining, and identify shortcomings. In more general terms, we call for the development of a true "Data Science" that—based on work in other domains, results in ML, and existing tools—develops needed data generators and builds up the knowledge needed to effectively employ unsupervised mining techniques.

This article is categorized under:

Fundamental Concepts of Data and Knowledge > Key Design Issues in Data Mining
Ensemble Methods > Structure Discovery
Internet > Society and Culture
Fundamental Concepts of Data and Knowledge > Motivation and Emergence of Data Mining

KEYWORDS

algorithmic comparison, clustering, parameter selection, pattern mining, result verification

This article is a significantly extended version of (Zimmermann, 2015).

1 | INTRODUCTION

An important aspect of Data Mining and Machine Learning research is its often *applied* or *engineering* nature. This does not mean that all work done therein does or should happen only in relation with a concretely defined real-life application, nor that such research cannot yield fundamental insights. But rather that we often build upon results of other disciplines, be they mathematics, physics, or, particularly in connection with image recognition and machine learning, biology, to develop what should be understood as *tools*, devices and algorithms that make it easier for humans to perform certain tasks. Tasks such as automating decision making in situations where data overwhelm human cognitive abilities, finding hidden regularities in the data, identifying relationships that should be investigated more thoroughly in an off-line setting etc. Tools typically require guidelines to make the most of their usage, and in the case of DM/ML tools, a practitioner will require at least three pieces of information:

1. Which algorithm to use given the available data, and the purpose of the operation.
2. How to parameterize the algorithm, which includes not only concrete parameter settings but also the choice of quality criteria, for instance.
3. How to interpret the results in light of the purpose.

ML/DM tools come mainly in two forms: (1) *supervised* methods that learn from labeled data how to *predict* labels for unseen data or how to *characterize* predefined subsets, and (2) *unsupervised* methods. A second category along which to group them has to do with the scope of their results: (a) they apply either to (almost) the entire data set—they are *global* in nature, such as classification models or clusterings, or (b) being *local*, they refer only to a (nonpredefined) subset of the data.

In practical terms, unsupervised methods hold arguably greater promise: labeling data is typically a time-consuming and expensive process, labeling errors cannot be avoided, and since the mechanisms leading to the label might change in the real world, one can rarely consider the work finally done. This can be alleviated by treating a variable as label that is already contained in the data, for example, the amount of money a customer spends on a shopping site, but this requires prior knowledge informing the data collection. Unsupervised methods avoid this bottle-neck.

The results of unsupervised mining can be exploited further in the domains whence the data were generated to, for instance, further research, improve logistics, or increase sales. A human expert might "just" need a well-defined subset of the data to turn his "I don't know"[1] into a much deeper insight into the field.

This notwithstanding, supervised model building—*classification* or *regression*—has seen the most progress, with occasional extensive evaluations performed and theories of learnability backing up empirical results. In addition, the presence of a label makes it often much easier to assess a method's performance. As a result, there are well-understood methods for giving guidelines for the use of supervised tools. The supervised form of pattern mining, often referred to as Subgroup Discovery (Klösgen, 1996), also benefits from these aspects to a certain degree.

In unsupervised data mining, the most effective means for developing the guidelines mentioned above are systematic evaluations, comparing different techniques against each other, evaluating different ways of deciding on how to parameterize methods, and compare derived results against the data. Such evaluations become the more important, the more tools there are available, and the more generally they can be applied. The space of possible combinations of techniques, parameter settings, and data sets is a search space such as any other, and the more entities in each dimension, the larger that space, and the more extensive explorations of the space are necessary. In this survey, we therefore pull together the papers that have studied those questions, both for unsupervised pattern mining, introduced in the seminal paper on frequent itemset mining (FIM) (Agrawal & Srikant, 1994), and for clustering (Jain, Murty, & Flynn, 1999).

In line with our remarks above, we have split the survey into three main sections. Given the differences in techniques, and in ways they are evaluated, we always discuss pattern mining and clustering in dedicated subsections. In Section 3, we will review to what degree techniques for different pattern mining tasks and for clustering have been compared to each other, and draw conclusions from the current state of the art. That section is somewhat parameter-agnostic in that we take the parameters chosen by authors of original work as a given. In Section 4, however, we will take a closer look on the empirical relationships that have been established between methods, data, and parameter settings to arrive at good (or not so good) and interesting results. In Section 5, we review the challenges that crop up when trying to draw knowledge about the underlying processes from results resulting from even the best-chosen and -parameterized mining processes. Readers that are already familiar with the situation regarding the different questions for pattern mining or clustering could therefore skip those sections.

What we find is that while there is important information and intriguing results in the literature, the overall picture remains murky. In particular, there are three large gaps in our understanding of pattern mining and clustering:

1. **We do not know how most unsupervised techniques actually perform quantitatively compared to each other!** Many unsupervised pattern mining algorithms are *rarely*, if ever, evaluated on additional data after they have been published. Clustering algorithms are often evaluated time and again on the same data sets, typically in comparison to newer techniques. Algorithms are rarely extensively compared against each other.
2. **We do not know how data set characteristics affect data mining algorithms, nor how to parameterize them well!** Both running times/memory consumption and the interestingness of mined patterns and models are influenced by parameter settings and characteristics of the data. Yet, there are not enough publications studying those relationships to give a comprehensive picture.
3. **We do not know how mined patterns/models relate to the generative processes underlying the data!** The current working interpretations of "interesting" recur to (objective or subjective) unexpectedness, or summarization/compression of the data. This undermines the interpretability of patterns and the applicability of derived knowledge to the real-life setting whence they originated.

These gaps in our knowledge undermine the necessary guidelines mentioned above: without an understanding of how data influence algorithm performance, a user is largely confined to guessing which technique to use. Without an understanding of how to parameterize the method, an operation might be running for weeks on end, a user might be overwhelmed by the amount of patterns, or might miss truly interesting phenomena underlying her data. And without an understanding of the way in which results reflect the generative processes underlying the data, patterns and clusters are at best only a thought-starter, and at worst stay at a superficial level. There is therefore need for more, and more extensive, evaluations in both pattern mining and clustering.

In addition to practical considerations, there are scientific reasons to close these gaps. A number of research domains find themselves challenged by what has been called a "reproducibility crisis," in particular the social sciences and psychology, but also life-sciences (Ioannidis, 2005). In short, this crisis refers to the fact that results, even ones published in highly reputable venues, turn out to be not verifiable or even get falsified shortly after having been published (Reaves, Sinha, Rabinowitz, Kruglyak, & Redfield, 2012). While there have been cases of outright forgery, for example, by Diederik Stapel or Yoshitaka Fujii, in most cases the cause is more subtle, for example, *p-hacking* (removing certain data points that prevent the result from achieving statistical significance), or conscious or unconscious data selection. Falsification is necessarily an important aspect of scientific research but ideally it is hypotheses that are being falsified, not empirical results.

We are not the only ones who have noticed problems with data mining evaluations. In fact, a few months before the submission of this manuscript, Van Mechelen et al. (2018) uploaded a preprint to arXiv decrying the lack of systematic evaluations in clustering research, and discussing challenges and best practices in detail. We discuss that paper, and how it relates to this survey, in some more detail in Section 6, after we have outlined the work that *does* exists with respect to the evaluation of clustering methods.

One reason why those unverifiable results *have* been identified in the first place is because of verification studies. Since it is obviously in the interest of the data mining community to avoid a reproducibility crisis of its own, reevaluating algorithms and comparing them against each other again on new data therefore renders an invaluable service to the community. In line with this, when we illustrate the knowledge gaps in Sections 3–5, we will highlight those papers that *have* questioned published results, and what they found.

As an example of a subfield in which comparisons and evaluations are much more common, we discuss the situation of *community detection* separately in Section 7, even though community detection can be considered a form of network clustering, strictly speaking.

An obvious question to ask at this point is why there is a lack of such comparison studies in unsupervised mining and as we will argue, an important contributing factor is the lack of data in general, and of data with controlled, diverse characteristics and known ground truth in particular:

- A *general lack* of data necessarily limits experimental evaluations and if no new data is added to the portfolio, algorithms cannot be meaningfully reevaluated.
- A *lack of diverse characteristics* means that we have seen algorithmic behavior only over a relatively narrow range of settings, and that we lack understanding of how small changes in such characteristics affect behavior.

- The *lack of data of which the ground truth is known*, finally, is the main factor that makes it so hard to fill in the third gap: supervised data contains a certain amount of ground truth in the labels, while this is missing for unsupervised settings.

In the spirit of directing readers to available data sources, we therefore continue the survey by reviewing the state of data availability for the discussed tasks in Section 8.

Computer science offers a potential solution to the data scarcity problem—artificial data generation. In surveying the state of data generation in Section 9, we find that there are considerable weaknesses in existing generators. While developing generators that lead to data with different characteristics could turn out to be relatively easy, knowing what kind of generative processes can be expected to occur in real-life data will be more challenging and such knowledge will often not be found with data miners but with real-world practitioners. As an example that we will discuss in some more detail later, real-life data often follows power law distributions whereas existing data generators and assumptions underlying mining approaches often assume normal, Poisson, or binomial distributions. This has to be remedied if we want to do more than propose tools that might or might not work.

Hence, we conclude the paper (in Section 10) by arguing that we need to add a deeper understanding of data—"data science" so-to-say—to the data mining research portfolio, a task that will require the collaboration with researchers and practitioners of other fields that currently is too often more statement than fact. As a thought (and maybe research) starter, we round this off by surveying the attempts at data generation that have been made outside computer science research, point towards methods from supervised and semi-supervised machine learning that could be adapted, and existing tools that should ease this effort.

Given the breadth of the topics we discuss in this survey, it will clearly depend on the reader on how they will navigate its contents. Figure 1 offers a number of options, showing a few short-cuts the informed reader could take. Before we begin the survey, however, we will outline the main concepts that we will discuss in this paper in the following section.

2 | PRELIMINARIES

This is a fairly nontechnical survey, and we assume that readers are familiar with most of the main concepts of pattern mining and clustering. In this section, we will nevertheless clarify the context within which we survey work.

2.1 | Mining and learning tasks

As we have mentioned in the preceding section, there are two main ways in which data mining and machine learning approaches can be differentiated: *supervised* versus *unsupervised* tasks, and *modeling* (or global) versus *pattern mining* (or local) tasks.

In the supervised setting, we assume that each data instance has at least one attached label, be it discrete, that is, a class label, numerical, or ordinal. In unsupervised settings, the information contained in a data instance is limited to its representation.

In modeling, the result of the data mining operation is supposed to characterize the *full data*, whereas pattern mining finds partial representations, the patterns, that identify and describe subsets that exhibit different characteristics from the full data.

This leads directly to four problem settings: in supervised modeling, the goal is to find a relationship between the data representation and instances' labels for the full data, often for predictive purposes, whereas supervised pattern mining finds descriptions of subsets of the data with an unusual label distribution.

Unsupervised modeling seeks to split the data into coherent subsets, which are assumed to have been generated by more or less distinct processes. Unsupervised pattern mining, finally, finds patterns that occur more (or less) frequently than one would expect based on the occurrence of their component parts, or that only occur in particular subsets of the data.

Supervised			**Unsupervised**
Model	**Classification**		**Clustering**
(all data)	Regression	Anomaly detection	**Summarization**
Pattern	Supervised descriptive	Semi-supervised learning	**Local pattern**
(some data)	Rule discovery		**Mining**

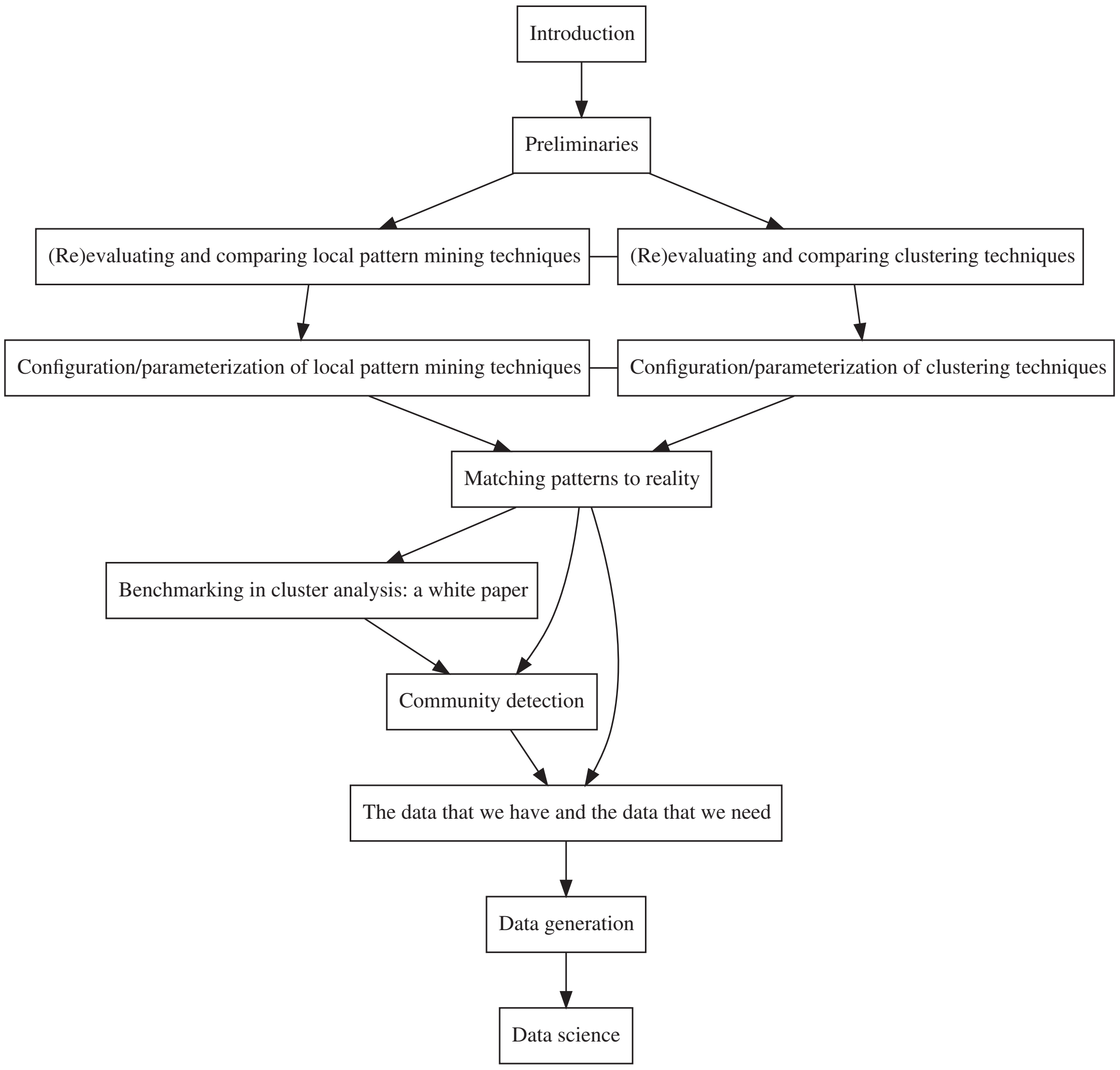
Introduction
Preliminaries
(Re)evaluating and comparing local pattern mining techniques
(Re)evaluating and comparing clustering techniques
Configuration/parameterization of local pattern mining techniques
Configuration/parameterization of clustering techniques
Matching patterns to reality
Benchmarking in cluster analysis: a white paper
Community detection
The data that we have and the data that we need
Data generation
Data science

FIGURE 1 Different paths through the survey

The preceding table shows a few prominent mining and learning tasks falling into the four different categories. For a number of reasons, supervised settings are out of the scope of this survey, with the settings that we discuss in **bold**.

- Classification models the relation of *all data* to a *discrete* label, regression to a *numerical* one.
- Clustering is concerned with assigning *all data* to groups of *similar* instances, while summarization aims to represent the data in such a way that storing them takes less space and/or querying them takes less time.
- Supervised descriptive rule discovery is a term that has been coined in (Kralj Novak, Lavrač, & Webb, 2009) to summarize different supervised pattern mining settings, such as Subgroup Discovery, contrast set mining, and emerging pattern mining.
- Local pattern mining (Hand, 2002; Morik, Boulicaut, & Siebes, 2005), finally, groups a large range of different tasks, ranging from itemset mining to more complex representations, and from frequent pattern mining to the use of sophisticated measures.

We have placed anomaly detection at the intersection of the categories since different methods draw on different approaches, yet tilted towards unsupervised techniques since in many cases, no explicit labels for anomalies are available. There is also the case of semi-supervised learning, where only some instances of the data are labeled, and this information is exploited to deal with unlabeled data. Finally, the informed reader will notice that we ignore settings such as multilabel learning.

2.2 | Evaluating mining techniques

There is a fundamental difference in how model building and pattern mining techniques are evaluated. The biggest difference is that the former attempt to find (an approximation to) a globally optimal model. The space of all possible models being very large, this can usually not be done in an exhaustive fashion but heuristics are employed instead. In clustering, one typically uses *external* evaluation measures that compare a found clustering to some predefined data partition (Rand, 1971), *internal* ones (Maulik & Bandyopadhyay, 2002), or both (He, Tan, Tan, & Sung, 2004; Kovács, Legány, & Babos, 2005). The second type often consists of calculating the sum of/average distance of objects within clusters and between clusters, a quantitative expression of the intuition that clusters should contain similar objects. External measures implicitly assume that there is an inherent structure in the data that the clustering technique should recover. In the case of summarization, better compression with respect to the full data is considered better. Notably, the truly minimal intracluster distance/best compression is typically not know.

The overwhelming majority of pattern mining techniques are guaranteed to find all local patterns satisfying certain constraints or quality requirements. They are therefore usually evaluated in terms of running times and memory consumption, and *if* they use heuristics (or sampling approaches), the result set can be compared to the set of exhaustively enumerated patterns to quantify how close to optimal it is. More recent work often has an optimization component, trying to find the (or a number of) best pattern(s). As in the case of heuristic mining, their results can be compared to exhaustively derived ones, and running times continue to be considered important. An important consideration in those types of evaluations is that *point-wise comparisons*, for example, relative running times at a given parameter setting, can be misleading when compared to *trend curves*, for example, how running times differences develop once a parameter valued is systematically varied.

3 | (RE)EVALUATING AND COMPARING MINING TECHNIQUES

There are fundamental insights into unsupervised data mining techniques. We know, for instance, a priori, that dense itemset data will lead to more frequent patterns and therefore more computational effort than sparse data. We know that strict formal concept analysis on real-life data will probably not result in good output compression since real-life data contains noise. Dense data can reduce the amount of formal concepts (or closed patterns), however, so that an interaction of the two has unpredictable effects on closed pattern mining. We are also aware of the curse of dimensionality: once vector data becomes very high-dimensional, all data points look equally (dis)similar and cluster assignment risks becoming arbitrary. Finally, we know that for reasons of how instance membership to clusters is decided, the K-MEANS algorithm has difficulty identifying nonspherical clusters. If K-MEANS fails to find good clusters on high-dimensional data, however, we do not know whether the shape of the clusters or the dimensionality is to blame. As the reader will notice, these statements are rather broad, and will not tell us much about concrete algorithmic proposals for addressing mining tasks or improving existing techniques.

In this section, we mainly review reevaluations and systematic comparisons of unsupervised mining techniques with the goal of providing insight into which techniques are to be preferred given certain data. Newly proposed mining techniques are evaluated for the first time in the paper where they are introduced. The choice of the used data and availability of comparison techniques limits what we can know about a technique's performance. Reevaluations occur mostly in papers in which other techniques are introduced, and compared to existing work. A new method is therefore typically compared against the most recently proposed method, which will have compared favorably to the one proposed before it, due to a positive-results bias in the field. This can become problematic both because it invites unjustified generalizations and it creates implied transitive chains of performance. If technique *B* performs better than technique *A* on certain data, and technique *C* performs better than technique *B*, it is assumed that *C* also performs better than *A*.

3.1 | Local pattern mining

We first take a look at local pattern mining approaches, encompassing the four pattern classes itemsets, sequences, trees, and graphs (with an honorable mention of episodes). Even though this subfield is relatively young, there are already too many publications to discuss all of them, so the following paragraphs will highlight papers that introduced certain evaluation modalities, those that reported results that contradicted or complemented existing results, and of course actual reevaluations and comparisons. We would like to stress that none of the following is intended to disparage authors' work but only to illustrate how difficult it can be to draw definitive statements from the current state of the literature. As we mentioned in Section 2.2, the techniques discussed in this section return all patterns satisfying certain constraints, and evaluations have therefore been performed in the form of running time/memory consumption measurements.

3.1.1 | Itemset mining

The seminal FIM paper (Agrawal & Srikant, 1994) used an artificial data generator to evaluate their approach, **Apriori**, on more than 45 data sets, generated by varying parameters. Notably, all those data share a common characteristic—they are sparse. Zaki (2000) (which introduced **Eclat**) used the same data generator to generate about half as many data sets, also all sparse. Han, Pei, and Yin (2000) (proposing **FP-Growth**), on the other hand, generated only two of those sparse data sets. Both (Han, Pei, & Yin, 2000; Zaki, 2000) reported better running times than **Apriori**. That these were problematic results, was shown empirically by Zheng, Kohavi, and Mason (2001). When comparing the artificial data to real-life data at their disposal, Zheng et al. noticed that the latter had different characteristics, with Poisson-distributed transaction lengths in the artificial data, and distributions following power laws in the real data. An experimental comparison of the follow-up algorithms showed that *claimed improvements in the literature did not transfer to the real-life data*—the improvements had been an artifact of the data generation process.

A second interesting result from itemset mining is that of **Charm** (Zaki & Hsiao, 1999), based on **Eclat**, and **Closet** (Han et al., 2000), based on **FP-Growth**. Both approaches mine closed frequent itemsets, a subset of frequent itemsets that carries the same amount of frequency information but can be mined more efficiently. Efficiency gains will be more pronounced for dense data and Zaki et al. therefore augmented three artificial data sets used with five real-life dense one. Pei et al. used only a subset of those data sets (one sparse artificial, two dense real-life) and reported run-time improvements over **Charm**. In the journal version of the **Charm** paper, Zaki and Hsiao (2002) reported that while **Closet** is faster than **Charm** on the sparse data set and one of the dense sets used by Pei et al., an advantage that disappears when support is lowered, it is slower for the other data.

> While we do not discuss supervised mining, we would like to quote from (Mutter, 2004) to illustrate the importance of implementations: "Our implementation of CBA cannot always reproduce the results out of the CBA paper [11]. Using the trick in line 17 of Figure 4.3 we are able to reproduce some of the results. In our opinion CBA uses other unpublished features as well. We have kindly received a latest copy of the CBA executable from Liu et al. [11]. The result obtained by using the executable and the results in the paper differ within a large scope and the result of the executable do not always outperform the paper results. In some cases the executable results are even worse."

