Saarland University
Center for Bioinformatics
Master's Program in Bioinformatics

Master's Thesis

Development of an Integrated Clustering Evaluation Framework for Cluster Analysis

submitted by

Christian Wiwie

on May 13, 2013

Supervisor
Prof. Dr. Jan Baumbach

Advisor
Richard Röttger

Reviewers
Prof. Dr. Jan Baumbach
Prof. Dr. Hans-Peter Lenhof

Wiwie, Christian
Development of an Integrated Clustering Evaluation Framework for Cluster Analysis
Master's Thesis in Bioinformatics
Universität des Saarlandes
Saarbrücken, Germany
May 2013

Abstract

In recent years the amount of biological data generated by large-scale experiments has grown rapidly. The increasing size of data sets requires computational analyses searching for structure in data in an automatized way. Clustering methods are unsupervised learning approaches tackling this challenge and structuring data by arranging similar entities together in groups. Many factors influence clustering processes like for example available clustering methods, cluster validity indices or data formats. This variety complicates cluster studies and makes them more difficult and time-consuming for the researcher.

In this work we present ClustEval, an open-source framework automatizing and simplifying clustering tasks by integrating and unifying many relevant factors within a single project. Our framework provides the ambitious researcher with a supporting tool when carrying out cluster studies on the one hand, and the interested practitioner with a website as a place to go, when there is need for knowledge derived from previously assessed cluster studies. ClustEval tackles, among other things, detection of optimal density parameters, comprehensive analysis of data sets for cluster tendency, clustering validation with various cluster validity indices and automatized format conversions. A website is developed to summarize and visualize clustering and analysis results in interactive tables and charts. We evaluated ClustEval on gene expression levels, protein sequence similarities and synthetic machine learning data sets with frequently used clustering methods including Affinity Propagation, Hierarchical Clustering, K-Means and Spectral Clustering.

ClustEval has been implemented as a standalone application based on a server-client-architecture, utilizing parallelism to accelerate calculations and providing plug-in interfaces for dynamic extensions. ClustEval is available online at `http://clusteval.mmci.uni-saarland.de`. Visitors can use the website for cluster results inspection or to download our framework and install it on a local server. ClustEval is published under the *GNU GPL* providing the possibility to modify and extend it to individual needs.

Declaration

I hereby confirm that this thesis is my own work and that I have documented all sources used.

I hereby declare that the submitted digital and hardcopy versions of this thesis correspond to each other. I give permission to the Saarland University to duplicate and publish this work.

Saarbrücken, May 13, 2013

Christian Wiwie

Acknowledgments

I would like to thank my advisor, Richard Röttger, for his well-founded advice and the time he invested into my thesis. I owe my gratitude to Jan Baumbach for the inspiring topic, his continuous support, the possibility to make independent decisions and the familiar atmosphere in his group.

Also, I would like to thank Hans-Peter Lenhof for being my second supervisor.

All my friends constantly supported me during the whole period of my thesis. I thank Nicolas Alcaraz, Nasimi Eldarov, Tobias Frisch, Arthur Gollmer, Tim Kacprowski and Nora Speicher for their helpful reviews and fruitful ideas. I am very grateful to Yannic, Franzi and Kerstin, who helped me taking my thoughts off things so many times.

Finally, I would like to deeply thank my family for their enduring support in many ways, for trusting and believing in me whatever I do and for being constantly interested in the advancements of my work.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

In recent years the amount of biological data produced by large-scale experiments has grown rapidly. The increasing size of data sets requires computational analyses that search for structure and meaning in data in an automatized way. In most cases the input data is given without class labels. In the field of unsupervised learning, techniques are developed to identify structure in this unlabeled data. Clustering approaches are one popular type of such techniques and identify structure in the inputs by arranging similar entities together in groups.

To date a large number of clustering methods has been proposed. The performance of a single method varies greatly over several data sets, caused by differing characteristics of the inputs. Also each of these approaches relies on its individual idea of how to uncover groups of close elements. Therefore, applying a set of methods to the same input may yield quite different results. All methods provide at least one density parameter that fine-tunes their behavior and allows to integrate pre-existing knowledge about the data into the clustering process.

We develop an automatized framework which guides through the complete clustering process and tackles the aforementioned problems. It enables the user to apply multiple clustering methods to several inputs in an automatized way. If good density parameters are unknown a priori, the framework offers methods to perform automated parameter optimization. It provides integrated tools to analyze data inputs and clustering results either in an independent or a combined fashion. Furthermore, data generators create data sets which allow to evaluate clustering methods on inputs with specific char-

acteristics. This helps to unravel connections between properties of the inputs and the performance of clustering methods. Our framework is designed in a modular way and allows for configuration and customization of almost any of the computational stages during the clustering process. For instance, clustering methods, data inputs, parameter optimization methods, cluster validity indices or input- as well as output-formats can be provided by the user.

Within this work we 1) develop the framework structure to automatize the clustering process, 2) integrate analysis tools for input data sets with respect to statistical and graph-theoretical properties, 3) realize autonomous density parameter optimization procedures and 4) design a database together with a website to store and represent the results.

1.2 Cluster Analysis and Backgrounds

Cluster analysis is a kind of exploratory data analysis and identifies groups of similar objects, called *clusters*, in a data set. Objects contained in the *same* cluster are *similar* and objects of *different* clusters are *dissimilar*. Where the objects are vectors of measurements and are also called samples. n denotes the number of samples and p the number of features. Samples can be interpreted as points in their p-dimensional feature space.