We show the relative results of these comparisons in Figure 2 (left-hand side). The graphs in this figure show what we have referred to as "transitive chains" before. They also show cycles, which indicate that, depending on the study, one algorithm was found to be superior or inferior to another one. Finally, undirected edges indicate that two techniques perform roughly the same. These graphs (and the ones that will follow) are mainly intended to quickly summarize the detailed information in the text, and to give some visual indication of nonstraight-forward results.

The take-away message already from those early experiments on pattern mining is that algorithms should be evaluated on data having a wide range of characteristics, ideally defined in a more fine-grained manner than "dense" versus "sparse," or "small" versus "large." Additionally, comparing algorithms on those data to which they have not been applied yet can significantly add to our understanding. Unfortunately, this is not a trend observable even in FIM papers. Instead, many of them use only a few data sets without specifying the selection criteria.

An additional subtle problem is that we do not compare abstract algorithms but concrete implementations. The organizers of the two workshops on "Frequent Itemsets Mining Implementations" (FIMI) (Bayardo, Goethals, & Zaki, 2004; Goethals & Zaki, 2003) clearly had this in mind, and some of the most informative comparisons can be found in the proceedings of those workshops. More recently, Kriegel, Schubert, and Zimek (2017) demonstrated that different implementations of the same algorithm can exhibit running time differences of up to three orders of magnitude. The FIMI organizers required that all submissions be evaluated on the same collection of data sets: artificial data, the dense data sets used by Zaki et al. and the data introduced by Zheng et al. for comparison experiments. This includes the work introducing the LCM algorithm (Uno, Asai, Uchida, & Arimura, 2003). By comparing against five techniques, including **Apriori**, **Eclat**, and **FP-Growth**, they performed an informative reevaluation of existing work on new data, notably finding that while **Eclat** is typically more efficient than **FP-Growth**, this is not true for all data and occasionally depends on the minimum support. The relative performance

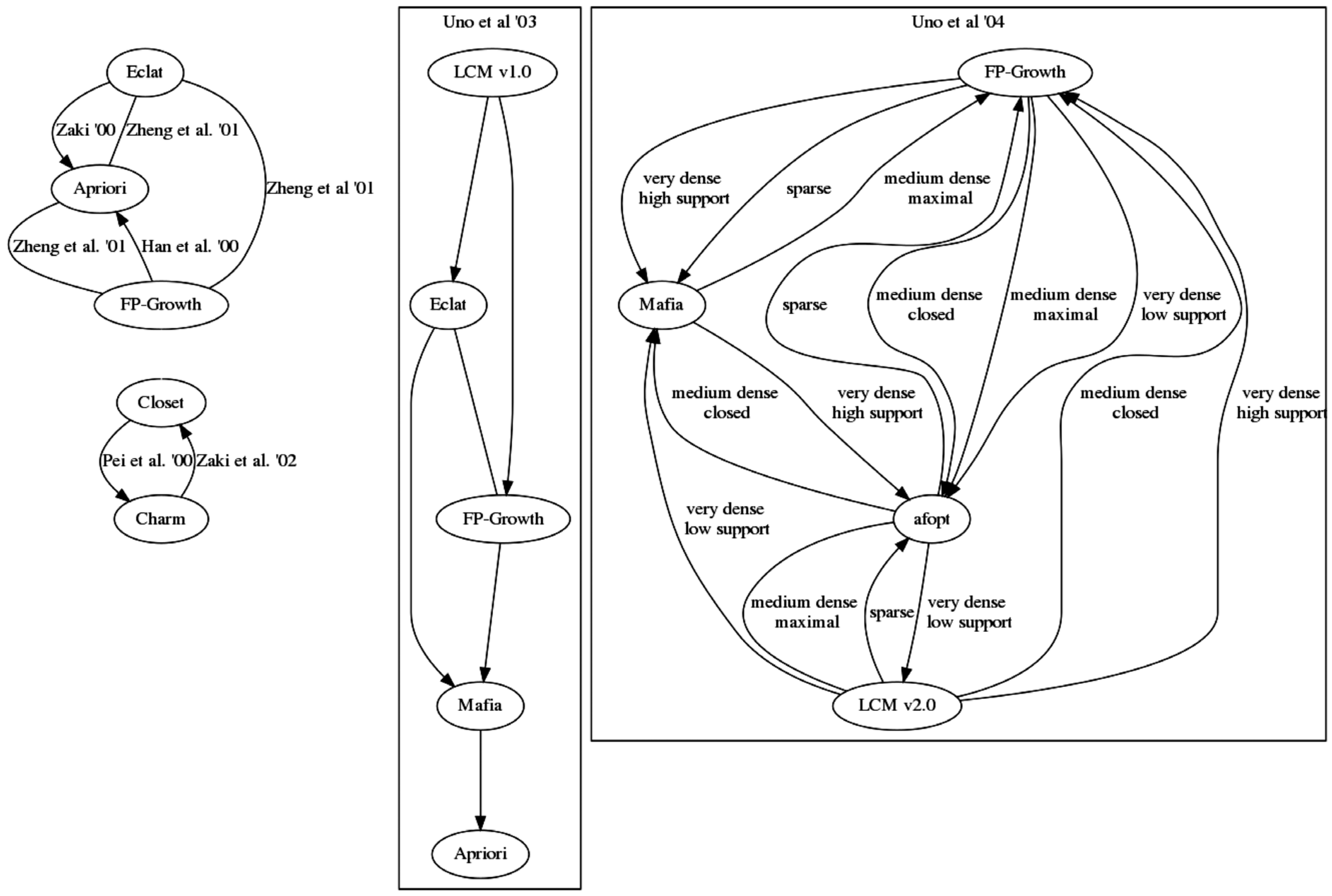

FIGURE 2 Relative performance graphs for several FIM publications. A directed edge from an algorithm to another means that the first algorithm ran faster, an undirected edge that they perform similarly

graph is also shown in Figure 2 (center). Remarkably enough, though, while Uno et al. refer to **Charm** as the most efficient closed itemset miner at the time, they did not compare to it due to deadline reasons, nor to **Closet**. When proposing further improvements to LCM in the second FIMI workshop[2] (Uno, Kiyomi, & Arimura, 2004), they compared to **afopt** and MAFIA, neither of which compared to **Charm**, breaking the transitive chain in this manner. They report results broken down by data density and show that this can change the order of algorithms, something that we attempt to capture on the right-hand side of Figure 2.

We have spent quite a few words on frequent itemset mining for two main reasons: the first is that this was the foundational pattern mining task, and that with the two FIMI workshops, there were clear attempts at systematic evaluations. The second is that the different algorithmic approaches developed early on have formed the foundation for more recent techniques that tackle different itemset mining problems, a trend that is also true for other pattern languages. Any insight into the differing performance of FIM techniques can therefore also inform our understanding of techniques that are based on them. In the following paragraphs, we will quickly discuss other pattern languages.

3.1.2 | Sequence mining

Sequence mining was first formalized and proposed at roughly the same time as FIM by the same authors: **Apriori-All** (Agrawal & Srikant, 1995), and GSP (Srikant & Agrawal, 1996). Originally, the authors used a data generator, motivated by similar reasoning as in (Agrawal & Srikant, 1994), with three real-life data sets added in the second publication. They generated fewer data sets than for the itemset case with the main difference between them being length and number of sequences. Sequence data sets are usually not discussed in terms of density but the data used in those two papers had between 10,000 (artificial data) and 71,000 items. Given the overall sampling procedure, one would therefore expect each item to occur relatively rarely, resulting in "sparse" data. To the best of our knowledge, the three real-life data sets are not in the public domain.

Han, Pei, Mortazavi-Asl, et al. (2000) (**FreeSpan**) used only artificial data, and reported improvements over GSP. Zaki (2001) (**Spade**) used the artificial data and added a data set from a planning domain that can be considered "dense" (77 items). A notable finding of that comparison to GSP is that performance improvements are much more pronounced on the artificial data (up to factor 80) than on the planning data (factor 2). Pei et al. (2001) (**PrefixSpan**) also used only artificial data, and report improvements over GSP and **FreeSpan**. The journal version (Pei et al., 2004) added one relatively sparse real-life data set (Gazelle) and reports systematic run-time improvements over **Spade**, **FreeSpan**, and GSP. That work exploited the fact that artificial data allows arbitrary increases in data set size and remarkably enough, GSP becomes the fastest algorithm for one of those data sets at large data set sizes and 1% minimum support. As in the case of FIM, sequence mining also moved towards closed patterns. Wang and Han (2004) (**Bide**) used three real-life data sets: Gazelle as well as two much denser ones, which contain amino acid sequences. Not only are those two data sets much denser but also much smaller and contain much longer sequences. Contrary to the results in the **PrefixSpan** paper, however, they report that **Spade** is systematically *faster* than **PrefixSpan** on Gazelle and even faster than closed sequence mining techniques for certain support values. There was no comparison between **Spade** and the other techniques on dense data sets. As in the case of FIM methods, we have summarized this information in Figure 3.

To the best of our knowledge, the most comprehensive comparison of sequence mining techniques has been undertaken by Mabroukeh and Ezeife (2010). Unfortunately, while they survey 13 techniques, they only compare six, representatives of the leaves in their taxonomy. Most of the data is artificially generated but using at most 120 items (and as few as 20), that is, rather dense data. Reported running times are only point estimates, and the results are too varied to discuss (or show) in detail but they report that SPAM (Ayres, Flannick, Gehrke, & Yiu, 2002), an algorithm that *did not terminate at all* on large data sets, becomes very competitive once the data is very dense and support thresholds not too low.

3.1.3 | Episode mining

Sequence mining was transferred from the transactional setting to finding recurrent patterns—*episodes*—in single large data sequences by Mannila and Toivonen (1995), with follow-up work in (Mannila, Toivonen, & Verkamo, 1997). An interesting aspect of this subfield is that data are rarely reused and algorithms are often not compared to each other. In 1997, Mannila et al. used two proprietary data sets—telecommunication alarms and browsing data—together with public-domain data: two text data sets (the second of which is a filtered version of the first), and amino acid sequences, to compare two techniques they had proposed (**WinEpi** and **MinEpi**). Insights from that paper include that the two techniques give rise to different patterns, and behavior on the text data is very different from that on the alarm data. Casas-Garriga (2003) used the amino acid data and a different text data set but *did not directly compare* their approach to the earlier techniques. Méger and Rigotti (2004) used publicly available earthquake monitoring data but did not compare to other approaches. Atallah, Gwadera, and Szpankowski (2004) used a publicly available Walmart data set and did not compare to any other technique. Laxman, Sastry, and Unnikrishnan (2005) compared their technique to **WinEpi** on artificially generated data in which short patterns are embedded

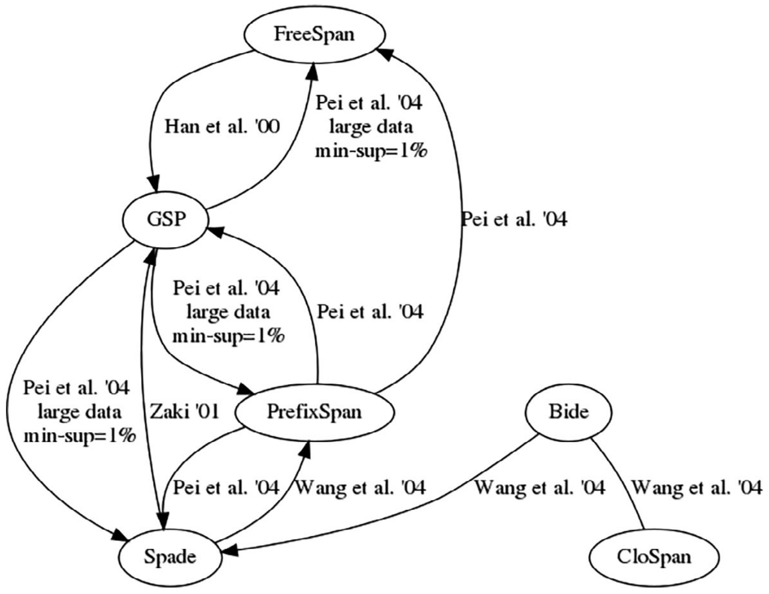

FIGURE 3 Relative performance graph of different FSM publications. Directed edges from one algorithm to another indicate that the former ran faster than the latter, edges without direction indicate similar behavior

in noise, and report large running time improvements. They claim improvements over **MinEpi** via transitivity. A proprietary GM data set is not used for comparison but only to report some quantitative results. The only work in which a data generator was used to generate data having a range of characteristics and compare different techniques is our own (Zimmermann, 2014), and we found that *temporal constraints have more impact than pattern semantics*, that is, whether episodes have to be closed and/or can have over-lapping occurrences, contrary to claims in the literature. There are not enough pairwise comparisons to show a relative performance graph.

3.1.4 | Tree mining

Tree mining is a field that has been somewhat under-explored. The seminal paper (Zaki, 2002) (**TreeMiner**) introduced an artificial data generator intended to mimic web browsing behavior and also used real-life browsing logs (CS-Log). In comparing their proposed approach to a baseline they report much more pronounced run time improvements on the artificial data. There exists a rather extensive comparison performed by Chi, Muntz, Nijssen, and Kok (2005), in which they compared a total of eight tree mining algorithms. There are different tree pattern definitions, and the comparison is therefore split accordingly. A first result is that **TreeMiner** performs much better than FREQT (Asai et al., 2004) on the artificial data down to a certain minimum support, below which both its running times, and memory consumption peak sharply. On CS-Log, three techniques, **uFreqT-New**, **HybridTreeMiner**, and **uFreqT** perform very similarly in terms of running times, something that disappears on a dense NASA multicast data set (MBONE), where the order becomes **PathJoin**, **uFreqT-new**, **HybridTreeMiner**, **uFreqT** from fastest to slowest. Their trend curves have the same shape, though. The same four algorithms on artificial data generated with a different data generator (introduced in (Chi, Yang, Xia, & Muntz, 2004)) perform very similar to each other, except for **PathJoin**, which slows down significantly for relatively small maximal tree sizes. When increasing the size of the data set generated with Zaki's generator, the running times of **uFreqT-new** and **HybridTreeMiner** stay almost flat, while that of **uFreqT** increases sublinearly. In the third set of experiments, Hybrid**TreeMiner** runs out of memory on CS-Log below a certain threshold, while **FreeTreeMiner** can keep going but also has running times more than an order of magnitude worse. This is an interesting result in the sense that it completes the results of the original evaluation in the **HybridTreeMiner** publication. On a second artificial data set generated with Zaki's generator, the authors report very similar running time trend lines for the three tree mining algorithms. When finally generating wide flat trees, running times of the depth-first **HybridTreeMiner** rises more sharply than that of the breadth-first **FreeTreeMiner** and eventually exceeds it. This comparison also cannot be summarized in a relative performance graph but for a different reason to the one mentioned above: where before we did not have enough pairwise comparisons, the current one is too detailed, breaking down into data characteristics, pattern semantics, and algorithmic frameworks.

3.1.5 | Graph mining

The fourth well-explored pattern class is that of graph patterns. Kuramochi and Karypis (2001) introduced both the FSG algorithm and an artificial data generator inspired by the one used in (Agrawal & Srikant, 1994). Notably, while they mention AGM (Inokuchi, Washio, & Motoda, 2000), they did not compare to it. Yan and Han (2002) evaluated their newly introduced algorithm **gSpan** against FSG on those artificial data showing consistent running time improvements. However, those are point estimates, for a single fixed support value (0.01). In addition, they used a molecular data set, PTE, and report running time improvements between 15 and 100 times. FFSM was introduced by Huan, Wang, and Prins (2003), and the authors report *consistent speed-ups* over **gSpan** on artificial data and the two smaller subsets (active, moderately active) of the DTP AIDS data set. A peculiarity of that paper is that while the authors claim running time improvements on the moderately active subset, the figure contained in the paper shows **gSpan** to be faster. The authors of **Gaston** (Nijssen & Kok, 2004) performed a rather extensive initial evaluation, comparing on two artificial data sets, the multicast data set, as well as five molecular data sets, ranging from very small (PTE: 340 molecules) to rather large (250 k). They report significant run time improvements over **gSpan**, especially for lower supports, which they conjecture to be due to their mining trees before graphs. In addition, they show FSG to be quickly overwhelmed on the real-life data, yet the *differences in running time between FSG and* **gSpan** *on PTE are much lower* than in (Yan & Han, 2002), in the order of 2–3. In a parallel to the LCM papers mentioned above, the authors report not having been able to acquire an implementation of FFSM in time for a comparison they themselves deem "the most interesting one."

That comparison came to pass in (Wörlein, Meinl, Fischer, & Philippsen, 2005), where the authors themselves implemented four algorithms: **gSpan**, **MoFa** (Borgelt & Berthold, 2002), FFSM, and **Gaston**, and compared them on seven molecular data sets. All of these techniques explore the search space depth-first. The authors reimplemented the algorithms to be able to benchmark different algorithmic components but for the sake of this survey we only focus on the relative running time results. The most interesting result is that they find FFSM *typically to be slower* than **gSpan**, contrary to the result in (Huan et al., 2003). **Gaston** is fastest, except for low support values on the largest data set. Using their own artificial data generator that allows them to control edge density, they report that **MoFa** shows steeply increasing running times while the other algorithms are virtually indistinguishable. In their conclusions, they challenge claims about algorithmic aspects that had been made in the literature. Chi et al. (2005) also evaluated graph mining algorithms, and report experiments in which **Gaston** runs out of memory on CS-Log below a certain threshold whereas FSG can continue, yet also has running times more an order of magnitude worse than the other technique. On a second artificial data set generated with Zaki's generator, the authors report very similar running time trend lines for the three graph mining techniques, with **gSpan** second-fastest after **Gaston**. When data set size is increased, however, FSG outperforms **gSpan**. When finally generating wide flat trees, running times of the depth-first algorithms **Gaston**, **gSpan** exceed those of the breadth-first FSG. Nijssen and Kok (2006), finally, generated artificial data and manipulated molecular data sets, and find for four out of six data sets that FFSM performs better than **gSpan**, and even better than one **Gaston** variant. Their most notable finding is that there is no strong relation between run times and the efficiency of the graph codes, as had been claimed in the literature, yet that **gSpan's** code is most efficient. We have summarized some results in the relative performance graph shown in Figure 4.

3.1.6 | Pattern set mining

A relatively recent entry to the field is *pattern set mining*, where the goal is to reduce the often very large result set of a Pattern Mining operation to a smaller set of nonredundant, informative patterns. Pattern set mining is somewhat related to model building, particularly summarization, since it tries to identify an optimal pattern set. As a consequence, many pattern set mining techniques have taken inspiration from model building approaches. When we surveyed the state-of-the-art with colleagues in a 2011 tutorial (Bringmann, Nijssen, Tatti, Vreeken, & Zimmermann, 2011), we remarked that those techniques had not been compared to each other yet, and this situation has not changed much in intervening years. One study that did compare techniques for compressing itemset data, (Smets & Vreeken, 2012), evaluated two methods: (1) a two-phase approach that first mines frequent sequences and selects from the result in a second step; (2) an iterative one they introduce that evaluates pairwise combinations of itemsets currently in the pattern set as candidates for extending the set. The authors use 19 data sets in total, a mix of FIMI data, UCI data, text data, and others. They report that the iterative method outperforms the two-phase one in terms of compression, in particular for very dense data. In terms of running times, the iterative method pays for the better quality on 10 of the data sets but the authors do not characterize the data sets that are harder for it.

A study looking at the same setting for sequential data is (Lam, Mörchen, Fradkin, & Calders, 2014). The authors introduced two methods, a two-phase one and a method heuristically finding an approximation to the best compressing pattern to add to a partial set. They compared the two methods among themselves and to a previously proposed approach called SQS on seven supervised data sets as well as two unsupervised data sets, a text data set representing abstracts of JMLR papers and an artificial one with embedded patterns. They report that SQS is slower than the two-phase method in large data or data with a

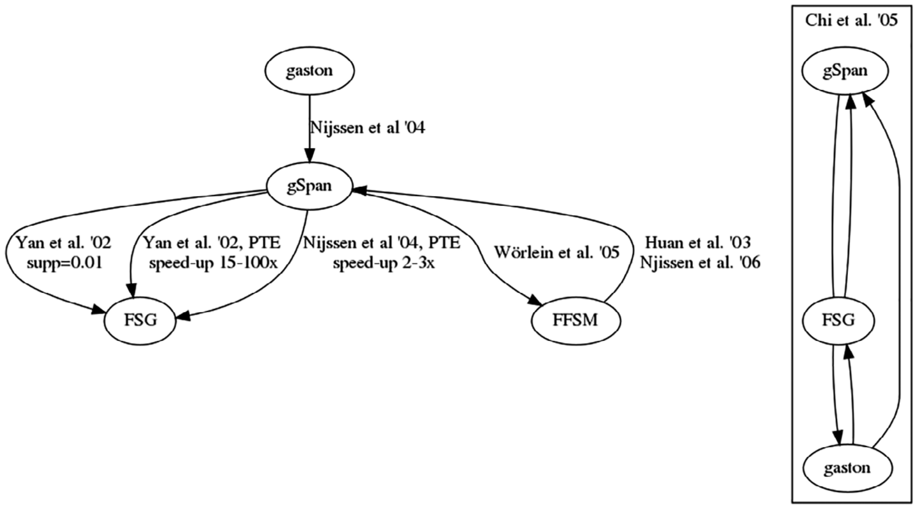
gaston
Nijssen et al '04
gSpan
Yan et al. '02 supp=0.01
Yan et al. '02, PTE speed-up 15-100x
Nijssen et al '04, PTE speed-up 2-3x
Wörlein et al. '05
Huan et al. '03 Njissen et al. '06
FSG
FFSM
Chi et al. '05
gSpan
FSG
gaston

FIGURE 4 Relative performance graph of different FGM publications. Directed edges from one algorithm to another indicate that the former ran faster than the latter, edges without direction indicate similar behavior

large alphabet but faster otherwise, and that both approaches are slower than the heuristic approach. In terms of the quality of results, they report very similar results on the text data, and that SQS shows worse recall than the heuristic method for the artificial data. In terms of compression, which was not evaluated on the artificial data, they report improvement by the heuristic techniques over the two-phase one on four data sets, and roughly equivalent results on the other four.

3.1.7 | Conclusion

As the preceding paragraphs show, systematic comparisons and reevaluations of pattern mining methods can give important insights into the relative running times of different local pattern mining methods. Strictly speaking, any information about the relative performance of two (or more) techniques is also limited to the data on which it was performed. Using new and additional data, moving from sparse to dense (or vice versa), or changing data set sizes can add to or even contradict the results derived from initial evaluations. While there are enough such studies in the field to pique one's curiosity, there is as of now not enough information to draw clear conclusions about the relative superiority and applicability of different techniques. In particular, it is not clear how to unify those results: if there are two studies reporting contradictory results, which one is to be believed? In the best case scenario, a third independent study backs one of the two others but even in that case, two-to-one does not exactly an iron-clad case make. Given that most of the comparative studies we discussed are relatively old, and that a variety of new techniques has been developed since, our understanding of how different pattern mining algorithms perform in terms of running times is spotty. Finally, especially for the more extensive comparisons, there are often almost as many techniques as data sets.

3.2 | Clustering

Clustering is a much older research field, with first algorithms introduced in the late 50s (Steinhaus, 1956), and the collection of developed techniques much larger (Jain et al., 1999). This makes it both harder and more important to do large-scale comparisons time and again to provide road maps through the field. As we mentioned above, the white paper by Van Mechelen et al. (2018) decries the lack of systematic evaluations in clustering research, and discusses challenges and best practices in detail. We discuss that paper, and how it relates to this survey, in some more detail in Section 6. Fortunately, there *have* been a number of papers experimentally comparing different methods to each other, particularly in the context of document clustering. Since in most evaluations of clustering techniques, the focus has been on recovering the correct number of clusters, as well as correct cluster assignments of data points, we treat this as the default and mention explicitly if other criteria were used.

3.2.1 | Vectorial data clustering

In a striking parallel to FIM, Milligan (1980) generated 108 error-free low-dimensional sets (at most eight dimensions), showing relatively clear cluster structure, and then proceeded to introduce different types of errors: outliers, perturbations of the distances matrices, addition of random noise dimensions. He also evaluated the effect of non-Euclidean distances and variable standardization. He used those data to evaluate four variants of k-Means (MacQueen et al., 1967), and 11 hierarchical agglomerative (HAC) algorithms, that is, algorithms that produce *dendrograms*, cluster-trees, starting from clusters containing only a single object. The arguably most interesting result is that for error-free data, two variants of k-Means only achieved around 90% accuracy (with the MacQueen formulation falling below) whereas HAC performed much better. Outliers, however, throw the HAC methods off much more than the k-Means ones. For these particular data, internal criteria align rather well with external ones.

Mangiameli, Chen, and West (1996) used 252 artificially generated data sets to compare self-organizing maps (SOM) (Kohonen, 1990) to HAC algorithms. Introduced errors are similar to the ones in the preceding study. Their data is again low-dimensional. They report that the SOM approach *overwhelmingly turns out to be best* in terms of identifying actual clusters. Notably, they show that increasing cluster dispersion makes things harder for HAC methods while SOM stays relatively stable, and that SOM is resistant to outliers and irrelevant dimensions. This is an intriguing result, given that SOMs are not widely used for clustering nowadays, even though the age of deep learning has dawned.

Given the right kind of distance measure, clustering techniques can be applied to just about any data, and Steinbach, Karypis, and Kumar (2000) compare a hierarchical algorithm to k-Means on document data, using eight document data sets. They report that HAC methods perform poorly and explain this with k-Means implicitly using a global optimization function. Authors from the same group used 12 document data sets, including the eight from the previous work, to compare different hierarchical methods in (Zhao & Karypis, 2002). They define two differences between methods, the first being bisecting versus agglomerative methods, the second in the optimization function. Clusterings are evaluated with respect to clusters' agreement with the classes defined in the data. Their main result is that *bisecting methods outperform agglomerative ones*, which is remarkable in itself given that bisecting hierarchical methods are rarely discussed nowadays for computational reasons. A second result is that what the authors call "constrained agglomerative trees," that is, approaches where a preliminary clustering is found via partitioning, and those clusters then combined, perform even better than purely bisecting ones. In a sense, this result anticipates the recently proposed **Cobra** system (Van Craenendonck, Dumancic, & Blockeel, 2017) and derived variants.

In (He et al., 2004), three different clustering paradigms, **k-Means**, SOM, and a neural network-based one (ART-C), are compared on two two-dimensional (2D) artificial data sets with clear cluster structure and one text data set contained in UCI (and used by Zhao et al.). The latter technique is evaluated with respect to four similarity measures. They report that one can identify the correct number of clusters for **k-Means** by interpreting internal criteria but caution that their approach is not easily generalizable. On the text data, they report SOM to perform slightly *worse* than **k-Means**. The ART-C approach created less homogeneous clusters but recovered document classes better.

In addition to comparing **k-Means** and **DBScan** (Ester, Kriegel, Sander, Xu, et al., 1996) on three artificially generated data sets of different difficulty, Kovács et al. (2005) also evaluate internal *evaluation criteria* for clustering results. They report that **DBScan** finds the correct clusters for the harder data sets, whereas **k-Means** does not, and that some criteria are not able to indicate when a clustering technique has correctly recovered the underlying clusters. The comparisons discussed so far are summarized on the right-hand side of Figure 5.

Verma and Meilă (2003) compare five spectral clustering algorithms and an single-link HAC baseline on one artificial and two real-life data sets, the NIST handwritten digits data set and gene expression data. The artificial data has ideal clustering structure but includes noise. On these data, multiway spectral algorithms perform best and are most resistant to noise, with the HAC method performing worse. For the real-life data, they report no significant difference between different spectral clustering methods, and on the gene expression data the HAC method is competitive with the spectral clustering approaches. The graph for this evaluation can be found on the right-hand side of Figure 5.

Halkidi and Vazirgiannis (2008) compared **k-Means**, **Cure**, **DBScan**, and a version of **Chameleon** (implemented in CLUTO) on three artificial data sets exhibiting challenging geometry, and report that **DBScan** recovers the underlying cluster structure best.

3.2.2 | Graph clustering

Given that most clustering methods only need a *similarity* value between data points, one can also work directly on structured data, using alignment scores, distance measures, or kernels. We have not found any comparisons of clustering algorithms on

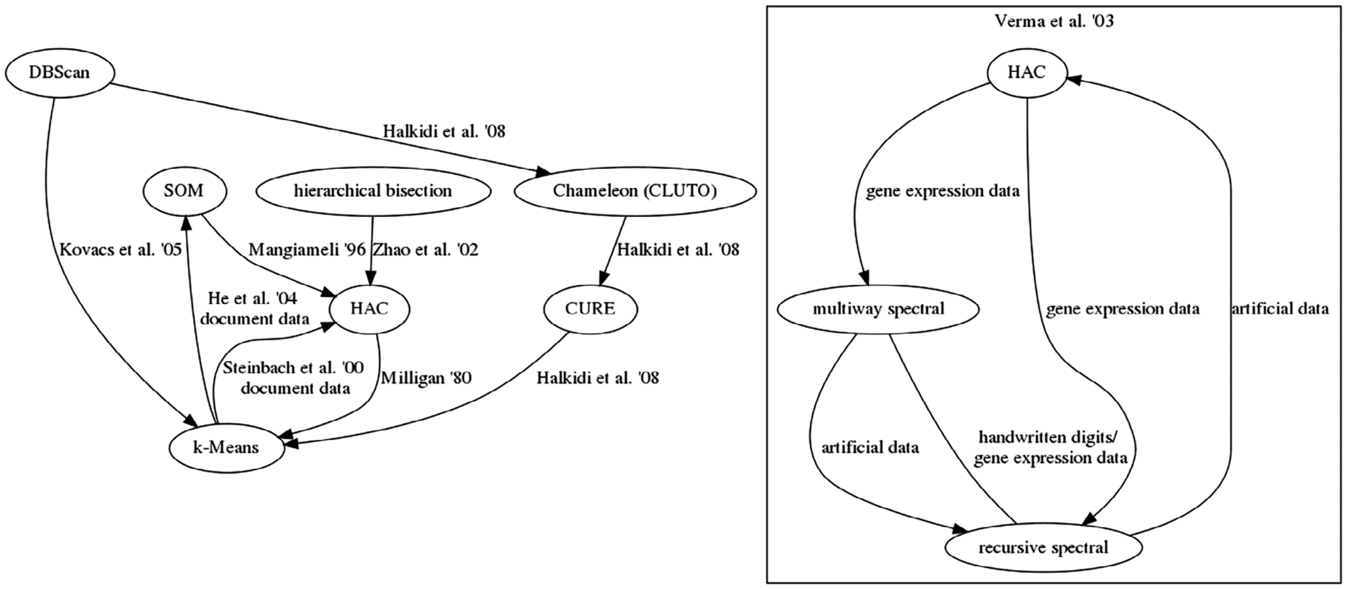

FIGURE 5 Relative performance graph of different clustering evaluations

sets of nonvectorial data. There is a different setting, however, where a single large network is used, and the clustering task consists of segmenting it into subcomponents.

It is this latter setting that has been evaluated in (Brandes, Gaertler, & Wagner, 2007), where the authors evaluate two methods, Markov Clustering (MCL) and Iterative Conductance Cutting (ICC), and a third one they proposed, Geometric MST Clustering (GMC), on a variety of artificially generated random networks, using different probabilities for inner/between cluster edges. They report that MCL is much slower than the other two techniques, and that ICC has a running time advantage on dense graphs. With respect to quality, they report that all techniques create clusterings of acceptable quality, with MCL and ICC somewhat better at recovering the original cluster structure, and GMC creating better-connected clusters.

In (Mishra, Shukla, Arora, & Kumar, 2011), two algorithms, MCL and Restricted Neighbourhood Search Clustering (RNSC), are evaluated and compared on 53 and 22 networks artificially generated by two different generators. Their evaluation is unusual since they focused on running times instead of cluster quality. They find that RNSC is faster than MCL for graphs of up to 1800 vertices, gets outperformed for up to 5,000 vertices, after which MCL running times increase significantly. They also report that MCL profits from dense graphs.

Brohee and Van Helden (2006), finally, focused on protein–protein interaction networks, and compare four techniques, MCL, RNSC, Super Paramagnetic Clustering, Molecular Complex Detection, on 41 networks that they derived from ground-truth data by adding/deleting edges randomly, as well as six networks derived from high-throughput experiments. They report that MCL is robust to edge noise, that correctly choosing optimal parameter settings is less important for RNSC than for the other techniques, that MCL gives superior results on the six high-throughput networks, and that the other two methods are not competitive.

3.2.3 | Current day challenges: big data and stream clustering

Focusing on algorithms that are able to handle *Big Data*, that is, data exhibiting the three Versus—**V**olume, **V**ariety, **V**elocity—the authors of a more recent work (Fahad et al., 2014) select five methods, **Fuzzy-CMeans**, Birch, Denclue, OptiGrid, EM, and evaluate them on 10 data sets, most of which have a large number of points. Instead of fixing a single evaluation criteria, the authors report a number of them and include running time comparisons. A remarkable result is that the venerable EM algorithm (Dempster, Laird, & Rubin, 1977) does best in terms of external criteria, followed by **Fuzzy-CMeans**. EM also performs well with respect to internal criteria and is joined by **Denclue** and **OptiGrid**, with those three methods also doing well with respect to the stability of found solutions. As is to be expected, however, EM runs very slowly compared to other techniques, among which **Denclue** is fastest.

Pereira and de Mello (2011) evaluated three stream clustering techniques on 90 artificial data sets falling into three categories: low-dimensional, low-dimensional with concept drift, and high-dimensional. The clusters in the data were elliptical and rather well-separated. They report that **CluStream** does best but is computationally more expensive. Carnein, Assenmacher, and Trautmann (2017) evaluated 10 stream clustering algorithms on four 2D artificial data sets of differing difficulty, as well as three large and relatively high-dimensional real-life data sets. Since the artificial data are not conceived as streaming data, they process the instances one by one in a random order, repeating this several times to avoid bias. They use the adjusted Rand index (ARI) to evaluate cluster agreements. The result are too varied to be easily summarized but they report that **DBStream**, which does worst on one artificial data set, consistently performs best on the other three, but also that its results depend heavily on the processing order. On the KDD99 Cup data, it is instead BICO that performs best, whereas **D-Stream** and **DBStream** show the best quality on a sensor data set. In their final conclusion, they stress the good performance of **DBStream**.

3.2.4 | Conclusion

As in the case of Pattern Mining evaluations, there are intriguing results to be found in systematic evaluations and comparisons of clustering methods, such as conflicting results or the information that methods that have fallen into disregard have been evaluated as performing very well in the past. But also as in the case of pattern mining, there is not enough overlap with respect to used data and evaluated methods to synthesize the results into a holistic view on the applicability of different clustering techniques. This impacts more strongly on our understanding of the relative performance of clustering methods since there is a much larger range of methods available.

4 | CONFIGURATION/PARAMETERIZATION

Let us entertain a thought experiment for a moment, assuming that the situation surveyed in the preceding section has been rectified and we have a good understanding of which algorithm is most efficient given certain data, or which one is most likely to recover an underlying cluster structure. That a pattern mining algorithm is more efficient than others does not preclude it from running for an extended time, however, and clustering algorithms can, for instance, be lead astray by requiring the wrong number of clusters to be discovered. It is therefore necessary to know how to best parameterize the chosen technique and in this section, we survey studies that have evaluated the effects of such parameterization. We use "parameterization" in a wide sense, including not only parameter values such as the minimum support or number of clusters, but also the choice of quality measures, similarity measures, stopping criteria etc.

4.1 | Local pattern mining

The pattern mining techniques we discussed in Section 3.1 are all frequent pattern mining techniques, the most-explored and most-published subfield of pattern mining. The frequency of patterns alone is not a good indicator for the quality of patterns, as the community has discovered early on. Instead, found patterns either need to be scored additionally using, for example, association and surprisingness measures (De Bie, 2011; Lenca, Meyer, Vaillant, & Lallich, 2008; Tan, Kumar, & Srivastava, 2002; Vaillant, Lenca, & Lallich, 2004), or the full result set needs to be reduced using pattern set mining methods (Bringmann & Zimmermann, 2009; De Raedt & Zimmermann, 2007; Knobbe & Ho, 2006; Lam et al., 2014; Smets & Vreeken, 2012; Tatti & Vreeken, 2012b; Vreeken, van Leeuwen, & Siebes, 2011). Complex association measures can usually not be used to prune the search space directly, instead being used to filter the results of a frequent pattern mining operation. In a similar manner, quite a few pattern set mining methods work in a two-step manner, first mining frequent patterns and then running the pattern set mining method on the resulting set of patterns. The amount of patterns derived from the first step will therefore have a strong influence on the running time of the second step. In fact, as we discussed in the subsection on pattern set mining above, the advantage of techniques that do not postprocess a result set typically stems from the ability to select lower-support patterns. Similar reasoning obviously holds for filtering via association measures, and in both cases running times of the frequent pattern mining operation itself will add to the full running times of data exploration.

It is therefore important to be able to assess how long a pattern mining operation runs and how many patterns it returns given certain data and certain parameter settings since this would allow to set parameter values to reduce both. Similarly, focusing a mining operation in such a way that resulting patterns have a higher probability to be statistically interesting would allow to reduce unneeded filtering effort. At the minimum, such knowledge calibrates users' expectations as to when they can expect the first results of their analysis operation.

4.1.1 | Threshold settings

In pattern mining, several papers established a relationship between the total number, distribution of length, and distribution of counts of mined frequent, closed, and maximal itemsets and algorithmic running times at different support thresholds (Flouvat, Marchi, & Petit, 2010; Gouda & Zaki, 2005; Ramesh, Maniatty, & Zaki, 2003; Zaki & Hsiao, 2002). Those results were derived using (subsets) of the data sets collected for FIMI. An important insight of this work is that the number of itemsets alone is less informative than their overall distribution with respect to the support thresholds for which certain algorithms perform well. Apart from the fact that such research does not exist for structured data and patterns, those approaches are faced with the obvious problem that extensive mining, at potentially extensive running times, is needed before the relationship can be established.

An improvement consists of estimating support distributions and running times based on partial mining results, or sampled patterns, as has been the approach of (Boley, Gärtner, & Grosskreutz, 2010; Boley & Grosskreutz, 2008; Geerts, Goethals, & den Bussche, 2005; Lhote, Rioult, & Soulet, 2005; Palmerini, Orlando, & Perego, 2004).

Lhote et al. discuss theoretical databases, either created by a Bernoulli or Markovian process, admitting that the former option is unrealistic and the latter not sufficient to model real-life databases. They prove that the number of frequent patterns for a given absolute threshold is polynomial in the number of transactions and exponential in the number of items, and for a relative threshold polynomial in the number of items. Palmerini et al. propose using the entropy information of two-itemsets at different support thresholds to identify how sparse or dense a data set is. They offer two ways of exploiting this information: (1) adapting pruning strategies to the density, or (2) using regression to estimate the number of frequent itemsets at a given support, and then decide on a support value based on the desired number of itemsets. They evaluate their proposal on seven data sets of different density, one of which has been generated by the QUEST generator. Geerts et al. prove upper bounds for the number of future frequent itemsets and the maximal depth of the search, based on the level a breadth-first search is on, and the number and involved items of frequent itemsets at that level. The further one advances in the search, the tighter the bounds become. They show the agreement of the bounds with patterns mined on four data sets, one of which has been generated by the QUEST generator. Boley et al. use Monte Carlo Markov Chains to sample itemsets, count their support, and derive frequency plots. This information can be used to set support thresholds to avoid the exponential explosion. They show the agreement of their estimates on eight data sets of varying density.

These works need less information from the mining process than the ones mentioned before but they need information from mining/sampling nonetheless. In addition, they only provide pointers to avoid the combinatorial explosion. Whether a threshold setting that allows the operation to finish in a reasonable amount of time will lead to (potentially) *interesting* patterns, is largely unexplored.

> Taking another detour from unsupervised pattern mining, we would be amiss not to mention the Coenen and Leng (2005) paper on class-association rule mining. In it, the authors show that the standard parameter settings (minimum support 0.01, minimum confidence 0.5) for this task do typically *not* give optimal results but that those depend on the data. It has, unfortunately, not led to a change in how parameters are set in papers on the topic.

A notable exception concerned with mining sequences under regular expression constraints can be found in (Besson, Rigotti, Mitasiunaite, & Boulicaut, 2008). By sampling patterns fulfilling data-independent constraints under assumptions about the symbol distribution (i.e., null models), they derive a model of background noise, and identify thresholds expected to lead to interesting results, that is, results that diverge from the expected support derived from super- and subpatterns. Given that many association measures reward precisely this kind of deviation from expectation, one could use this information to set both minimum and maximum supports to return only a subset of the result set that promises a high probability of interesting patterns. A similar idea can be found in (van Leeuwen & Ukkonen, 2014), which uses sampling and isotonic regression to arrive at *pattern frequency spectra* for FIM. They offer two contributions: first, a method for estimating the number of frequent itemsets for a given data set and minimum support threshold. Such an estimate, combined with the knowledge developed in the studies at the beginning of this section, could be used to decide support thresholds. Second, a method for estimating the total number of frequent patterns for all possible support thresholds, the frequency spectrum. By comparing this spectrum to one constructed from data in which all items are independent allows to zero in on those threshold settings for which the result set is expected to contain interesting patterns. They performed their evaluation on FIMI and UCI (Dheeru & Karra Taniskidou, 2017) data. The authors claim that their method can be extended to other pattern types but we are not aware of corresponding work.

Those two papers come closest to actually giving guidance for parameter selection but are limited to frequency thresholds. Yet, the *blue print* for building up knowledge about the interplay of data characteristics and parameter settings—for instance by leveraging the work done in Meta-Learning (Vilalta & Drissi, 2002) for storage in experiment databases (Vanschoren, Blockeel, Pfahringer, & Holmes, 2012)—is there at least, but requires to have a large collection of well-characterized data sets, *or* evidence for current data being representative.

4.1.2 | Quality measures

As we mentioned above, we consider the choice of association measure also to be part of parameterization. Tan et al. (2002) discuss a total of 21 different association measures in theoretical terms. They augment this analysis by generating 10,000 random contingency tables, use the different measures to rank them, and calculate correlations. This evaluation is further sharpened by successively removing contingency tables that correspond to low-support patterns. Their most interesting finding is that for unconstrained contingency tables, 12 of the measures cluster into a single group, with the rest clustering into two pairs of over-lapping clusters. That means that in many cases the choice of association measure is somewhat arbitrary, as least compared to alternatives in the same equivalence class. Tightening support thresholds leads to additional grouping of association measures. Vaillant et al. (2004) performed similar work on a slightly different set of association measures, finding four clusters, whereas Lenca et al. (2008) added “intelligibility” to assess whether a user could easily interpret a measure's score and changes to it. They add two decision criteria, depending on whether counter-examples are tolerated to a certain degree, and assign weights to different conceptual criteria. They report the *intensity of implication* and *probabilistic discriminant index* as best-ranked for an expert that tolerates counter-examples, *Bayes factor* for an expert that does not, and *Loevinger* as well-placed in both lists.

Such association measures are limited to the data from which patterns have been derived, whereas in supervised settings, validation sets can be used to assess the quality of resulting models, and parameters adjusted accordingly. There is work based on related ideas, which derive *null models* directly from the data, already found patterns, or user knowledge and use statistical testing to remove all those patterns that are not unexpected with respect to the null hypothesis (De Bie, 2011; Gionis, Mannila, Mielikäinen, & Tsaparas, 2007; Milo et al., 2002), or use multiple comparisons correction and hold-out sets (Webb, 2007). To evaluate whether either association measures or such methods for mining significant patterns find truly *interesting* patterns would require either expert feedback, which is not available for most data, or data for which ground-truth patterns or processes are known.

4.1.3 | Conclusion

The question of how the outcome of unsupervised pattern mining operations is related to data set characteristics and parameter settings has attracted interest from early on but only in recent years have papers emerged that would allow to derive guidelines for parameter settings. An unavoidable step seems to be partially mining or sampling the data, and while the studies estimating frequency plots or spectra show good agreement with empirical results, this has always been shown on a limited collection of data sets. The results for quality measures, which indicate that there is much redundancy in practical terms, are more robust since they do typically do not depend on specific data.

4.2 | Clustering

In clustering, in general there are decisions to be made about one or several of the following: the objective function to be optimized, the similarity measures between data points, the number of clusters to be found. Especially the last parameter will depend strongly on the data and the underlying group structure, and internal criteria are often used to identify a "sweet spot." Increasing the number of clusters that will have to be found, for instance, will initially decrease intracluster dissimilarity and intercluster similarity as data can be separated into purer and purer subpopulations. At some point, very similar subpopulations will be split apart and this yields diminishing returns. It should therefore probably not be surprising that much effort has been spent on identifying criteria that consistently allow to identify the correct number of clusters.

4.2.1 | Internal validation criteria

The literature on validation measures is extensive, with different authors intending to address different aspects of quality assessment, and proposing improvements over (perceived) shortcomings of other approaches. Before discussing the evaluation and comparison of different measures in the next section, we list and describe those measures in this section to aid the reader in understanding the similarities and differences. Detailed discussions listing many, yet not all, of the measures that we will discuss in the following can be found in (Charrad, Ghazzali, Boiteau, & Niknafs, 2012; Vendramin, Campello, & Hruschka, 2010) (including formulas). Most internal criteria take into account three characteristics of the found clustering solution: (1) the compactness of clusters, or how *similar* instances within a single cluster are, (2) the separability of clusters, that is, how *dissimilar* instances belonging to different clusters are, and (3) the total number of clusters.

Centroid-based criteria

A number of criteria exploit cluster centroids, implicitly assuming that clusters are spherical (or elliptical).

Calinski-Harabasz (CH) This measure divides the *trace* of the between-cluster distances matrix by the trace of the within-cluster distances matrix. To avoid overfitting by more and more, and therefore more and more fine-grained, clusters, the term is normalized via multiplication with $(N - k)/(k - 1)$, with N the number of points in the data and k the number of clusters. Depending on the literature, a distance matrix trace is also referred to as the *scatter or dispersion* matrix of the data, and can be interpreted as the distance of points from the centroid of their cluster or the centroid of the full data. CH is intended to reward compactness (of clusters) and the selected solution is the one achieving the maximal value.

Davies-Bouldin (DB) considers pairs of clusters and divides the sum of their average within-cluster distances (or scatter) by the distance between their centroids. For each cluster, the maximal such quotient is chosen, representing the worst-case scenario for it, that is, a situation where compactness or separability are worst. The solution for which the sum over all clusters' quotients is minimal is selected.

$\mathcal{I}$ is the sum of distances of all points to the *data centroid*, divided by the sum of distances for all points to their *cluster centroids*, multiplied by the maximal distance between any two cluster centroids. To normalize, this value is divided by the number of clusters, and the solution that maximizes the measure is selected.

Score uses similar quantities as CH, averaged distances of cluster centroids to the data centroid, as well as averaged within-cluster distances. The sum of those values is used as the negated exponent in the exponential function, the result of which is subtracted from 1.0. The larger the value, the better the solution.

Wemmert-Gançarski divides each point's distance from its cluster centroid by its distance to the next-nearest centroid, sums over all points in a cluster and subtracts that sum from the cluster's cardinality. The maximum of that subtraction and 0 is averaged over all clusters. The selected solution is the one that maximizes this measure.

KCE multiplies the number of clusters with the total sum of the distances of points from their respective cluster centroids. The selected solution is the one that minimizes this measure.

WB-index is **KCE** divided by the weighted sum of distances between cluster centroids and the data centroid (cluster scatter). It also should be minimized.

S_Dbw is a complex measure using the standard deviations (σ) of the full data and of individual clusters. It rewards lower σ for clusters, as well as clusters whose points have lower distance to the centroid than the cluster's σ while **not** being closer to the mid-point between two cluster centroids. The solution minimizing the measure is selected.

General distance-based criteria

Other measures do not take the short-cut of cluster centroids but interpret distances between data points directly.

Silhouette involves the average distance of each point to all others in its cluster, as well as the minimum average distance of that point to all objects in another cluster. The former is subtracted from the latter and the result normalized by the larger of the two values. This is summed over all points in the data and the solution maximizing this sum is selected. This maximization, again, rewards compactness and separability. A simplified version calculates distances between points and cluster centroids instead.

Duda and Hart (DH) assumes a two-cluster setting, and divides the sum of pairwise within-cluster distances for both clusters by the sum for a single cluster merging the two. Solutions are accepted or rejected based on a significance value. For settings with $k > 2$, DH can be applied iteratively, making it well-suited for hierarchical clustering approaches.

Dunn's also considers pairs of clusters and divides the minimal distance between any two points that are in different clusters by the maximal distance between any two points that share *any* cluster (that cluster's diameter). For each cluster, the minimal such quotient is chosen, and the entire clustering assessed via the smallest of those quotients in turn. The solution maximizing this value is selected. There are three remarks to be made: (1) the proposed numerator is the *single-link* merging criterion often used in hierarchical agglomerative clustering, (2) instead of summing over the entire clustering, it is represented by only a single distance, and (3) increasing the number of clusters is penalized implicitly. By modifying the set distance between two clusters and/or the diameter calculation, variants of Dunn's can be defined that correspond to the complete/average link strategies.

C-Index exploits pairwise distances between points, as well as the number of pairs sharing clusters. This latter number n_w is calculated as $\sum_{i=1}^{k} \frac{|C_i|(|C_i|-1)}{2}$. The sum d_w adds up the n_w pairwise distances of all points where both points are in the same cluster. $\min(d_w)$ adds up the n_w *smallest* pairwise distances, no matter whether the points share a cluster or not, and $\max(d_w)$ the *largest* distances. The measure itself is calculated as $\frac{d_w - \min(d_w)}{\max(d_w) - \min(d_w)}$ and the solution minimizing it is selected.

Gamma compares all pairwise *within-cluster* distances to all pairwise *between-cluster* distances. The number of times one of the former is smaller than one of the latter is counted (referred to as *consistent*), as well as the opposite cases (*inconsistent*), and a final value calculated as $\frac{consistent - inconsistent}{consistent + inconsistent}$. The solution maximizing the measure is selected.

Point-biserial employs similar quantities as the two preceding measures. It subtracts the average *within-cluster* distance from the average *between-cluster* distance. This value is multiplied with the square root of the product of n_w and the equivalent number of between-cluster pairs, divided by $N(N-1)/2$. The resulting value, finally, is divided by the standard deviation over all pairwise distances. The measure should be maximized.

COP index divides the average within-cluster distance for each cluster by the distance to the furthest neighbor in another. The full measure consists of the weighted average for all clusters, and the solution minimizing it is selected.

Rubin's index calculates the log value of the quotient of the determinant of the scatter matrix of the full data and the determinant of the within-cluster scatter matrix. The original authors propose comparing solutions of k and $k + 1$ clusters and selecting the solution for which the *increase* of the log value is largest.

Hartigan's index also compares clusterings of size k and $k + 1$, $k \in [1, N - 2]$ by dividing the trace of the cluster-scatter matrix of the former by the latter. Subtracting 1 and multiplying with $N - k - 1$ gives the final value. This is another measure that is well-suited to hierarchical approaches, and the maximal difference between clusterings of consecutive size is interpreted as indicating the appropriate number of clusters.

Density-based

Density-based measures cannot simply use distances between points but also need to take the distribution of distances around given points into account. They are therefore typically much more complex to calculate and difficult to summarize.

CDbw uses *r representative points* from each cluster, instead of only centroids (i.e., $r = 1$). They are selected *farthest-first*, that is, the first point is the one farthest from the center, following ones farthest from the point selected before it. The closest representative points from two different clusters are then paired up and the sets of each pair of clusters' closest representative points intersected, similar to a shared nearest neighbor distance computation. The distance between any pair of closest representative points is normalized by the average σ of both clusters and weighted by the normalized number of points from both clusters whose distance to the midpoint between the two representative points is $<\sigma$. To derive the *intercluster density*, the sum over all clusters is calculated, with each cluster contributing the *maximal* value it has with any other cluster. This density is then used as a normalizing value for cluster separation. The numerator also sums over all clusters but each cluster contributes the *minimal* sum of distances of representative points. CDbw extends the idea of using the distance of representative points, as do methods using centroids, and exploits the standard deviation in a similarly manner as S_Dbw (which was proposed by the same authors). Compactness is also calculated based on representative points and σ, and different values of the measures are derived by varying r.

DBCV is influenced by the ideas underlying **DBScan**, that is, the notion of core points and reachability distances. To derive the criterion, a reachability graph is constructed, with distances as weights on the edges. This graph is then translated into minimum spanning trees within or between clusters, which are used to calculate the density *sparseness* (of individual clusters) and density *separation* (of pairs of clusters). The value lies between -1 and $+ 1$, with higher being better.

4.2.2 | Evaluating internal validation criteria

Milligan and Cooper (1985) used 108 small, artificial, low-dimensional data sets with clear cluster structure in Euclidean space to evaluate 30 internal criteria as to their ability to identify the correct number of clusters for hierarchical clustering approaches. Notably, there were at most five clusters in the data. They report the index developed by Calinski and Harabasz (CH) as performing best, closely followed by Duda and Hart. They note that techniques other than CH under-performed, in particular for data where only two clusters are present.

Bandyopadhyay and Maulik (2001) evaluate four criteria, the Davies-Bouldin index (DB), and three versions of Dunn's Index on three artificial, and two UCI (Dheeru & Karra Taniskidou, 2017) data sets, using genetic clustering algorithms, notably *not* including CH, as well as a new index they propose and refer to simply as $\mathcal{I}$. They show that DB generally performs well but that $\mathcal{I}$ improves upon it for the two data sets where it misses. The already mentioned work by Maulik and Bandyopadhyay (2002) evaluated DB, Dunn's, CH, and $\mathcal{I}$ on similar data as in their prior work. They report that only $\mathcal{I}$ identifies the correct number of clusters for each data set, and that two of the other ones fail on four of the data sets (CH fails on two).

Vendramin, Campello, and Hruschka (2009) point out that those prior comparisons looked only on the correct *number* of clusters, instead of assessing the correct *grouping* of data points. Using the artificial data from Milligan et al.'s original work, they use external criteria to asses the agreement of **k-Means** results with predefined partitions, which they use in turn to evaluate the CH, DB, Dunn's, Silhouette index and versions thereof. The authors report that for this particular collection of data sets, the original Silhouette index performs best, followed by CH, with simplified Silhouette indices close in performance. They also give a practical recommendation to limit the maximum number of clusters for an approach to $\sqrt{N}$ with N the size of the data set. We have grouped those studies into the relative performance graph seen in Figure 6.

Vendramin et al. (2010) followed up their earlier work by increasing the number of internal criteria to 40, generating more data sets (that remain relatively small and low-dimensional, however), and refining the evaluation methodology. Using **HAC** with four different merging criteria and **k-Means** with a number of different k, they generate a relatively large number of different partitions, and calculate the correlation between the score of internal criteria and ARI. They find that Silhouette and the

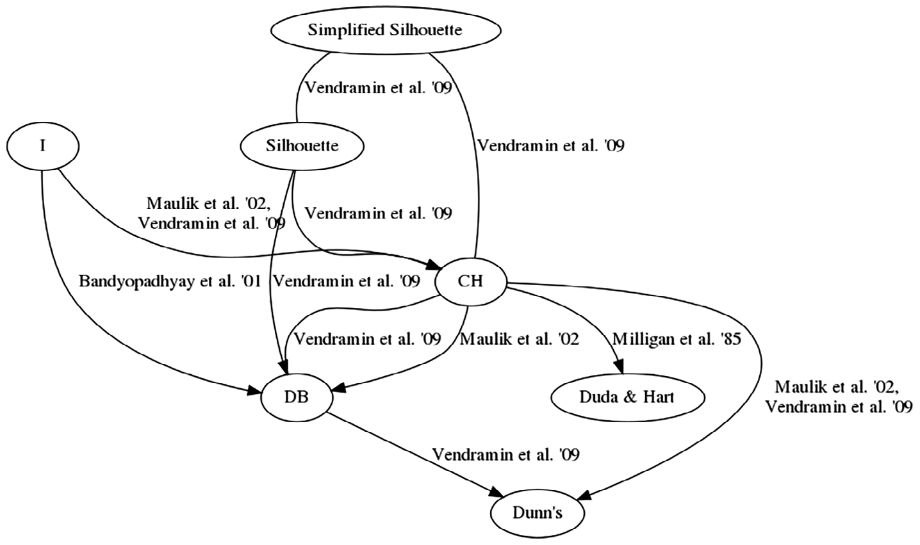

FIGURE 6 Relative performance graphs of several evaluations of internal validation criteria

newly evaluated Point-biserial perform best, followed by CH and the newly evaluated $\mathcal{I}$. They discourage using C-Index or Gamma in real-world situations when data might be noisy and the number of clusters high. They also report that optimization criteria perform better than criteria that compare successive partitions, such as Rubin's or Hartigan's indices.

Liu, Li, Xiong, Gao, and Wu (2010) generated five 2D artificial data sets, with five clusters each, to evaluate the effect of five data set and algorithmic characteristics on 11 internal criteria: the effect of increasing the value of the parameter controlling how many clusters to find, outliers or noise, different densities, subclusters, and skewed distribution of points. All data sets exhibited a rather clear cluster structure. Except for the last experiments, all data are clustered using k-Means, while for the last one they employed Chameleon. Their final conclusion lists the capability or not of different criteria to handle those aspects, which they recommend be used to help make the decision which criterion to choose. It is not clear, however, how one would use this information, which requires in-depth knowledge of the data that goes even beyond knowing the actual clusters. They also report that the S_Dbw validity index is the only one capable of handling all conditions.

Guerra, Robles, Bielza, and Larrañaga (2012) also added noise or outliers to artificial data. They evaluated five internal criteria, Silhouette, CH, C-Index, DB, Gamma, using artificial data with different yet relatively low dimensionality, number of clusters, outliers, and noise. As in the early work, they evaluated whether internal criteria identify the correct number of clusters. They clustered using k-Means, a HAC method, and a mixture of Gaussians fitted via EM. In general terms, they report that for data with outliers, CH clearly performs best for all three techniques, whereas noisy data is best handled by Gamma, followed by CH and Silhouette. Additionally, they show that Silhouette, DB, and Gamma deteriorate strongly in the presence of outliers, and that the C-Index actually does worse on well-separated data than on data with outliers. Interestingly, the authors also evaluated partitions using the Adjusted Rand Index (ARI) as external criterion (Morey & Agresti, 1984), and while they do not discuss this result further, they report that all algorithms were capable of finding *partitions that are very close to the ground truth* even if the number of clusters was off. We show the relative performance graphs of those two studies in Figure 7.

Another large-scale comparison has been performed in (Arbelaitz, Gurrutxaga, Muguerza, Pérez, & Perona, 2013), where 30 internal criteria have been evaluated. As Vendramin et al., they use agreement of results of k-Means and two HAC methods with different merging strategies with the ground truth partition via external criteria to rank internal criteria. As data, both artificial data and 20 data sets from the UCI repository were used. They report that on the synthetic data Silhouette performs best and is the only one to be correct at least half the time, a value much lower than in earlier work, followed by a version of DB and CH. Discussing characteristics of the artificial data, the authors report that overlap significantly deteriorates the performance of all criteria, and that the best-performing criteria are rather robust to noise. For real data, however, the three criteria drop in the rankings, and the top is taken by Score function, graph-theoretic versions of DB and Dunn's, and COP index. Regarding algorithms, finally, CH does best for k-Means and Silhouette for the two HAC approaches.

The work in (Chouikhi, Charrad, & Ghazzali, 2015) is interesting in that it uses a number of criteria not evaluated in other work, two of which, Hartigan's and Rubin's index, place higher in their evaluation than CH and DB. They evaluated those indices over a number of different HAC methods using different merging distances, on seven UCI data sets.

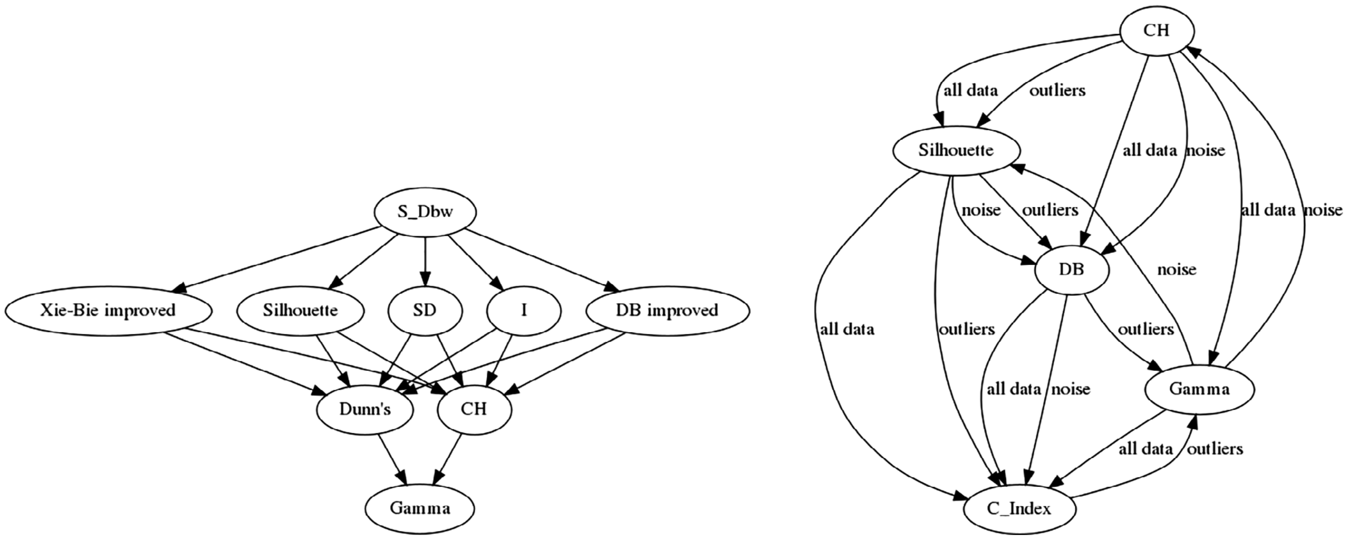

FIGURE 7 Relative performance graphs for the comparisons by Liu et al. (left) and Guerra et al. (right). In the left-hand graph, criteria at the same level have similar capabilities

Van Craenendonck and Blockeel (2015) considered four internal validation criteria, Silhouette, DB, CH, DBCV, which they use to identify the highest scoring solution for six clustering algorithms, **k-Means**, DBScan, Ward, EM, meanshift and spectral clustering. Notably, the authors also use those criteria to select the best parameter setting for each algorithm. The data consist of 27 UCI data sets. In their discussion, they point out that comparing to agreement with class labels via external criteria (such as ARI) can be misleading since classes might be arranged in a hierarchy. In addition, they discuss that highly imbalanced clusterings created by **DBScan** and meanshift clustering tend to score well for Silhouette and DBCV, and that explicitly removing a few points as noise gets rewarded disproportionately by CH and DBCV. In their conclusion, they state that **k-Means** (which forms elliptical clusters via centroids) is easily capable of achieving good scores for Silhouette and CH, and that **DBScan** does best for DB and DBCV, but state this to be true mainly due to undesirable properties. A final interesting insight is that they state DBCV to only be useful on data with clearly defined cluster structure.

Hämäläinen, Jauhiainen, and Kärkkäinen (2017), finally, recently evaluated seven internal criteria for "prototype-based" (or centroid-based) clustering methods, that is, **k-Means** and its relatives. Notably, neither Silhouette nor S_Dbw are among them. They used 56 artificial data sets well suited to **k-Means** type algorithms, that is, containing relatively well-separated elliptical clusters of differing dimensionality and hardness due to overlap, noise, density etc. They assessed whether internal criteria identified the correct number of clusters. They also used six UCI data sets but did not take class labels to represent ground truth and instead evaluated the stability of the suggested number of clusters. They report that no criterion consistently identifies the correct number of clusters on the artificial data, with Wemmert-Gançarski doing best, followed by KCE and the WB-index. An interesting additional result is that they evaluated different similarity measures, which are then also used to calculate the internal criteria, and find that those influence how well the correct number of clusters can be identified.

> Van Craenendonck and Blockeel (2017) also proposed a semi-supervised approach to selecting clustering methods and parameter settings: employing instance-level constraint, that is, must/cannot-link (points have to/must not be in the same cluster), they select the clustering solution violating the smallest amount of constraints.

We have spent so much time on these evaluations for several reasons. First, because studies on this question are by far the most prevalent in the clustering literature. Second, because knowing how to reliably use internal criteria could allow to circumvent guidelines for other parameter settings: instead, one could run an algorithm with a variety of different similarity measures, objective functions, and parameter values, and use a well-understood internal criterion to select the best result (as van Craenendonck et al. did). Third, because these studies illustrate the limitations of the used data: data sets are often relatively easy to cluster, not always including noise, outliers, or overlap. Fourth, because the choice of algorithms is interesting: almost all of them use **k-Means** or HAC, the latter often with the single-link merging criterion. This is remarkable because both techniques have a clear drawback: **k-Means** can only handle elliptical clusters and single-link tends to form "chains," that is, clusters where instances are close to their neighbors but rather far away from all other members of the cluster. At the same time, as we pointed out in the discussion of different validation criteria, almost of all them have been created to assess solutions created by those two types of clustering methods.

In fact, we have found only two studies comparing the use of internal criteria for density-based clustering, both in the context of introducing a new criterion. Halkidi and Vazirgiannis (2008) introduced CDbw, an internal criterion taking density into account, and report that it outperforms five other criteria, including S_Dbw and DB for **DBScan** and a version of **Chameleon**, according to ARI. Moulavi, Jaskowiak, Campello, Zimek, and Sander (2014) introduced a criterion they call Density Based Clustering Validation (DBCV) and compared it to Silhouette, CH, Dunn, $\mathcal{I}$ and CDbw on four artificial data sets exhibiting challenging geometry, three gene-expression data sets, and four UCI data sets. The clustering techniques used were **DBScan**, **OPTICS-Autocluster**, and H**DBScan**. They report DBCV to have overall best performance, by a large margin in the case of the artificial data. Such comparisons are to be taken with a grain of salt, though, since papers introducing new measures are unlikely to be published if the new measures do not improve on the state-of-the-art, and Van Craenendonck and Blockeel (2015) provide evidence against that claim.

4.2.3 | Initialization methods

There *are* studies that have evaluated other aspects of clustering technique parameterization, just fewer. Among those are studies regarding the initialization procedure for **k-Means**: Steinley and Brusco (2007) evaluated 12 initialization methods on artificial data. They report that an initialization approach recommended by Milligan in 1980 performs best in terms of cluster recovery according to ARI, and an approach due to Steinley (2003) in terms of finding an optimum of the objective function. Celebi, Kingravi, and Vela (2013) evaluated eight linear-complexity initialization methods on 32 UCI data sets as well as ~4,000 artificial data sets of differing clustering complexity. Notably, they *do not* use any of the internal criteria discussed above to decide the number of clusters but instead supplied the ground-truth. They report that in the average case, deterministic methods outperform nondeterministic ones, whereas nondeterministic ones have better best results. Among the nondeterministic methods, Bradley and Fayyad is reported to perform best, followed by greedy k-means++. For the deterministic ones, PCA-Part and Var-Part, two preclustering methods, perform best.

Kriegel et al. (2017) focused on running times instead of cluster quality and discuss, among other aspects, the different interpretations of the **k-Means** algorithm. They report that the **k-Means++** initialization procedure does not outperform random initialization on a data set created from histograms of image data. In addition, they found that the “standard algorithm” of Lloyd and Forgy repeatedly is slowest, which is problematic since this is both the algorithm typically understood to be “the” **k-Means** algorithm, and the version most often implemented in widely available toolkits.

4.2.4 | Similarity measures

Another question is that of which similarity measure to use, either in **k-Means** or HAC methods to group points, spectral clustering to populate the similarity matrix, or density-based approaches to define the proximity of points. Strehl, Ghosh, and Mooney (2000) looked at website clustering, treating websites as text documents, represented as bags of words. Using SOM, **k-Means**, weighted graph-partitioning, and hypergraph-partitioning, they evaluated Euclidean normalized similarity, Cosine, Pearson's and Extended Jaccard. They report Cosine and Extended Jaccard as doing best. Zhang, Huang, and Tan (2006) evaluated six distance measures for trajectory clustering in outdoor surveillance videos, using a spectral clustering method on data with ground-truth that they augment with lost points in the trajectory and Gaussian noise. They report that PCA + Euclidean distance performs as well as or even better than the much more expensive to calculate Hausdorff, HMM-based, and Longest Common Subsequence distances, as well as Dynamic Time Warping. Huang (2008) compared similar measures as Strehl et al. for text clustering, using **k-Means** on seven document data sets represented as TFIDF vectors. They report that the Euclidean distance does not perform well, which is in line with other results in the literature, yet also that the Cosine distance performs somewhat worse than Jaccard, Pearson's and the averaged Kullback–Leibler divergence. Recently, Shirkhorshidi, Aghabozorgi, and Wah (2015) cast a wider net, evaluating 12 similarity measures on 15 numerical data sets, selected from UCI and a repository of the Speech and Image Processing Unit, University of Eastern Finland. They used two prototype-based techniques, **k-Means** and **k-Medoids**, as well as HAC with single- and average-link. Their main intention was to evaluate the effects of dimensionality yet most of the data sets are low-dimensional, with only four having more than five dimensions, two of which with more than 20. Notably, the authors consider four and more dimensions high-dimensional. The quality of clusters was scored by treating classes as underlying clusters and using the Rand index. They report that the ranking of similarity measures changes depending on the algorithm and whether data sets have fewer than four dimensions.

A related question to that of similarity measures is whether one should standardize numerical data. Milligan and Cooper (1988) evaluated eight standardization approaches on artificial data, generated in a similar manner to the authors' other work and clustered using HAC with different merging criteria and Euclidean distance as the similarity measure. They report that different error conditions (which typically are not known beforehand) are better corrected by different standardization procedures, that standardizing by the interval [*min*, *max*] always ends up in the group of best-performing techniques, and that z-score standardization is *not* always helpful.

4.2.5 | Other parameters

Meilă and Heckerman (2001) explored a single clustering method, a multinomial mixture model, by changing the parameter estimation method, and comparing the resulting clusterings of artificial data and a handwritten digits data set. They report that EM performs best. Following this, they evaluated three initialization methods for EM parameters, a data-independent prior based on a Dirichlet distribution, a data-dependent method, and preclustering using agglomerative clustering. While they report the data-independent method to perform worse on the artificial data, this disappears for the real-life data set.

The work reported in (Aliguliyev, 2009) evaluated 12 objective functions for density-based clustering on document data. They used five data sets, three of which were also used by Zhao et al. The author focused on weighting instances' contribution to cluster descriptions, depending on how similar they are to the rest of the cluster. He reports that weighting documents improves over unweighted objective functions.

4.2.6 | Current day challenges: big data and stream clustering

The arguably most important characteristic of Big Data is the sheer volume and information about running times of different clustering techniques would be sufficient to decide which algorithm to use. There is, however, also the aspect of high dimensionality, and apart from Shirkhorshidi et al., we have not found anyone who evaluated that aspect, and as we indicated above, their concept of "high-dimensional" might not be in agreement with the way it is understood in a Big Data context.

With respect to stream mining, Hassani and Seidl (2017) evaluated the internal criteria that we have encountered in this section so far, using data generators contained in the MOA (Massive Online Analysis) toolbox. They used two algorithms, **CluStream** with **k-Means** as macroclusterer, and **DenStream**. Their evaluation criterion is the correct number of clusters for **CluStream** and external criteria for **DenStream**. They report that CH does well for CluStream, that $\mathcal{I}$ and S_Dbw are rather vulnerable to typical error conditions in streaming scenarios, and that S_Dbw does best for the density-based clusterer. Their results are in line with conclusions drawn for static data but are biased by the fact that they only included measures that had been evaluated as working well in the static context. In addition, they *did not* evaluate measures that have been proposed explicitly for a streaming context.

4.2.7 | Conclusion

To summarize this section, there is definitely a significant amount of work out there that can be drawn on to decide how to parameterize clustering techniques. The problem is, however, that the literature is piecemeal: connecting studies to each other to gain a comprehensive picture becomes difficult since different studies used different data, time and again evaluated the best-case scenario in terms of data characteristics, evaluated different methods, and certain methods are underrepresented or completely absent. The clearest result is arguably to be found with respect to which internal validation criterion to use to identify the most appropriate clustering.

5 | MATCHING RESULTS TO REALITY

To begin this section, we propose another thought experiment, one in which we actually know which algorithm to choose, and have identified how to parameterize it. After reasonable running time, we have a result that we are pretty confident is meaningful. At this point, two paths diverge (Figure 8): first, the unsupervised mining operation was only a first step in a longer data analysis pipeline, in which patterns are used to encode data for clustering, for instance, or the discovered clusters are used as classes for learning a predictive model. In that case, the usefulness of the result can be assessed by the success

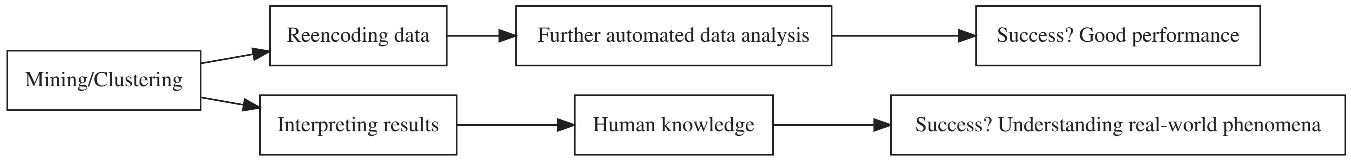
Mining/Clustering
Reencoding data
Further automated data analysis
Success? Good performance
Interpreting results
Human knowledge
Success? Understanding real-world phenomena

FIGURE 8 Data mining results can either act as inputs in further data analysis, or can be interpreted by humans to arrive at knowledge about the world

(or failure) of the larger operation. There is a second setting, however, in which data mining is embedded in a process of "knowledge discovery from databases" (Fayyad, Piatetsky-Shapiro, & Smyth, 1996). "Knowledge" implies human understanding and in the context of unsupervised mining, this means that humans should be able to interpret patterns or clusterings to understand the reality that gave rise to the data better. But to be able to do this, one needs to have an idea how the results of a data mining operation "typically" map to the generating processes underlying the data.

5.1 | Pattern mining

To a certain degree, this is a view that is supported by the pattern mining community, with the development of pattern set mining techniques, such as those we mentioned at the end of Section 3.1. One of the motivations often cited for pattern set mining is that humans cannot process a large number of patterns, and result sets should therefore be reduced to the 10 or 20 most interesting or relevant ones. Subjective interestingness (De Bie, 2011) and interactive mining (Goethals, Moens, & Vreeken, 2011; Van Leeuwen, 2014) take this even further: users are supposed to be able to evaluate patterns and give feedback to adjust quality measures or search space traversal. To be able to do this, however, users need to have some idea of how to map the patterns back to the data they came from.

Given that we define the patterns that algorithms mine, we know at a superficial level what is contained in mining results. In FIM, for instance, if there is set of items that is often bought together, it will be found. Yet so will all of its subsets, and maybe intersections with other itemsets, and it can be difficult to decide which of those are meaningful. Furthermore, FIM papers will often give the motivation behind performing such mining by stating that supermarkets can group items that are bought together close to each other, to motivate those customers who did not buy them together to do so. An alternative proposal states supermarkets should group such items far apart to motivate customers to traverse the entire store space and potentially make additional purchases.

These strategies implicitly assume two different types of customer behavior, though: in the latter case, the copurchases are systematic and can be leveraged to generate additional business. In the former, the copurchases are somewhat opportunistic, which also means that using the first strategy could lead to loss of business. Maybe the two types of behavior actually lead to different expressions of the pattern in the data. And maybe the layout of the supermarket at the time of purchase has an effect on this expression, for instance because it enables the opportunistic behavior. But to assess this, we need to be able to compare to the underlying process.

Studies comparing found patterns to patterns known to be in the data, or to a known generative process, will provide important information regarding what to make of mining results. We review such studies in this section.

A first type of evaluation takes a page out of the clustering playbook, trying to recover patterns that are known to be in the data, possibly distorted by noise. Laxman et al. (2005) elaborates a connection between Hidden Markov Models and episodes, and showed experimentally that their approach could indeed recover HMM-generated sequences embedded in noise. Tatti and Vreeken (2012a) generate artificial data without patterns, with 10 and 50 patterns embedded in noise and report that their proposed method finds no patterns in the first case, recovers all 10 in the second, and almost all patterns in the third. Zimmermann (2013) used the Quest generator and evaluated the capability of different condensed representations and association measures to recover itemsets. The evaluation showed that several association measures do not manage to filter out spurious itemsets, even for uncorrupted patterns, and that mining closed itemsets helps strongly in reducing the result set to an approximation of the embedded patterns, allowing the relatively simple *confidence* measure to perform very well. Webb and Vreeken (2014) follow a similar evaluation protocol, embedding 15 patterns and report that their method does not return *any* spurious itemsets but that it *does* return subsets of embedded itemsets, as well as subsets of those itemsets' unions. Lam et al. (2014) used artificial data with patterns generated by five independent parallel processes without noise to compare two pattern set mining techniques, SQS and GoKrimp, and report that while precision@10 is good for both of them, SQS has worse recall. The extensive evaluation in (Zimmermann, 2014) varied different parameters of a data generator embedding episodes into noise, such as alphabet size, maximal and distributions for temporal gaps, length and number of episodes etc., and evaluated different episode mining techniques' ability to recover the patterns. The main reported finding is that temporal constraints have a stronger influence on whether patterns are being recovered than pattern semantics. Kaytoue, Plantevit, Zimmermann, Bendimerad, and Robardet (2017) evaluated their approach by aggregating the random walks of simple agents described by itemsets into a multigraph, and report that their approach manages to recover approximations of the itemsets together with approximations of the walkers' trajectories in the graph.

A second type goes one step further and looks at whether generating *models* can be (partially) recovered. This is, for instance, the case in (Tatti & Vreeken, 2012b) where the authors generate a data set that is similar to a Mondrian painting and show that their approach can recover the (effectively noiseless) model. Mampaey, Vreeken, and Tatti (2012) generated three artificial data sets and argue that their approach recovers representations of the generating models. The model used in (Webb & Vreeken, 2014) is a Bayes Net and the proposed method is reported to recover itemsets corresponding to connected components in the net and identifying the strongest local positive correlations. The problem setting in (Prakash, Vreeken, & Faloutsos, 2014) consists of identifying patients zero for an infection from a snapshot of an infected graph. They report that their approach is capable of identifying the correct number and often the correct seed nodes in artificially generated graphs.

The third option consists of actually interpreting found patterns. In most cases, this will be difficult to do unless one has access to a domain expert but patterns from text data is accessible to everyone, which is why it has often been the data of choice. Tatti and Vreeken (2012a) used abstracts of JMLR papers and show examples of meaningful frequent term combinations that are not redundant. Those data were also used in (Lam et al., 2014) and they report that three pattern set mining techniques for sequential data, SQS, **SeqKrimp**, and **GoKrimp** all return similar collections of term combinations that refer to research topics and scientific concepts in the ML community. Mampaey et al. (2012) used data sets representing geographical animal and plant distributions and interpret the found patterns. They also used ICDM paper abstracts to compare their approach to three others, without interpreting the found patterns, however. NIPS abstracts have been used in (Webb & Vreeken, 2014) and the authors list the top-25 patterns together with short interpretations.

A particular pattern type can be found in (Y. Liu, Nie, Han, Zhang, & Rosenblum, 2015): the patterns of this work are more complex than episodes since the relationships between different activities are more complex than *before/after*. The goal of that work is activity recognition from sequences of atomic actions that can also overlap or performed in parallel. The data to which the approach is applied have been generated by human actors performing scripted or freely chosen actions to achieve certain activities. This is attractive in the sense that patterns can be compared to performed action sequences, and are interpretable by human users who will also have their own experiences in how to achieve different activities.

5.2 | Clustering

We have to deviate from surveying studies in this section since we have found no work to survey. This might feel surprising since clustering is about finding the groups of instances that are truly in the data at first glance, and as we have discussed before, there are quite a few studies that have evaluated whether different clustering techniques do that. Especially Milligan et al. also took care to introduce noise and outliers into those evaluations.

But when one digs a bit deeper, this does not exactly tell us something about the relationship between discovered and existing clusters. A Rand index of 90% tells us that 10% of all pairs of data points are not correctly grouped, that is, points grouped together that should be apart or vice versa. Knowing how to characterize those instances that are not being placed correctly, for example, as local outliers, would be very useful in interpreting clusters derived from real-life data and how to identify candidates for points that need to be corrected. But to the best of our knowledge, no such characterizations exists.

In our opinion, as in the case of pattern mining, it would be even more interesting to know how clusters relate to generating processes. If data has been generated by a mixture of multivariate Gaussians, for instance, a method that estimates the parameters of such a mixture, like the EM algorithm, will learn an approximation of it. How close such an approximation is under controlled conditions could be very informative when reasoning from a clustering model learned on real-life data but again, we cannot survey any studies on this subject.

A roundabout way of evaluating a learned clustering model could consist of predicting attribute values of held-out instances, or even sampling complete instances from clusters and comparing to what would be generated for the data itself. This is known as *imputation*, a clustering use case we have not mentioned so far (Li, Deogun, Spaulding, & Shuart, 2004;

S. Zhang, Zhang, Zhu, Qin, & Zhang, 2008). This has the advantage that no explicit cluster model is needed and that the results can be compared to those of generating processes that do not come in neat closed form. Again, doing this under noise and outlier conditions would allow to assess the reliability of cluster-based imputation but, again, we are not aware of work in this direction.

5.3 | Conclusion

Especially in recent years, pattern mining researchers have started to look into whether mined patterns agree with patterns hidden in the data, whether generative models can be recovered, or even whether (and how) patterns can be interpreted. Given that the ground truth is typically not available, the former type of evaluations has employed artificially generated data, limited to a few generator types. The latter type has used English-language text data since interpretations of patterns in such data are easily accessible to nonexperts. In clustering, we are not aware of studies that go beyond comparing found clusters to predefined groupings, which are considered the ground truth.

6 | BENCHMARKING IN CLUSTER ANALYSIS: A WHITE PAPER—COMPLEMENTARY GOALS AND FOCUS

We take a break from surveying at this point to put our work into perspective with respect to the white paper we have mentioned above (Van Mechelen et al., 2018). That work is concerned with benchmarking of clustering methods and could therefore be expected to have the same goals as our survey. Their focus is, however, rather on how to *design* comparisons, making sure that the same similarity measures, normalizations etc. are being used. The guidelines they lay out therefore partially depend on the insights derived from the existing comparisons that we discuss in this work.

van Mechelen et al. explicitly mention the issues that arise from using classes as ground-truth for clusters, and discuss the problem of external measures not rewarding methods that get close to the underlying cluster structure. In addition, they stress the effects of outlier removal, dimensionality reduction etc., be they positive (making the clustering task easier) or negative (giving the impression that certain methods perform better than they do). They also discuss topics that are out of the scope of this work, such as the difference between "crisp" versus. "fuzzy membership," that is, whether a point belongs only to a single cluster or can exhibit probabilistic membership.

Regarding data, finally, van Mechelen et al. discuss the pros and cons of different data sets in more detail than this survey does, laying out which data sets can be more useful for certain tasks.

To summarize, the white paper is much more focused on how to design good evaluation and comparison studies than this survey. The relationship between our work and theirs is almost symbiotic. Many of the topics that we discuss are meaningful in the context of what van Mechelen et al. aim for but while our work is focused on the *practical* aspects, the authors of the white paper cast a wider net, going beyond data sets and validation criteria to propose a general framework that would allow to decide which of them are meaningful. As such, it is therefore indispensable for anyone intending to fill the gaps in our knowledge since only well-designed evaluations will result in reliable insights.

7 | COMMUNITY DETECTION

After having surveyed the state of the art in pattern mining and clustering, we have to come to the conclusion that there are too few studies to inform us how to choose techniques, how to parameterize them, and how to exploit the results. Most of the surprising and contradictory information is just enough to question results reported in the literature but not enough to draw definite conclusions. As a contrast to this situation, we would like to shine a spotlight on a field where extensive evaluations and comparisons *are* much more available.

Graph clustering in networks is also known as *(social) community detection* (Fortunato, 2010) and the change of focus makes a large difference in the amount of available studies. The field is of interest to social scientists, life scientists such as biologists and medical researchers, physicists, and computer scientists in different subfields, as attested to by the different venues in which papers on the topic have been published. The survey cited in this paragraph includes a long discussion of comparative evaluations of community detection methods yet the author in his conclusion calls for more, and more in-depth evaluations. A reader interested in a detailed treatment of the field is therefore directed to Fortunato's work. Here, we indicate a number of particularly interesting results with respect to the three topics we have discussed so far, as well as a number of comparisons performed after 2010.[3]

7.1 | (Re)evaluations

Bagrow (2008) evaluated combinations of three local community detection methods and different stopping criteria on 10s of relatively simple artificial networks, and reports that LWP performs relatively well when communities are well-separated but deteriorates when there is overlap. Lancichinetti, Fortunato, and Radicchi (2008) introduced a new data generation model and evaluated two community detection methods, modularity optimization and Test of Potts, on that new data. They report that modularity optimization starts to fail once communities are better connected and once networks get larger, and benefits from vertices having large degrees. They also point out that modularity optimization is superior to the Potts test approach. Orman and Labatut (2009) also used artificial data generated according to the LFR framework (Lancichinetti et al., 2008) to compare five techniques. They report that SpinGlass and WalkTrap generally perform well, and that all algorithms improve when the average degree increases, as reported by Leskovec, Lang, and Mahoney (2010) report the results of comparing eight different algorithms, and 12 optimization functions on four real-world networks and claim in the paper to have performed the evaluation on more than 40 such networks. Since the actual communities for such networks are not always known, they evaluate algorithms according to an internal criterion, conductance. The number of techniques alone means that the results cannot be easily summarized but one of their conclusions is that Local Spectral, a spectral-based partitioning method, gives very good results while being computationally cheaper than other techniques. This is a simplification of the full discussion in their work, however, which stresses that changes in data characteristics can have rather strong effects on the performance of different techniques. Spam/ham e-mail identification was used to validate six community detection algorithms on a single network by Moradi, Olovsson, and Tsigas (2012). They report that there is no quantitative difference between algorithms. Harenberg et al. (2014) evaluated 13 methods, eight of which were published after 2010, on five real-life networks with known ground-truth. They report that TopGC performs best in terms of density and clustering coefficient while SLPA gives best conductance, and that SLPA recovers ground-truth communities best. Wang, Wang, Yu, and Zhang (2015) reimplemented and systematically evaluated 10 methods on seven real-life networks and ones generated using the LFR model. They propose a framework to break all algorithms down into eight components and analyze those separately. A remarkable characteristic of this study is that it mostly does *not* evaluate the methods used in the other studies in this section. They report that M-KMF is generally fastest, that SCP, M-KMF, MB-DSGE and gCluSkeleton do well on networks *with* outliers, whereas LPA and HANP give best results on networks without outliers, and that HANP, LPA and CNM find the structurally highest-quality communities. All in all, this is a very rich evaluation, hampered somewhat by the fact that it is hard to connect to the rest of the state-of-the-art. Recently, Yang, Algesheimer, and Tessone (2016) reported on the comparison of eight algorithms on more than a 100 networks artificially generated using the LFR model. They report that Infomap, Multilevel and Walktrap have relatively stable results for large network sizes, and that Label propagation has too large deviations to give reliable results. In addition, they point out that Fastgreedy consistently underestimates the number of communities in the data, and that for certain parameter values of the generator, Label propagation is incapable of recovering any community. Notably, they report that larger network size completely stymie Infomap and Label propagation. Wagenseller III and Wang (2017) compared five existing and a newly proposed algorithm on two real-world data sets, Twitter topology and DBLP. They focus on the assumption that *social* communities above a certain size (150 members) are not realistic and categorize communities based on their size. They report that modularity-maximizing techniques tend to produce much too large communities whereas their newly proposed method and Infomap concentrate their results in the ≤150 range. In addition, those two techniques place a (near-) majority of users in those communities of a "desired" size. They evaluated five other quality criteria and report that modularity maximization methods separate discovered communities better from the rest of the graph, and that their newly proposed techniques scores best in terms of triangle participation ratio. Dao, Bothorel, and Lenca (2017) evaluated 11 methods on seven real-world and five artificial data sets and show that depending on the internal evaluation criterion, different methods perform "best."

Community detection is therefore arguably the area in which relative algorithmic performance is best established, probably because the problem setting is of high interest in a number of different fields, for example, sociology, epidemiology, statistical physics, urban planning etc. The downside to this range of problem settings is that the comparison studies are also spread out over a number of different fields and their respective publications. In addition, as the preceding section (and the absence of a comparison graph) shows, even though there are quite a few comparisons, it remains very difficult to combine them into a full picture. In a sense, rigorous meta-studies, as prepared in the life sciences, might be needed to distill the available information into useful guidelines.

7.2 | Parameterization

Bagrow (2008) also tested three stopping criteria and reports that "trailing least squares" performs best overall but is rather dependent on the starting vertex in the graph. Lancichinetti and Fortunato (2009b) evaluated different parameter settings for the CPM method on artificial data created by a generator they introduce in that work, and report that certain parameter settings can lead to results deviating from the ground truth, without giving guidelines for how to set parameters. Leskovec et al. (2010) tested a number of objective functions that they group according to whether they focus on intracommunity or intercommunity links. While they report that the majority of objective functions behave more or less similarly, they also caution that optimizing them "aggressively" can lead to suboptimal results, such as barely connected communities. Using their framework, M. Wang et al. (2015) proposed improvements to the objective function of two algorithms, LPA and MB-DSGE, and show experimentally that the updated algorithms give better results. Yang and Leskovec (2015) analyzed 230 real-life networks using 13 structural quality measures. They show that those measures fall into four categories. They also define four functional criteria for "good" communities: separability, density, cohesiveness, and clustering coefficient, and evaluate how well the quality measures align with them. According to their analysis, Conductance and Triangle Participation Rate perform best with respect to the functional criteria overall, and Modularity has severe problems with it. Yang et al. (2016) discuss in detail how generative parameters can lead algorithms astray but point out that this information is not accessible to an end user. They discuss how certain algorithms (**Infomap**, Label propagation) are incapable of recovering actual data characteristics, whereas **Multilevel** is not too far off. Wageseller III and Wang (2017) also assessed the modularity measure under their assumption of ideal sizes for social communities, and argue that for social community detection settings, modularity maximization should be done while controlling for community sizes. They also evaluated parameter settings for their technique under this constraint, and recommend setting the growing threshold to 0.7, and the overlapping threshold to less than 0.6.

There seems to be a certain consensus that modularity is not a good criterion to optimize in community detection. We cannot help but notice, however, that far fewer works exist that give guidelines on which algorithm and which parameter settings to choose for new data, than on how algorithms compare on given data.

7.3 | Matching results to reality

There are data sets for community mining where communities are known, for instance in the form of Facebook groups, or based on affiliation information from publication networks such as that extracted from the DBLP. When such information is known, it has been used to evaluate the quality of found solutions but left the question of how to use communities to reason about the underlying processes aside. An exception to this can be found in (Lee & Cunningham, 2014) where a framework is proposed that uses ML techniques to infer missing values of node attributes in communities.

In addition, there have been a number of surprising and counter-intuitive results. Moradi et al. (2012) report that *structurally* strong communities, that is, well-connected ones, do not have semantic meaning.

Dao et al. (2017) point out that algorithmically discovered communities are *structurally better* than ground truth communities and argue that straight-up comparisons are therefore unfair to the discovery techniques, and caution that using discovered communities as stand-ins for real-life ones can be problematic.

Other work (Brandes et al., 2007; Lancichinetti & Fortunato, 2009a; Mishra et al., 2011; Orman & Labatut, 2009), have mainly focused on the relationship between properties of the model used to generate artificial networks and the found communities but since these models rarely employ the social mechanisms that give rise to communities, as we will argue below, it will be difficult to use the derived information to infer something about the real world underlying the data.

Hric, Darst, and Fortunato (2014) have also pointed out that there is a clear difference between structural communities and ground-truth communities as currently defined in the literature. Peel, Larremore, and Clauset (2017) argue that the current approach to assigning ground-truth communities in real-life networks, based on meta-data such as membership to Youtube groups, is faulty, explaining the disconnect between structural criteria and ground-truth communities mentioned above. They propose two methods, one for assessing the strength of the correlation between meta-data and ground-truth communities, the second to assess whether the two reveal different aspects of a network's structure. A combination of those methods with the results of community detection techniques on data with known meta-data and ground-truth communities could provide a guideline for identifying actual communities.

7.4 | Conclusion

As this section shows, the state of comparative evaluations, guidelines for parameterization, and discussions of how algorithmic results relate to the reality in the data is much richer for community detection than for pattern mining and clustering. Fortunato's criticism notwithstanding, a practitioner is therefore much more likely to understand how to apply a community detection technique. We suspect that the main reason for this is that this concrete data analysis task is of strong interest to researchers from a number of different disciplines.

8 | THE DATA THAT WE HAVE AND THE DATA THAT WE NEED

As the preceding sections show, there is a certain imbalance between developed techniques and the systematic evaluations comparing them or assessing parameter settings and the relevance of results. Many of the found solutions are ingenious and elegant, and the impressive tool kit that has been amassed should enable practitioners to address many real-world problems more effectively. But faced with data, a typical practitioner will not know *which* tools to choose, *how* to set the parameters without extensive trial-and-error, and *what* conclusion to draw from the resulting patterns or models—unless we fill in the gaps. So what will be needed is a new type of research program, one that fills them in.

There are, however, still a few obstacles in the way of such a program. Some of those are related to what kind of research is rewarded with publication, for example, positive results and proposing new techniques, which in turn relates to hiring and promotion policies. To what degree those obstacles actually prevent doing and publishing comparative studies is somewhat subjective, and overcoming them will require structural changes. In purely practical terms, reevaluating and comparing is becoming easier, with more and more researchers sharing their implementations and prepared data. This is also strongly encouraged (or even made mandatory) by a number of journals and conferences.

Discussing those questions is out of the scope of this survey. Instead, we want to draw attention to an objective factor that reared its head in each of the preceding sections. Many times when discussing evaluations, we have pointed out that data used for pattern mining was only of a sparse nature, or that data for clustering contained easily identifiable cluster or was noiseless.

Somewhat ironically, given the name "Data Mining," the data that we have is not the data that we need!

This sentence is a bit simplified because there are in fact two issues at play here: first, curated data repositories are limited in what they offer. This is a problem that could be resolved with some work since there are far more data sets in the public domain than are typically being used. Second, however, to systematically evaluate methods, we need informative descriptions of the characteristics of those data, and collections of data for which different data sets differ in only one particular characteristic. Regarding the first point, we survey the state of the available data here.

In reaction to the work of Zheng et al., the data sets they introduced were added to the benchmark data sets for FIM and reliance on the data generator from (Agrawal & Srikant, 1994) was reduced. Several other data sets were added over time and the collection is currently downloadable at the FIMI website.[4] Yet the totality of this collection comprises only 11 data sets. The link to the twelfth data set, Gazelle from KDD Cup 2000, is nonfunctional at the time of writing. That data set is still available on the web, though.[5]

A repository of data for sequence mining has been assembled by Fournier-Viger,[6] it contains eight real-life data sets, one of which is Gazelle and a second one that is also provided at the FIMI website, and links to three data generators, one of which is the sequential version of QUEST. An alternative repository has been established by the TREC conference series,[7] containing several 10s of text document data sets. Most of the real-life data used in episode mining papers are covered by non-disclosure agreements and have therefore never entered the public domain. Both Zaki's original tree data generator and the CS-Logs data can be found at his website.[8] The different data sets used in graph mining papers are, to the best of our knowledge, not available in a single curated repository. This does not mean that those data sets are not available anymore—some can be found by searching on the web, or by contacting the original authors. But when time is of the essence, and we have mentioned two studies where time constraints precluded the acquisition of a comparison implementation, a researcher might not be willing or able to go to the effort.

This is also the caveat that has to be kept in mind when thinking about how to address this first problem: the *open data* movement has made more and more data sources accessible, whether governmental, other public, or private, and Kaggle, for instance, hosted 13,806 data sets at the time of writing.[9] But unless a (group of) researcher(s) curates a standard-setting benchmark and offers an easily accessible gateway, for example, by registering it with Google's Data set search engine,[10] the simple existence of more data will not solve the problem. If data providers and publishers take care to publish their data as *linked data* sources, and the corresponding identifiers are included in published work, this should make accessing and reusing them much easier. This path is not a panacea, however: as we mentioned in the preceding paragraph, links can go stale, and there are a number of other problems that can crop up with linked open data (Rula, Maurino, & Batini, 2016) that can undermine reuse of data sets.

Generally speaking, the situation is far better for clustering. As we have mentioned before, all data can be considered clustering data and the UCI repository for machine learning (Dheeru & Karra Taniskidou, 2017) offers 463 data sets,[11] the freshly updated UCR collection of data for time series classification and clustering (Dau et al., 2018) 128,[12] and the Mulan repository for multilabel learning 44.[13] There are also curated repositories dedicated purely to clustering, offering differing numbers of data sets.[14] How to choose from all these data is not obvious but researchers from the University of Eastern Finland have proposed what they refer to as a "basic" clustering benchmark set consisting of 52 data sets.[15] In the publication where they introduced this collection, they caution to use those data *only for methods claimed to be capable of finding elliptical clusters*, such as k-Means. Notably, there is significant overlap between the different curated repositories (and the data hosted by Kaggle) and/or the UCI repository. Similarly to the FIMI workshop, the DIMACS Implementation Challenge—Graph Partitioning and Graph Clustering (Bader et al., 2012) has assembled a repository of network data,[16] containing a mix of real-life and artificially generated data.

A large collection of data sets for community detection has been assembled at Stanford (Leskovec & Krevl, 2014),[17] eight of which contain ground-truth communities. It shows some overlap with the DIMACS repository. Analyzing five of those data sets, Harenberg et al. (2014) found that even among social networks, ground-truth communities do not share the same structural characteristics, and that especially Youtube communities do not have the structural properties of "good" communities in the theoretical sense. There is also the KONECT repository at the university of Koblenz,[18] which provides 261 networks from different domains that are annotated with structural information such as number of triangles or degree distribution.

However, while a large number of available data sets is a *necessary* condition for systematically evaluating unsupervised, it is not a *sufficient* one. First off, even a large number of data sets might have more or less the same characteristics, and they might not be representative of the real-world at large. Certain data are easier to collect, or more likely to be put into the public domain. Second, systematic evaluations ideally vary a single characteristic to evaluate the effects of those changes on the method at hand. If changing the dimensionality means changing the size of the data set at the same time, or the distribution of attribute values in the data, it is hard to tease apart which changes in the method's behavior are due to which change. Vendramin et al. (2009) even go one step further, stressing that a single result on a data set is not very reliable and evaluations should be based on a series of sets of roughly similar data sets, for example, same dimensionality, size, and generative process. In much existing work, researchers have attempted to address this problem by oversampling or under-sampling instances, in which case certain phenomena will in all likelihood be duplicated or lost. Changing the dimensionality in that way would be even harder: randomly deleting or adding attributes can introduce or remove redundancies and therefore change behavior in unexpected ways. Third, there are certain characteristics that are difficult to assess or control on real-life data such as noise or outliers. Given that one typically does not know whether there is already noise in the data, removing it is not possible and it is unclear how much noisier adding noise will make it. And for outliers, one would need to know the ground-truth, which brings us to the final point. Many data sets do not contain any ground-truth that results can be compared against to address the problem discussed in Section 5. While we find this to be obvious for pattern mining, it is also often the case for clustering. Existing work, when they did not use artificially generated data, refers to class labels but classes are often different from clusters, that is, groups of similar data instances, as Färber et al. (2010) discuss in detail.

9 | DATA GENERATION IN UNSUPERVISED DATA MINING

Luckily for computer scientists, there is an alternative to assembling ever increasing collections of real-life data, or rather a complement: artificial data generation. As we have seen in preceding sections, many of the most interesting results have been derived on artificial data where characteristics such as the dimensionality of the data, noise, overlap between clusters, or fan-out and size of trees in the data could be finely controlled. This is also the solution that has been chosen in exploring phenomena in the SAT solving community (Pennock & Stout, 1996), for instance.

9.1 | Pattern mining

The problem is, however, that the current state of data generation in Pattern Mining does not yet achieve satisfying results. The data generator used by Agrawal and Srikant (1994) was discredited by Zheng et al. This has repercussions since the data generators used in sequence and graph (and arguably tree) mining papers are based on similar considerations. Cooper and Zito

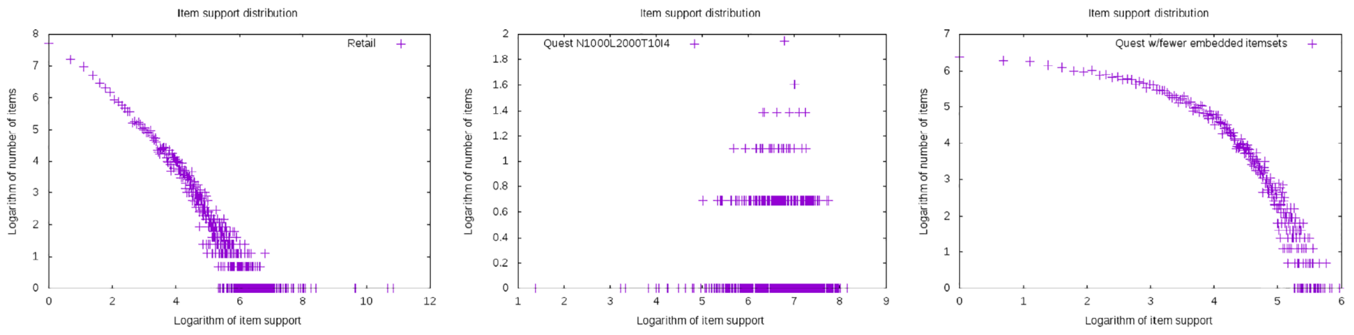

FIGURE 9 Item support distribution for the retail data set (left), Quest N1000L2000T10I4 (center), and Quest with L80 (right)

(2007), remarked that the item-support distribution in the real-life "retail" data set (Figure 9, left-hand side) showed a significantly different shape from that of one of the standard QUEST data sets (Figure 9, middle), and proposed a preferential-attachment based generator to address this phenomenon. In doing so, they kept the Poisson distribution for transaction sizes that had been identified as unrealistic by Zheng et al. In addition, as Figure 9, right-hand side, shows, lowering the number of embedded itemsets from 2000 to 80 very much changes the distribution's shape, even if it does not exactly match "retail." We mention this example as an illustration that the generating process of the generator alone is not enough to determine data set characteristics but that different parameter settings can significantly affect how generated data looks.

But also for data generators that have not been explicitly criticized in terms of statistical differences to real-life data, experimental comparisons give different results from real-life data, as in the cases of generated sequential versus planning data, and generated tree versus the CS-Log data.

Generators leading to data resembling the one already available can help supporting the proposal of Vendramin et al. yet suffer from the fact that they do not solve the problem of the data bottle neck. Approaches such as (Ramesh et al., 2003; Ramesh, Zaki, & Maniatty, 2005; Vreeken, van Leeuwen, & Siebes, 2007) take the output of an FIM operation and generate databases that will result in similar output. Thus, they take the existence of data as a given, as do the approaches that create null models based on the data. Furthermore, the data generated by the former is expected to result in the same mix of relevant and irrelevant patterns as the old one. Those generators *could* aid in solving the problem of scaling up existing data, however, and in creating collections of data sets with roughly similar characteristics. The latter type, that is, null models, masks or even seeks to break the underlying processes, however, generating data that while with respect to certain characteristics is very *similar* to existing data, is very *different* in terms of the results that will be produced.

An interesting opportunity, related to the work on activity detection discussed in Section 5.1, is provided by Liu, Cheng, Liu, Jia, and Rosenblum (2016). They learn a generative model from existing activity data that combines the type of complex patterns found in (Liu et al., 2015) with a probabilistic model. The result is a generator that can be used to generate *human-like* activity data yet, as in cases mentioned above, it is limited by the data from which the model has been learned.

9.2 | Clustering

In attribute-value clustering, the work by Milligan (1985) has been very influential early on, as we have discussed. Slightly more than a decade ago, Pei and Zaïane (2006) have proposed a generator allowing to embed various cluster structures of differing difficulty in data, limiting the data to only two dimensions, however. Qiu and Joe (2006) proposed an improvement on the work of Milligan et al. allowing for better separation of clusters and different cluster shapes, similar to what Pei et al. propose. Steinley and Henson (2005) focus most on controlling the overlap between clusters. There exist more recent work (Frasch, Lodwich, Shafait, & Breuel, 2011; Sadikin, 2014) that have not yet experienced much uptake, so that the jury is still out on their quality. While Dries (2015) situates his proposal in the context of supervised learning, it is based on first-order logic and he claims the capability to generate also structured data, that is, sequence, tree, graph data. Melnykov, Chen, and Maitra (2012) offer an tool for creating (Gaussian and non-Gaussian) mixtures.

9.3 | Community detection

The survey undertaken in (Chakrabarti & Faloutsos, 2006) lists 22 generators for network data, all of which attempt to reproduce certain statistical properties of real-life data, such as degree distribution, or clustering coefficient. With the exception of a few that attempt to model particular types of networks, the authors found that none of the proposals gets it fully right. The work of Lancichinetti et al. (2008), mentioned therein, was itself motivated by the insight that the artificial graphs proposed by Girvan and Newman (2002) lack certain realistic characteristics. Lancichinetti and Fortunato (2009b) proposed a generator for directed and weighted networks. In his survey of community detection, Fortunato (2010) also discusses graph generators and points out that nearly all benchmark networks existing at that moment were motivated by the same underlying model and criticizes them as not realistic.

More recent works have since explored the statistical properties of communities in real-life networks (Leskovec, Lang, Dasgupta, & Mahoney, 2008), or the effect of the realism of generated graphs on algorithm evaluations (Orman, Labatut, & Cherifi, 2011). Delling, Gaertler, Görke, Nikoloski, and Wagner (2006) explore the issue in detail, identifying relationships between data generation, data characteristics, and algorithmic performance, and caution against naïve evaluations. Coming from a very different direction, collaborative learning of agents, Veillon, Bourgne, and Soldano (2017) showed how some widely-accepted network characteristics do not capture vital aspects of how information flows in networks.

The arguably most interesting result comes from Dao et al. (2017) who show that for several internal evaluation measures, algorithms achieve *better* results than the ground-truth communities in the data, be they real or artificial. This can be considered an illustration of the clusters-to-classes problem mentioned above.

Recently, Dao, Bothorel, and Lenca (2018) used eight different community detection methods on 108 real-world networks grouped into six categories: "Biological," "Communication," "Information," "Social," "Technological," and "Miscellaneous." Using two quantitative criteria of found communities—*type transitivity* and *hub dominance*—they illustrate that different categories have different profiles in the 2D space spanned by those criteria. In addition, they show where the networks resulting from different graph generators fall, and why LFR networks are not representative of real-world data.

9.4 | Conclusion

Artificial data generation enables us to create data with a wide range of characteristics, to assess the effects of different kinds of noise on the ability to recover patterns, and to simulate different generative processes. As we have discussed in the survey so far, many of the most interesting and most counter-intuitive results have been derived from experiments on artificial data. Existing work on data generation has a number of shortcomings however, making them not fully exploitable for meaningful systematic evaluation. Work that uses artificial data to carefully tease out what changes in the data characteristics means for algorithm performance always runs the risk to have its results challenged as nonrepresentative of real-world data. It is therefore necessary that artificial data comes with guarantees with respect to the relationship to real data, and we are convinced that to get there, we, or at least some of us, have to become *data scientists*.

10 | DATA SCIENCE—A PROPOSAL FOR A FUTURE RESEARCH DIRECTION

We conclude this review in the same manner as we have begun it, by likening our branch of computer science to an engineering science. In engineering, developing a single component is not enough, whether it is a gear train in mechanical engineering, a synthesizing process in chemical engineering, a sorting algorithm in software engineering, or a data mining method in our field.

When media and nonacademics refer to data miners or data analysts, the term "data scientist" is often used. But what this term, and the explanations around it, imply is that such a person understands the data, that they can make the three informed decision we have mentioned in the introduction: which technique to choose, how to parameterize it, and how to interpret the results to derive knowledge for the nondata miner experts. As this survey has shown, while there are interesting and important studies about those aspects, there are too few and they are too loosely connected to enable most users to do this, data miner or not.

Part of this is arguably by design—as we wrote in the beginning, the promise of unsupervised data mining is to find interesting structures in a largely unsupervised manner and an early dream consisted of doing this entirely automatically, without the need of human intervention at all. Incidentally, this is a dream that has been revived in the field of supervised mining and learning, most prominently in the form of AutoML.[19] AutoML is based precisely on the kind of research that is *lacking* for unsupervised mining, using the quality of mining results to tune parameter settings. The naïve interpretation of this promise, furthermore, is similarly flawed as the claim that in the age of "Big Data," discovering correlation replaces understanding causation (Anderson, 2008).[20]

To properly form data scientists, we should therefore fill in the three knowledge gaps. To this end, we need to systematically (re)evaluate existing techniques, develop guidelines for how to parameterize methods, and knowledge about how to map unsupervised mining results to the generating processes that created the data. Our best chance to do this is using artificial data generators that allow us to control and manipulate data characteristics, as well as knowing the underlying phenomena of the data. Such research would not need to start from scratch but could build on work that has been done in other fields, some of which we will discuss in the final paragraphs.

10.1 | Artificial data generation in other research domains

There are already a number of fascinating proposals for generating data, formulated by experts in the respective fields, of which we will describe a selection. Downs and Vogel (1993) described how to encode numerous chemical reactions in Fortran to model the behavior of a chemical plant in detail under the influence of certain process parameters. Faced with the fact that data from high-energy cosmic ray particles striking the atmosphere are both rare and noisy, Heck, Schatz, Knapp, Thouw, and Capdevielle (1998) proposed Monte Carlo simulations to help with the evaluation of methods for identifying such phenomena in real data. Social sciences, in particular, have developed a rich tradition of generating artificial data for the simple reason that large-scale real-life experiments are impossible, unethical or both. Wu et al. (2018), when proposing to use existing macrolevel population data to generate artificial populations, cite a number of works from computational sociology that follow a similar approach (Barrett et al., 2009; Bisset et al., 2006; Ma & Srinivasan, 2015; Müller & Axhausen, 2011; Namazi-Rad, Mokhtarian, & Perez, 2014). Unfortunately, they seem not to have been aware of the influence that agent-based interactions have on those data. Ever since Macy and Willer (2002), computational sociology has realized that interactions among relatively simple agents can give rise to complex phenomena that one cannot create based purely on distributional characteristics. Sociologists have focused on the big problems but something as small as a client shopping, or an agent with certain interests step-by-step building connections with other agents with similar interests could be simulating precisely the data that unsupervised data mining techniques explore. Given that there are definite behavior vectors for agents, one can compare data mining results to what the entities that generated the data "intended." In a very recent book summarizing his research on cultural evolution, Laland (2018) repeatedly refers to in silico simulations supporting or rejecting theories of the evolution of teaching strategies, for instance, since experiments on evolutionary time scales are impossible.

In short, for a number of different problem settings for which it is very time-consuming or impossible to acquire real-life data, noncomputer science researchers have been much more willing to resort to artificial data generation than data miners seem to have been.

Such experts can therefore propose generating processes and parameter settings to generate realistic data. In addition, there are also researchers in computer science that have made an effort to understand how to generate realistic data (Hall & Posner, 2010), or at least to characterize existing data for use in future generators (Lancichinetti, Kivelä, Saramäki, & Fortunato, 2010; Orman, Labatut, & Cherifi, 2013).

In a similar manner, systematic explorations of already existing data generators should explore how different distributions and parameter settings interact. As we have illustrated anecdotally at the beginning of Section 9, changing a single parameter setting of the **Quest** data generator changes data characteristics in an unexpected and rather significant way.

10.2 | Data adaptation techniques

In supervised learning, a fundamental assumption is that training data and the unseen data on which predictions will be performed exhibit largely the same characteristics. This means that class label distributions should be roughly similar, as should attribute value distributions etc. If this is not the case, it is necessary to perform *domain adaption* (Jiang, 2008) to align the different data sets. A related but slightly different problem setting is one where not enough labeled data are available in the domain of interest. One can then build a model on plentiful data and use *transfer learning* (Pan & Yang, 2009) to apply the model to the desired domain. Domain adaptation and transfer learning techniques exist in unsupervised (Fernando, Habrard, Sebban, & Tuytelaars, 2013), semi-supervised (Donahue, Hoffman, Rodner, Saenko, & Darrell, 2013), and supervised form (Chen, Weinberger, & Blitzer, 2011), and future data scientists could build on those techniques to derive new data from existing real-life data sets.

10.3 | Existing tools and infrastructure

In addition to the knowledge already existing in other fields and in the computer science literature, tools also exist that can make building up the necessary knowledge easier. A 2010 proposal for a modular data generator in the Knime toolbox (Adä & Berthold, 2010) allows the relatively low-cost testing of at least itemset and attribute-value data generation. The aforementioned experiment databases, for instance in their implementation at https://www.openml.org/, could be extended for unsupervised mining (they are currently limited to supervised tasks). This database already includes a number of data set characteristics developed by the Meta-Learning community, which would of course also need to be extended to include more meaningful information with respect to unsupervised data, including those already reported in the literature (cf. Sections 3 and 4). With respect to clustering, the concept of clusterability has been introduced and discussed (Ackerman, Adolfsson, & Brownstein, 2016; Ackerman & Ben-David, 2009), which assesses how easy (or not) it is to cluster a data set, a relevant consideration when evaluating clustering techniques. Experiment databases, as other crowd-sourced projects, are reliant on what contributors upload, which means in particular that data is generated to specifications and properly described, but this can be leveraged since discrepancies, either on different data sets or on a supposedly identical one, can provide first hints at something that should be investigated in more detail.

10.4 | Conclusion

We hope to have convinced the reader of this survey that undertaking principled reevaluations and comparisons of unsupervised mining techniques, systematic studies on how to parameterize techniques for given data, and explorations of how derived results can be interpreted to draw inferences about the real world is a worthwhile endeavor.

The existing literature on pattern mining and clustering already contains surprising results that are often not reflected in the general use and interpretation of mining techniques, challenging or extending existing results. Many of those results have been derived on artificial data with controlled characteristics, yet most data generators do not generate realistic data (or at least they cannot be guaranteed to do so). As we discussed in the preceding paragraphs, there is much work available on which to build to create realistic data generators.

Concretely, researchers in other fields have developed sophisticated data generators rooted in their understanding of their research matter. Having such generators offer three attractive features: (1) realism, (2) the possibility to vary only certain characteristics, and (3) a ground truth in the form of generating processes. Researchers in machine learning have developed approaches to transform models so that they can be applied to data from which they have not been derived. Tools to both prototype generators and store results in an accessible manner are already (almost) available. As such, working towards a body of knowledge that contains much fewer gaps (or better yet, none at all) is clearly feasible.

ACKNOWLEDGMENTS

We are grateful to Matthijs van Leeuwen, Arno Siebes, and Jilles Vreeken for reviewing preliminary versions of this article, Marc Plantevit, Wouter Duivesteijn, and Arnaud Soulet for providing the same service for the first revision, as well as the anonymous reviewers. They all gave valuable feedback, sharpened the questions, and discussed possible answers.

CONFLICT OF INTEREST

The author has declared no conflicts of interest for this article.

ENDNOTES

[1] http://www.realkd.org/subgroup-discovery/the-power-of-saying-i-dont-know-an-introduction-to-subgroup-discovery-and-local-modeling/.

[2] There were no further iterations of the FIMI workshop since: "LCM, implemented in C, is simply faster than anything else." (Bart Goethals, personal communication).

[3] The year in which Fortunato's survey was published.

[4] http://fimi.ua.ac.be/data/, accessed 20/10/2018.

[5] http://www.kdd.org/kdd-cup/view/kdd-cup-2000, accessed 01/01/2019.

[6] http://www.philippe-fournier-viger.com/spmf/index.php?link=datasets.php, accessed 19/12/2018.

[7] https://trec.nist.gov/data.html, accessed 17/07/2018.

[8] http://www.cs.rpi.edu/ zaki/www-new/pmwiki.php/Software/Software#toc41, accessed 21/10/2018.

[9] https://www.kaggle.com/datasets, accessed 01/01/2019.

[10] https://toolbox.google.com/datasetsearch.

[11] Accessed 01/01/2019.

[12] Accessed 01/01/2019.

[13] http://mulan.sourceforge.net/datasets.html, accessed 01/01/2019.

[14] https://www.uni-marburg.de/fb12/arbeitsgruppen/datenbionik/data?language_sync=1, https://ifcs.boku.ac.at/repository/datasets.html, https://github.com/deric/clustering-benchmark, http://www.gagolewski.com/resources/data/clustering/, https://data.world/datasets/clustering, all accessed 21/10/2018.

[15] http://cs.joensuu.fi/sipu/datasets/, accessed 21/10/2018.

[16] https://www.cc.gatech.edu/dimacs10/downloads.shtml, accessed 01/01/2019.

[17] https://snap.stanford.edu/data/, accessed 01/01/2019.

[18] http://konect.uni-koblenz.de/networks/, accessed 25/06/2019.

[19] https://cloud.google.com/automl/.

[20] A claim that has experienced tremendous push-back.

RELATED WIREs ARTICLES

Filtered-top-k association discovery

ORCID

Albrecht Zimmermann https://orcid.org/0000-0002-8319-7456

FURTHER READING

Han, J., Wah, B. W., Raghavan, V., Wu, X., & Rastogi, R. (Eds.). (2005). *Fifth ieee international conference on data mining*. Houston, TX: IEEE.

Pei, J., Han, J., & Mao, R. (2000). Closet: An efficient algorithm for mining frequent closed itemsets. In *ACM SIGMOD workshop on research issues in data mining and knowledge discovery* (pp. 21–30).

Zaïane, O. R., Goebel, R., Hand, D., Keim, D., & Ng, R. (2002). Knowledge discovery and data mining. In *Proceedings of the eighth ACM SIGKDD international conference on knowledge discovery and data mining, July 23–26, 2002, Edmonton, Alberta, Canada*. New York, NY: ACM.

REFERENCES

Ackerman, M., Adolfsson, A., & Brownstein, N. (2016). An effective and efficient approach for clusterability evaluation. *arXiv preprint arXiv: 1602.06687*.

Ackerman, M., & Ben-David, S. (2009). Clusterability: A theoretical study. In *Artificial intelligence and statistics* (pp. 1–8). JMLR.org.

Adä, I., & Berthold, M. R. (2010). The new iris data: modular data generators. In B. Rao, B. Krishnapuram, A. Tomkins, & Q. Yang (Eds.), *KDD* (pp. 413–422). New York: ACM.

Agrawal, R., & Srikant, R. (1994). Fast algorithms for mining association rules in large databases. In *Proceedings of the 20th international conference on very large databases* (pp. 487–499). Santiago de Chile, Chile: Morgan Kaufmann.

Agrawal, R., & Srikant, R. (1995). Mining sequential patterns. In P. S. Yu & A. L. P. Chen (Eds.), *ICDE* (pp. 3–14). Washington, DC: IEEE Computer Society.

Aliguliyev, R. M. (2009). Performance evaluation of density-based clustering methods. *Information Sciences*, *179*(20), 3583–3602.

Anderson, C. (2008). *The end of theory: The data deluge makes the scientific method obsolete*. Retrieved from http://archive.wired.com/science/discoveries/magazine/16-07/pb_theory.

Arbelaitz, O., Gurrutxaga, I., Muguerza, J., Pérez, J. M., & Perona, I. (2013). An extensive comparative study of cluster validity indices. *Pattern Recognition*, *46*(1), 243–256.

Asai, T., Abe, K., Kawasoe, S., Sakamoto, H., Arimura, H., & Arikawa, S. (2004). Efficient substructure discovery from large semi-structured data. *IEICE Transactions on Information and Systems*, *87*(12), 2754–2763.

Atallah, M. J., Gwadera, R., & Szpankowski, W. (2004). Detection of significant sets of episodes in event sequences. In *ICDM* (pp. 3–10). Washington, DC: IEEE Computer Society.

Ayres, J., Flannick, J., Gehrke, J., & Yiu, T. (2002). Sequential pattern mining using a bitmap representation. In *Proceedings of the eighth ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 429–435). New York: ACM.

Bader, D., Meyerhenke, H., Sanders, P., Schulz, C., Schumm, A., & Wagner, D. (2012). A benchmarking set for graph clustering and partitioning. In *Encyclopedia of Social Network Analysis and Mining*.

Bagrow, J. P. (2008). Evaluating local community methods in networks. *Journal of Statistical Mechanics: Theory and Experiment*, *2008*(05), P05001.

Bandyopadhyay, S., & Maulik, U. (2001). Nonparametric genetic clustering: Comparison of validity indices. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, *31*(1), 120–125.

Barrett, C. L., Beckman, R. J., Khan, M., Anil Kumar, V., Marathe, M. V., Stretz, P. E., … Lewis, B. (2009). Generation and analysis of large synthetic social contact networks. In *Winter simulation conference* (pp. 1003–1014). IEEE.

Bayardo Jr., R. J., Goethals, B., & Zaki, M. J. (Eds.). (2004). *FIMI '04, proceedings of the IEEE ICDM workshop on frequent itemset mining implementations, Brighton, UK, November 1, 2004*.

Besson, J., Rigotti, C., Mitasiunaite, I., & Boulicaut, J.-F. (2008). Parameter tuning for differential mining of string patterns. In *ICDM workshops* (pp. 77–86). IEEE Computer Society.

Bisset, K., Atkins, K., Barrett, C. L., Beckman, R., Eubank, S., Marathe, A., … Kumar, V. (2006). *Synthetic data products for societal infrastructures and proto-populations: Data set 1.0* (Tech. Rep.). Tech. Rep. TR-06-006, Network Dynamics and Simulation Science Laboratory, Virginia Tech, Blacksburg, VA.

Boley, M., Gärtner, T., & Grosskreutz, H. (2010). Formal concept sampling for counting and threshold-free local pattern mining. In *Sdm* (pp. 177–188). SIAM.

Boley, M., & Grosskreutz, H. (2008). A randomized approach for approximating the number of frequent sets. In *ICDM* (pp. 43–52). IEEE Computer Society.

Borgelt, C., & Berthold, M. R. (2002). Mining molecular fragments: Finding relevant substructures of molecules. In *2002 IEEE international conference on Data mining, 2002. ICDM 2003. Proceedings* (pp. 51–58). IEEE.

Brandes, U., Gaertler, M., & Wagner, D. (2007). Engineering graph clustering: Models and experimental evaluation. *ACM Journal of Experimental Algorithmics*, *12*(1.1), 1–26.

Bringmann, B., Nijssen, S., Tatti, N., Vreeken, J., & Zimmermann, A. (2011). Mining sets of patterns (tutorial at ICDM).

Bringmann, B., & Zimmermann, A. (2009). One in a million: Picking the right patterns. *Knowledge and Information Systems*, *18*(1), 61–81.

Brohee, S., & Van Helden, J. (2006). Evaluation of clustering algorithms for protein-protein interaction networks. *BMC Bioinformatics*, *7*(1), 488.

Carnein, M., Assenmacher, D., & Trautmann, H. (2017). An empirical comparison of stream clustering algorithms. In *Proceedings of the computing frontiers conference* (pp. 361–366). New York: ACM.

Casas-Garriga, G. (2003). Discovering unbounded episodes in sequential data. In N. Lavrac, D. Gamberger, H. Blockeel, & L. Todorovski (Eds.), *PKDD* (Vol. 2838, pp. 83–94). Berlin, Heidelberg: Springer.

Celebi, M. E., Kingravi, H. A., & Vela, P. A. (2013). A comparative study of efficient initialization methods for the k-means clustering algorithm. *Expert Systems with Applications*, *40*(1), 200–210.

Chakrabarti, D., & Faloutsos, C. (2006). Graph mining: Laws, generators, and algorithms. *ACM Computing Surveys*, *38*(1), 2–es.

Charrad, M., Ghazzali, N., Boiteau, V., & Niknafs, A. (2012). Nbclust package: finding the relevant number of clusters in a dataset. *Journal of Statistical Software*, *61*, 1-36.

Chen, M., Weinberger, K. Q., & Blitzer, J. (2011). Co-training for domain adaptation. In *Advances in neural information processing systems* (pp. 2456–2464). Red Hook: Curran Associates Inc.

Chi, Y., Muntz, R. R., Nijssen, S., & Kok, J. N. (2005). Frequent subtree mining–an overview. *Fundamenta Informaticae*, *66*(1–2), 161–198.

Chi, Y., Yang, Y., Xia, Y., & Muntz, R. R. (2004). CMTreeminer: Mining both closed and maximal frequent subtrees. In *Pacific-Asia conference on knowledge discovery and data mining* (pp. 63–73). Berlin, Heidelberg: Springer.

Chouikhi, H., Charrad, M., & Ghazzali, N. (2015). A comparison study of clustering validity indices. In *2015 global summit on Computer & information technology (GSCIT)* (pp. 1–4). IEEE.

Coenen, F., & Leng, P. (2005). Obtaining best parameter values for accurate classification. In J. Han, B. W. Wah, V. Raghavan, X. Wu, & R. Rastogi (Eds.), *Proceedings of the fifth IEEE international conference on data mining* (pp. 597–600). Houston, TX: IEEE.

Cooper, C., & Zito, M. (2007). Realistic synthetic data for testing association rule mining algorithms for market basket databases. In J. N. Kok, J. Koronacki, R. L. de Mántaras, S. Matwin, D. Mladenic, & A. Skowron (Eds.), *PKDD* (Vol. 4702, pp. 398–405). Berlin, Heidelberg: Springer.

Dao, V.-L., Bothorel, C., & Lenca, P. (2017). Community detection methods can discover better structural clusters than ground-truth communities. In *Proceedings of the 2017 IEEE/ACM international conference on advances in social networks analysis and mining 2017* (pp. 395–400). New York: ACM.

Dao, V.-L., Bothorel, C., & Lenca, P. (2018). An empirical characterization of community structures in complex networks using a bivariate map of quality metrics. *arXiv preprint arXiv:1806.01386*.

Dau, H. A., Keogh, E., Kamgar, K., Yeh, C.-C. M., Zhu, Y., Gharghabi, S., … Batista, G. (2018). *The UCR time series classification archive*. (https://www.cs.ucr.edu/eamonn/time_series_data_2018/)

De Bie, T. (2011). Maximum entropy models and subjective interestingness: An application to tiles in binary databases. *Data Mining and Knowledge Discovery*, *23*(3), 407–446.

De Raedt, L., & Zimmermann, A. (2007). Constraint-based pattern set mining. In *Proceedings of the seventh SIAM international conference on data mining*. SIAM.

Delling, D., Gaertler, M., Görke, R., Nikoloski, Z., & Wagner, D. (2006). *How to evaluate clustering techniques*. Univ., Fak. für Informatik, Bibliothek.

Dempster, A., Laird, N., & Rubin, D. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, B*, *39*, 1–39.

Dheeru, D., & Karra Taniskidou, E. (2017). *UCI machine learning repository*. Retrieved from http://archive.ics.uci.edu/ml.

Donahue, J., Hoffman, J., Rodner, E., Saenko, K., & Darrell, T. (2013). Semi-supervised domain adaptation with instance constraints. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 668–675). IEEE.

Downs, J., & Vogel, E. (1993). A plant-wide industrial process control problem. *Computers & Chemical Engineering*, *17*(3), 245–255. Retrieved from http://www.sciencedirect.com/science/article/pii/009813549380018I (Industrial challenge problems in process control). https://doi.org/10.1016/0098-1354(93)80018-I

Dries, A. (2015). Declarative data generation with problog. In *Proceedings of the sixth international symposium on information and communication technology* (pp. 17–24). New York: ACM.

Ester, M., Kriegel, H.-P., Sander, J., & Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. In *KDD* (Vol. 96, pp. 226–231). AAAI Press.

Fahad, A., Alshatri, N., Tari, Z., Alamri, A., Khalil, I., Zomaya, A. Y., … Bouras, A. (2014). A survey of clustering algorithms for big data: Taxonomy and empirical analysis. *IEEE Transactions on Emerging Topics in Computing*, *2*(3), 267–279.

Färber, I., Günnemann, S., Kriegel, H.-P., Kröger, P., Müller, E., Schubert, E., … Zimek, A. (2010). On using class-labels in evaluation of clusterings. In *Multiclust: 1st international workshop on discovering, summarizing and using multiple clusterings held in conjunction with KDD* (p. 1).

Fayyad, U., Piatetsky-Shapiro, G., & Smyth, P. (1996). From data mining to knowledge discovery in databases. *AI Magazine*, *17*(3), 37.

Fernando, B., Habrard, A., Sebban, M., & Tuytelaars, T. (2013). Unsupervised visual domain adaptation using subspace alignment. In *Proceedings of the IEEE international conference on computer vision* (pp. 2960–2967). IEEE.

Flouvat, F., Marchi, F. D., & Petit, J.-M. (2010). A new classification of datasets for frequent itemsets. *Journal of Intelligent Information System*, *34*(1), 1–19.

Fortunato, S. (2010). Community detection in graphs. *Physics Reports*, *486*(3–5), 75–174.

Frasch, J. V., Lodwich, A., Shafait, F., & Breuel, T. M. (2011). A bayes-true data generator for evaluation of supervised and unsupervised learning methods. *Pattern Recognition Letters*, *32*(11), 1523–1531.

Geerts, F., Goethals, B., & den Bussche, J. V. (2005). Tight upper bounds on the number of candidate patterns. *ACM Trans. Database Syst.*, *30*(2), 333–363.

Gionis, A., Mannila, H., Mielikäinen, T., & Tsaparas, P. (2007). Assessing data mining results via swap randomization. *TKDD*, *1*(3), 14–es.

Girvan, M., & Newman, M. E. (2002). Community structure in social and biological networks. *Proceedings of the National Academy of Sciences of the United States of America*, *99*(12), 7821–7826.

Goethals, B., Moens, S., & Vreeken, J. (2011). Mime: A framework for interactive visual pattern mining. In *Proceedings of the 17th ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 757–760). New York: ACM.

Goethals, B., & Zaki, M. J. (Eds.). (2003). *FIMI '03, frequent itemset mining implementations, proceedings of the ICDM 2003 workshop on frequent itemset mining implementations, 19 December 2003, Melbourne, Florida* (Vol. 90). Retrieved from CEUR-WS.org.

Gouda, K., & Zaki, M. J. (2005). Genmax: An efficient algorithm for mining maximal frequent itemsets. *Data Mining and Knowledge Discovery*, *11*(3), 223–242.

Guerra, L., Robles, V., Bielza, C., & Larrañaga, P. (2012). A comparison of clustering quality indices using outliers and noise. *Intelligent Data Analysis*, *16*(4), 703–715.

Halkidi, M., & Vazirgiannis, M. (2008). A density-based cluster validity approach using multi-representatives. *Pattern Recognition Letters*, *29*(6), 773–786.

Hall, N. G., & Posner, M. E. (2010). The generation of experimental data for computational testing in optimization. In *Experimental methods for the analysis of optimization algorithms* (pp. 73–101). Berlin, Heidelberg: Springer.

Hämäläinen, J., Jauhiainen, S., & Kärkkäinen, T. (2017). Comparison of internal clustering validation indices for prototype-based clustering. *Algorithms*, *10*(3), 105.

Han, J., Pei, J., Mortazavi-Asl, B., Chen, Q., Dayal, U., & Hsu, M. (2000). Freespan: Frequent pattern-projected sequential pattern mining. In R. Ramakrishnan, S. J. Stolfo, R. J. Bayardo, & I. Parsa (Eds.), *KDD* (pp. 355–359). New York: ACM.

Han, J., Pei, J., & Yin, Y. (2000). Mining frequent patterns without candidate generation. In W. Chen, J. F. Naughton, & P. A. Bernstein (Eds.), *SIGMOD conference* (pp. 1–12). New York: ACM.

Hand, D. J. (2002). Pattern detection and discovery. In *Pattern detection and discovery* (pp. 1–12). Berlin, Heidelberg: Springer.

Harenberg, S., Bello, G., Gjeltema, L., Ranshous, S., Harlalka, J., Seay, R., … Samatova, N. (2014). Community detection in large-scale networks: A survey and empirical evaluation. *Wiley Interdisciplinary Reviews: Computational Statistics*, *6*(6), 426–439.

Hassani, M., & Seidl, T. (2017). Using internal evaluation measures to validate the quality of diverse stream clustering algorithms. *Vietnam Journal of Computer Science*, *4*(3), 171–183.

He, J., Tan, A.-H., Tan, C.-L., & Sung, S.-Y. (2004). On quantitative evaluation of clustering systems. In *Clustering and information retrieval* (pp. 105–133). Berlin, Heidelberg: Springer.

Heck, D., Schatz, G., Knapp, J., Thouw, T., & Capdevielle, J. (1998). *Corsika: A monte carlo code to simulate extensive air showers* (Tech. Rep.).

Hric, D., Darst, R. K., & Fortunato, S. (2014). Community detection in networks: Structural communities versus ground truth. *Physical Review E*, *90*(6), 062805.

Huan, J., Wang, W., & Prins, J. (2003). Efficient mining of frequent subgraphs in the presence of isomorphism. In *Third IEEE International Conference on Data Mining* (pp. 549-552). IEEE.

Huang, A. (2008). Similarity measures for text document clustering. In *Proceedings of the sixth New Zealand computer science research student conference (nzcsrsc2008), Christchurch, New Zealand* (pp. 49–56).

Inokuchi, A., Washio, T., & Motoda, H. (2000). An apriori-based algorithm for mining frequent substructures from graph data. In D. A. Zighed, H. J. Komorowski, & J. M. Zytkow (Eds.), *PKDD* (Vol. 1910, pp. 13–23). Berlin, Heidelberg: Springer.

Ioannidis, J. P. (2005). Why most published research findings are false. *PLoS Medicine*, *2*(8), e124.

Jain, A. K., Murty, M. N., & Flynn, P. J. (1999). Data clustering: A review. *ACM Computing Surveys (CSUR)*, *31*(3), 264–323.

Jiang, J. (2008). *A literature survey on domain adaptation of statistical classifiers* (1–12). Retrieved from http://sifaka.cs.uiuc.edu/jiang4/domainadaptation/survey.

Kaytoue, M., Plantevit, M., Zimmermann, A., Bendimerad, A., & Robardet, C. (2017). Exceptional contextual subgraph mining. *Machine Learning*, *106*(8), 1171–1211.

Klösgen, W. (1996). Explora: A multipattern and multistrategy discovery assistant. In U. Fayyad, G. Piatetsky-Shapiro, P. Smyth, & R. Uthurusamy (Eds.), *Advances in knowledge discovery and data mining*. Cambridge, MA: The MIT Press.

Knobbe, A. J., & Ho, E. K. Y. (2006). Pattern teams. In J. Fürnkranz, T. Scheffer, & M. Spiliopoulou (Eds.), *10th European conference on principles and practice of knowledge discovery in databases* (pp. 577–584). Berlin, Heidelberg: Springer.

Kohonen, T. (1990). The self-organizing map. *Proceedings of the IEEE*, *78*(9), 1464–1480.

Kovács, F., Legány, C., & Babos, A. (2005). Cluster validity measurement techniques. In *6th International symposium of Hungarian researchers on computational intelligence*. Stevens Point, Wisconsin: World Scientific and Engineering Academy and Society (WSEAS).

Kralj Novak, P., Lavrač, N., & Webb, G. I. (2009). Supervised descriptive rule discovery: A unifying survey of contrast set, emerging pattern and subgroup mining. *Journal of Machine Learning Research*, *10*, 377–403.

Kriegel, H.-P., Schubert, E., & Zimek, A. (2017). The (black) art of runtime evaluation: Are we comparing algorithms or implementations? *Knowledge and Information Systems*, *52*(2), 341–378.

Kuramochi, M., & Karypis, G. (2001). Frequent subgraph discovery. In N. Cercone, T. Y. Lin, & X. Wu (Eds.), *Icdm* (pp. 313–320). IEEE Computer Society.

Laland, K. N. (2018). *Darwin's unfinished symphony: How culture made the human mind*. Princeton, NJ: Princeton University Press.

Lam, H. T., Mörchen, F., Fradkin, D., & Calders, T. (2014). Mining compressing sequential patterns. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, *7*(1), 34–52.

Lancichinetti, A., & Fortunato, S. (2009a). Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities. *Physical Review E*, *80*(1), 016118.

Lancichinetti, A., & Fortunato, S. (2009b). Community detection algorithms: A comparative analysis. *Physical Review E*, *80*(5), 056117.

Lancichinetti, A., Fortunato, S., & Radicchi, F. (2008). Benchmark graphs for testing community detection algorithms. *Physical Review E*, *78*(4), 046110.

Lancichinetti, A., Kivelä, M., Saramäki, J., & Fortunato, S. (2010). Characterizing the community structure of complex networks. *PLoS One*, *5*(8), e11976.

Laxman, S., Sastry, P. S., & Unnikrishnan, K. P. (2005). Discovering frequent episodes and learning hidden markov models: A formal connection. *IEEE Transactions on Knowledge and Data Engineering*, *17*(11), 1505–1517.

Lee, C., & Cunningham, P. (2014). Community detection: Effective evaluation on large social networks. *Journal of Complex Networks*, *2*(1), 19–37.

Lenca, P., Meyer, P., Vaillant, B., & Lallich, S. (2008). On selecting interestingness measures for association rules: User oriented description and multiple criteria decision aid. *European Journal of Operational Research*, *184*(2), 610–626.

Leskovec, J., & Krevl, A. (2014). *SNAP Datasets: Stanford large network dataset collection*. http://snap.stanford.edu/data.

Leskovec, J., Lang, K. J., Dasgupta, A., & Mahoney, M. W. (2008). Statistical properties of community structure in large social and information networks. In J. Huai, et al. (Eds.), *Www* (pp. 695–704). New York: ACM.

Leskovec, J., Lang, K. J., & Mahoney, M. (2010). Empirical comparison of algorithms for network community detection. In *Proceedings of the 19th international conference on world wide web* (pp. 631–640). New York: ACM.

Lhote, L., Rioult, F., & Soulet, A. (2005). Average number of frequent (closed) patterns in bernouilli and markovian databases. In J. Han, B. W. Wah, V. Raghavan, X. Wu, & R. Rastogi (Eds.), *ICDM* (pp. 713–716). Houston, TX: IEEE.

Li, D., Deogun, J., Spaulding, W., & Shuart, B. (2004). Towards missing data imputation: A study of fuzzy k-means clustering method. In *International conference on rough sets and current trends in computing* (pp. 573–579). Berlin, Heidelberg: Springer.

Liu, L., Cheng, L., Liu, Y., Jia, Y., & Rosenblum, D. S. (2016). Recognizing complex activities by a probabilistic interval-based model. In *Thirtieth AAAI conference on artificial intelligence*. AAAI Press.

Liu, Y., Li, Z., Xiong, H., Gao, X., & Wu, J. (2010). Understanding of internal clustering validation measures. In *2010 IEEE 10th international conference on data mining (ICDM)* (pp. 911–916). IEEE.

Liu, Y., Nie, L., Han, L., Zhang, L., & Rosenblum, D. S. (2015). Action2activity: Recognizing complex activities from sensor data. In *Twenty-fourth international joint conference on artificial intelligence*.

Ma, L., & Srinivasan, S. (2015). Synthetic population generation with multilevel controls: A fitness-based synthesis approach and validations. *Computer-Aided Civil and Infrastructure Engineering*, *30*(2), 135–150.

Mabroukeh, N. R., & Ezeife, C. I. (2010). A taxonomy of sequential pattern mining algorithms. *ACM Computing Surveys (CSUR)*, *43*(1), 3.

MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability* (Vol. 1, pp. 281–297). Berkeley, Calif: University of California Press.

Macy, M. W., & Willer, R. (2002). From factors to actors: Computational sociology and agent-based modeling. *Annual Review of Sociology*, *28*(1), 143–166.

Mampaey, M., Vreeken, J., & Tatti, N. (2012). Summarizing data succinctly with the most informative itemsets. *TKDD*, *6*(4), 16. https://doi.org/10.1145/2382577.2382580

Mangiameli, P., Chen, S. K., & West, D. (1996). Comparison of SOM neural network and hierarchical clustering. *European Journal of Operational Research*, *93*(2), 402–471.

Mannila, H., & Toivonen, H. (1995). Discovering frequent episodes in sequences. In *Proceedings of the first international conference on knowledge discovery and data mining (KDD '95)* (pp. 210–215). AAAI Press.

Mannila, H., Toivonen, H., & Verkamo, A. I. (1997). Discovery of frequent episodes in event sequences. *Data Mining and Knowledge Discovery*, *1*(3), 259–289.

Maulik, U., & Bandyopadhyay, S. (2002). Performance evaluation of some clustering algorithms and validity indices. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *24*(12), 1650–1654.

Méger, N., & Rigotti, C. (2004). Constraint-based mining of episode rules and optimal window sizes. In J.-F. Boulicaut, F. Esposito, F. Giannotti, & D. Pedreschi (Eds.), *PKDD* (Vol. 3202, pp. 313–324). Berlin, Heidelberg: Springer.

Meilă, M., & Heckerman, D. (2001). An experimental comparison of model-based clustering methods. *Machine Learning*, *42*(1–2), 9–29.

Melnykov, V., Chen, W.-C., & Maitra, R. (2012). Mixsim: An r package for simulating data to study performance of clustering algorithms. *Journal of Statistical Software*, *51*(12), 1.

Milligan, G. W. (1980). An examination of the effect of six types of error perturbation on fifteen clustering algorithms. *Psychometrika*, *45*(3), 325–342.

Milligan, G. W. (1985). An algorithm for generating artificial test clusters. *Psychometrika*, *50*(1), 123–127.

Milligan, G. W., & Cooper, M. C. (1985). An examination of procedures for determining the number of clusters in a data set. *Psychometrika*, *50*(2), 159–179.

Milligan, G. W., & Cooper, M. C. (1988). A study of standardization of variables in cluster analysis. *Journal of Classification*, *5*(2), 181–204.

Milo, R., Shen-Orr, S., Itzkovitz, S., Kashtan, N., Chklovskii, D., & Alon, U. (2002). Network motifs: Simple building blocks of complex networks. *Science*, *298*(5594), 824–827.

Mishra, R., Shukla, S., Arora, D., & Kumar, M. (2011). An effective comparison of graph clustering algorithms via random graphs. *International Journal of Computer Applications*, *22*(1), 22–27.

Moradi, F., Olovsson, T., & Tsigas, P. (2012). An evaluation of community detection algorithms on large-scale email traffic. In *International symposium on experimental algorithms* (pp. 283–294). Berlin, Heidelberg: Springer.

Morey, L. C., & Agresti, A. (1984). The measurement of classification agreement: An adjustment to the rand statistic for chance agreement. *Educational and Psychological Measurement*, *44*(1), 33–37.

Morik, K., Boulicaut, J., & Siebes, A. (Eds.). (2005). *Local pattern detection, international seminar, dagstuhl castle, Germany, April 12–16, 2004, revised selected papers* (Vol. 3539). Berlin, Heidelberg: Springer. https://doi.org/10.1007/b137601

Moulavi, D., Jaskowiak, P. A., Campello, R. J., Zimek, A., & Sander, J. (2014). Density-based clustering validation. In *Proceedings of the 2014 SIAM international conference on data mining* (pp. 839–847). SIAM.

Müller, K., & Axhausen, K. W. (2011). Hierarchical IPF: Generating a synthetic population for Switzerland. *Arbeitsberichte Verkehrs-und Raumplanung*, *718*.

Mutter, S. (2004). Classification using association rules. In *A thesis of Diploma of computer science*. Aotearoa, New Zealand: *University of Freiburg, Hamilton*.

Namazi-Rad, M.-R., Mokhtarian, P., & Perez, P. (2014). Generating a dynamic synthetic population–using an age-structured two-sex model for household dynamics. *PLoS One*, *9*(4), e94761.

Nijssen, S., & Kok, J. (2006). Frequent subgraph miners: Runtimes don't say everything. In T. Gärtner, G. Garriga, & T. Meinl (Eds.), *Proceedings of the workshop on mining and learning with graphs* (pp. 173–180) Retrieved from https://lirias.kuleuven.be/handle/123456789/134452

Nijssen, S., & Kok, J. N. (2004). A quickstart in frequent structure mining can make a difference. In *Proceedings of the tenth ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 647–652). New York: ACM.

Orman, G. K., & Labatut, V. (2009). A comparison of community detection algorithms on artificial networks. In *International conference on discovery science* (pp. 242–256). Berlin, Heidelberg: Springer.

Orman, G. K., Labatut, V., & Cherifi, H. (2011). Qualitative comparison of community detection algorithms. In H. Cherifi, J. M. Zain, & E. El-Qawasmeh (Eds.), *DICTAP (2)* (Vol. 167, pp. 265–279). Berlin, Heidelberg: Springer.

Orman, G. K., Labatut, V., & Cherifi, H. (2013). Towards realistic artificial benchmark for community detection algorithms evaluation. *International Journal of Web Based Communities*, *9*(3), 349–370.

Palmerini, P., Orlando, S., & Perego, R. (2004). Statistical properties of transactional databases. In H. Haddad, A. Omicini, R. L. Wainwright, & L. M. Liebrock (Eds.), *SAC* (pp. 515–519). New York: ACM.

Pan, S. J., & Yang, Q. (2009). A survey on transfer learning. *IEEE Transactions on Knowledge & Data Engineering*, *22*(10), 1345–1359.

Peel, L., Larremore, D. B., & Clauset, A. (2017). The ground truth about metadata and community detection in networks. *Science Advances*, *3*(5), e1602548.

Pei, J., Han, J., Mortazavi-Asl, B., Pinto, H., Chen, Q., Dayal, U., & Hsu, M. (2001). Prefixspan: Mining sequential patterns by prefix-projected growth. In D. Georgakopoulos & A. Buchmann (Eds.), *ICDE* (pp. 215–224). IEEE Computer Society.

Pei, J., Han, J., Mortazavi-Asl, B., Wang, J., Pinto, H., Chen, Q., … Hsu, M.-C. (2004). Mining sequential patterns by pattern-growth: The prefixspan approach. *IEEE Transactions on Knowledge & Data Engineering*, *16*(11), 1424–1440.

Pei, Y., & Zaïane, O. (2006). *A synthetic data generator for clustering and outlier analysis* (Tech. Rep.).

Pennock, D. M., & Stout, Q. F. (1996). Exploiting a theory of phase transitions in three-satisfiability problems. In *AAAI/IAAI* (Vol. 1, pp. 253–258). AAAI Press.

Pereira, C. M., & de Mello, R. F. (2011). A comparison of clustering algorithms for data streams. In *Integrated computing technology* (pp. 59–74). Berlin, Heidelberg: Springer.

Prakash, B. A., Vreeken, J., & Faloutsos, C. (2014). Efficiently spotting the starting points of an epidemic in a large graph. *Knowledge and Information Systems*, *38*(1), 35–59. Retrieved from. https://doi.org/10.1007/s10115-013-0671-5

Qiu, W., & Joe, H. (2006). Generation of random clusters with specified degree of separation. *Journal of Classification*, *23*(2), 315–334.

Ramesh, G., Maniatty, W., & Zaki, M. J. (2003). Feasible itemset distributions in data mining: Theory and application. In *PODS* (pp. 284–295). New York: ACM.

Ramesh, G., Zaki, M. J., & Maniatty, W. (2005). Distribution-based synthetic database generation techniques for itemset mining. In *IDEAS* (pp. 307–316). IEEE Computer Society.

Rand, W. M. (1971). Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, *66*(336), 846–850.

Reaves, M. L., Sinha, S., Rabinowitz, J. D., Kruglyak, L., & Redfield, R. J. (2012). Absence of detectable arsenate in dna from arsenate-grown gfaj-1 cells. *Science*, *337*(6093), 470–473.

Rula, A., Maurino, A., & Batini, C. (2016). Data quality issues in linked open data. In *Data and information quality* (pp. 87–112). Berlin, Heidelberg: Springer.

Sadikin, M. (2014). *A binary matrix synthetic data and its bi-set ground truth generator*. AM Publications Group.

Shirkhorshidi, A. S., Aghabozorgi, S., & Wah, T. Y. (2015). A comparison study on similarity and dissimilarity measures in clustering continuous data. *PLoS One*, *10*(12), e0144059.

Smets, K., & Vreeken, J. (2012). Slim: Directly mining descriptive patterns. In *Proceedings of the 2012 SIAM international conference on data mining* (pp. 236–247). SIAM.

Srikant, R., & Agrawal, R. (1996). Mining sequential patterns: Generalizations and performance improvements. In P. M. G. Apers, M. Bouzeghoub, & G. Gardarin (Eds.), *EDBT* (Vol. 1057, pp. 3–17). Berlin, Heidelberg: Springer.

Steinbach, M., Karypis, G., & Kumar, V. (2000). A comparison of document clustering techniques. In *KDD workshop on text mining*.

Steinhaus, H. (1956). Sur la division des corp materiels en parties. *Bulletin of the Polish Academy of Sciences*, *1*(804), 801.

Steinley, D. (2003). Local optimain K-means clustering: what you don't know may hurt. *Psychological methods*, *8*(3), 294–304.

Steinley, D., & Brusco, M. J. (2007). Initializing k-means batch clustering: A critical evaluation of several techniques. *Journal of Classification*, *24*(1), 99–121.

Steinley, D., & Henson, R. (2005). Oclus: An analytic method for generating clusters with known overlap. *Journal of Classification*, *22*(2), 221–250.

Strehl, A., Ghosh, J., & Mooney, R. (2000). Impact of similarity measures on web-page clustering. In *Workshop on artificial intelligence for web search (AAAI 2000)* (Vol. 58, p. 64).

Tan, P.-N., Kumar, V., & Srivastava, J. (2002). Selecting the right interestingness measure for association patterns. In *KDD* (pp. 32–41). New York: ACM.

Tatti, N., & Vreeken, J. (2012a). Discovering descriptive tile trees - by mining optimal geometric subtiles. In P. A. Flach, T. D. Bie, & N. Cristianini (Eds.), *Machine learning and knowledge discovery in databases—European conference, ECML PKDD 2012, Bristol, Uk, September 24–28, 2012. Proceedings, part I* (Vol. 7523, pp. 9–24). Berlin, Heidelberg: Springer. https://doi.org/10.1007/978-3-642-33460-3_6

Tatti, N., & Vreeken, J. (2012b). The long and the short of it: Summarising event sequences with serial episodes. In Q. Yang, D. Agarwal, & J. Pei (Eds.), *The 18th ACM SIGKDD international conference on knowledge discovery and data mining, KDD '12, Beijing, China, August 12–16, 2012* (pp. 462–470). New York: ACM. https://doi.org/10.1145/2339530.2339606

Uno, T., Asai, T., Uchida, Y., & Arimura, H. (2003). Lcm: An efficient algorithm for enumerating frequent closed item sets. In *FIMI* (Vol. 90). CEUR-WS.

Uno, T., Kiyomi, M., & Arimura, H. (2004). Lcm ver. 2: Efficient mining algorithms for frequent/closed/maximal itemsets. In *FIMI* (Vol. 126). CEUR-WS.

Vaillant, B., Lenca, P., & Lallich, S. (2004). A clustering of interestingness measures. In E. Suzuki & S. Arikawa (Eds.), *Discovery science* (Vol. 3245, pp. 290–297). Berlin, Heidelberg: Springer.

Van Craenendonck, T., & Blockeel, H. (2015). Using internal validity measures to compare clustering algorithms. In *Benelearn 2015 poster presentations (online)* (pp. 1–8).

Van Craenendonck, T., & Blockeel, H. (2017). Constraint-based clustering selection. *Machine Learning*, *106*(9–10), 1497–1521.

Van Craenendonck, T., Dumancic, S., & Blockeel, H. (2017). Cobra: A fast and simple method for active clustering with pairwise constraints. In *Proceedings of the twenty-sixth international joint conference on artificial intelligence* (pp. 2871–2877). Berlin, Heidelberg: Springer.

Van Leeuwen, M. (2014). Interactive data exploration using pattern mining. In *Interactive knowledge discovery and data mining in biomedical informatics* (pp. 169–182). Berlin, Heidelberg: Springer.

van Leeuwen, M., & Ukkonen, A. (2014). Fast estimation of the pattern frequency spectrum. In T. Calders, F. Esposito, E. Hüllermeier, & R. Meo (Eds.), *Machine learning and knowledge discovery in databases—European conference, ECML PKDD 2014, Nancy, France, September 15–19, 2014. Proceedings, part II* (Vol. 8725, pp. 114–129). Berlin, Heidelberg: Springer. https://doi.org/10.1007/978-3-662-44851-9_8

Van Mechelen, I., Boulesteix, A.-L., Dangl, R., Dean, N., Guyon, I., Hennig, C., … Steinley, D. (2018). Benchmarking in cluster analysis: A white paper. *arXiv preprint arXiv:1809.10496*.

Vanschoren, J., Blockeel, H., Pfahringer, B., & Holmes, G. (2012). Experiment databases—A new way to share, organize and learn from experiments. *Machine Learning*, *87*(2), 127–158. https://doi.org/10.1007/s10994-011-5277-0

Veillon, L.-M., Bourgne, G., & Soldano, H. (2017). Effect of network topology on neighbourhood-aided collective learning. In *Conference on computational collective intelligence technologies and applications* (pp. 202–211). Berlin, Heidelberg: Springer.

Vendramin, L., Campello, R. J., & Hruschka, E. R. (2009). On the comparison of relative clustering validity criteria. In *Proceedings of the 2009 SIAM international conference on data mining* (pp. 733–744). SIAM.

Vendramin, L., Campello, R. J., & Hruschka, E. R. (2010). Relative clustering validity criteria: A comparative overview. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, *3*(4), 209–235.

Verma, D., & Meilă, M. (2003). *A comparison of spectral clustering algorithms* (Tech. Rep.). University of Washington.

Vilalta, R., & Drissi, Y. (2002). A perspective view and survey of meta-learning. *Artificial Intelligence Review*, *18*(2), 77–95.

Vreeken, J., van Leeuwen, M., & Siebes, A. (2007). Preserving privacy through data generation. In N. Ramakrishnan & O. Zaïane (Eds.), *ICDM* (pp. 685–690). IEEE Computer Society.

Vreeken, J., van Leeuwen, M., & Siebes, A. (2011). Krimp: Mining itemsets that compress. *Data Mining and Knowledge Discovery*, *23*(1), 169–214.

Wagenseller III, P., & Wang, F. (2017). Size matters: A comparative analysis of community detection algorithms. *arXiv preprint arXiv:1712.01690*.

Wang, J., & Han, J. (2004). Bide: Efficient mining of frequent closed sequences. In *20th international conference on data engineering, 2004. Proceedings* (pp. 79–90). IEEE.

Wang, M., Wang, C., Yu, J. X., & Zhang, J. (2015). Community detection in social networks: An in-depth benchmarking study with a procedure-oriented framework. *Proceedings of the VLDB Endowment*, *8*(10), 998–1009.

Webb, G. I. (2007). Discovering significant patterns. *Machine Learning*, *68*(1), 1–33.

Webb, G. I., & Vreeken, J. (2014). Efficient discovery of the most interesting associations. *Transactions on Knowledge Discovery from Data*, *8*(3), 15–11.

Wörlein, M., Meinl, T., Fischer, I., & Philippsen, M. (2005). A quantitative comparison of the subgraph miners MoFa, gSpan, FFSM, and Gaston. In A. Jorge, L. Torgo, P. Brazdil, R. Camacho, & J. Gama (Eds.), *PKDD* (pp. 392–403). Berlin, Heidelberg: Springer.

Wu, H., Ning, Y., Chakraborty, P., Vreeken, J., Tatti, N., & Ramakrishnan, N. (2018). Generating realistic synthetic population datasets. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, *12*(4), 45.

Yan, X., & Han, J. (2002). gSpan: Graph-based substructure pattern mining. In *ICDM* (pp. 721–724). IEEE Computer Society.

Yang, J., & Leskovec, J. (2015). Defining and evaluating network communities based on ground-truth. *Knowledge and Information Systems*, *42*(1), 181–213.

Yang, Z., Algesheimer, R., & Tessone, C. J. (2016). A comparative analysis of community detection algorithms on artificial networks. *Scientific Reports*, *6*, 30750.

Zaki, M. J. (2000). Scalable algorithms for association mining. *IEEE Transactions on Knowledge and Data Engineering*, *12*(3), 372–390.

Zaki, M. J. (2001). Spade: An efficient algorithm for mining frequent sequences. *Machine Learning*, *42*(1/2), 31–60.

Zaki, M. J. (2002). Efficiently mining frequent trees in a forest. In *KDD* (pp. 71–80). New York: ACM.

Zaki, M. J., & Hsiao, C.-J. (1999). *Charm: An efficient algorithm for closed association rule mining (Tech. Rep.)*. Computer Science Department, Rensselaer Polytechnic Institute.

Zaki, M. J., & Hsiao, C.-J. (2002). Charm: An efficient algorithm for closed itemset mining. In R. L. Grossman, J. Han, V. Kumar, H. Mannila, & R. Motwani (Eds.), *SDM*. SIAM.

Zhang, S., Zhang, J., Zhu, X., Qin, Y., & Zhang, C. (2008). Missing value imputation based on data clustering. In *Transactions on computational science i* (pp. 128–138). Berlin, Heidelberg: Springer.

Zhang, Z., Huang, K., & Tan, T. (2006). Comparison of similarity measures for trajectory clustering in outdoor surveillance scenes. In *18th international conference on pattern recognition, 2006. ICPR 2006* (Vol. 3, pp. 1135–1138). IEEE.

Zhao, Y., & Karypis, G. (2002). Evaluation of hierarchical clustering algorithms for document datasets. In *Proceedings of the eleventh international conference on information and knowledge management* (pp. 515–524). New York: ACM.

Zheng, Z., Kohavi, R., & Mason, L. (2001). Real world performance of association rule algorithms. In *KDD* (pp. 401–406). New York: ACM.

Zimmermann, A. (2013). Objectively evaluating condensed representations and interestingness measures for frequent itemset mining. *Journal of Intelligent Information Systems*, *45*, 1–19.

Zimmermann, A. (2014). Understanding episode mining techniques: Benchmarking on diverse, realistic, artificial data. *Intelligent Data Analysis*, *18*(5), 761–791.

Zimmermann, A. (2015). The data problem in data mining. *ACM SIGKDD Explorations Newsletter*, *16*(2), 38–45.

How to cite this article: Zimmermann A. Method evaluation, parameterization, and result validation in unsupervised data mining: A critical survey. *WIREs Data Mining Knowl Discov*. 2020;10:e1330. https://doi.org/10.1002/widm.1330