Grouping is a very elemental task, found in many different scientific areas like machine learning, artificial intelligence or pattern recognition. Clustering has been employed in astronomy, plant taxonomy, marketing, geography, medicine, chemistry or history [33]. Although the information conveyed in data differs, the task of clustering remains universal. In this section we will discuss the relevant factors influencing cluster studies.

- The data may contain origin-specific artifacts, feature values may correspond to different scales or types. Data preprocessing and normalization reduces the influence of these factors on the clustering results. We will explain the concepts in some detail and give (counter-)examples. Data sets are also stored in many possible formats. This emphasizes the need for automated format conversions and standards.

- Part of available clustering methods require numerical coordinates as input, and part pairwise similarities between the data objects. When data with numerical coordinates are given, distance measures need to be used to assess similarities for pairs of objects. The choice of the right distance measures can influence the results significantly.

- A clustering method will, when applied to data, always return a clustering, irrespective of whether the data contained some natural grouping structure or not. Before carrying out a cluster study on a data set, it is reasonable to assess whether there is any grouping hidden in the data using cluster tendency analysis.
- As there is *no universal definition* of similarity of objects, a *manifold* of *clustering methods* exist, unraveling groupings based on their optimization criterion, often based on individual definitions of object similarity. We will describe the methods that are included in ClustEval in more detail.
- A clustering of two or three dimensional data can be inspected by eye. In high dimensional instances the qualitative assessment of a clustering becomes more difficult and reasoning, whether the clustering makes sense more subjective. Mathematically defined measures are employed, to generalize clustering judgment from low to higher dimensional data in an automated way. Based on a multitude of definitions of reasonable groupings, many cluster validity indices exist.
- Assessment of clustering behavior and performance of methods under controlled conditions with known outcomes can help to identify advantages, biases and pitfalls of clustering methods. Knowing when a clustering method behaves in what way can be of great value for the researcher, because he can decide beforehand, whether it makes sense to employ a certain method on the data at hand.

1.2.1 Data Origins

The information contained in data depends on the scientific field. Researchers in bioinformatics may measure gene expression levels, protein-protein interactions or DNA methylation levels, computer linguists may be interested in the occurrences of words, in medical informatics the fluorescence levels of tumor markers in the brain are pictured for disease diagnosis. Based on these origins data exhibits specific artifacts, which need to be kept in mind to apply clustering successfully. Here we describe data origins of data sets, which we included in ClustEval.

1.2.1.1 Gene Expression Levels

Genes of an organism are expressed in different amounts, influenced by many factors, for example tissue type [18, 22], health status and cell cycle state [36].

These information contained in the gene expression levels can be exploited by clustering methods to distinguish between healthy and abnormal cells, to identify cell lines or co-regulated genes [2], or to group cell samples based on their tissue types [52].

1.2.1.2 Protein-Protein Interactions

Proteins are molecules consisting of chains of amino acids, folding in complex ways. They may bind to each other in pairs or even complexes that consist of more than two proteins. Proteins may exhibit very dedicated functions only when they are interacting.

Protein structure prediction techniques predict protein structure from sequence. Since protein structure determines which proteins can bind together, these prediction methods can be used to uncover possible interactions.

In many cases, the protein structure is unknown and a prediction based on its sequence alone is not sufficient. An alternative way to identify potential protein-protein interactions or complexes is based on computational methods, including clustering methods. PPI data sets are usually represented in a binary format, containing a "1" for interacting protein pairs and "0" for non-interacting pairs. Clustering can be used to identify groups of proteins possessing a significantly enriched number of pairwise interactions, highlighting potential new protein complexes which can then be further investigated by researchers [34, 54].

1.2.1.3 Protein Sequence Similarities

Hand curated databases exist, which classify proteins based on sequence and structure into homologous families [43], trying to predict protein function. Since manual annotation is time-consuming and expensive, automated functional and structural annotation of proteins is desirable. Several methods exist, assessing similarities of protein sequences [44, 3]. The proteins can then be clustered based on these protein sequence similarities into taxonomy classes (e.g. families) [16].

1.2.1.4 Synthetic Data

Synthetically generated data sets can support the evaluation and validation of clustering methods in predefined scenarios. This is especially useful in areas, where data is sparse and thus predictions become more difficult and less reliable. This strategy is often used by authors of new clustering methods to proof its efficiency, but also by reviewers of clustering methods to rank sets

of such methods based on their performance on data sets ranging from easy to difficult revealable cluster structure.

Ensembles of synthetic data sets can be employed to benchmark the robustness of clustering methods with regard to noise of different types and levels.

1.2.2 Data Scales

The scale of a variable defines by which operations its values can be related to each other. *Interval* variables take only values of a certain interval, defined operators are "=","≠",">","<","+","−". The *ratio* scale is also defined over a certain interval, but can take 0 values and additionally defines the operators "∗" and "÷". Interval and ratio scales are also called *quantitative*. Most measurements are given on ratio scales and clustering methods usually require there inputs on this scale. There are the *nominal*, *ordinal*, *interval* and *ratio* scales, which are explained in more detail in [14]. We limit this work to data input of the ratio scale.

1.2.3 Data Types

The type of a variable indicates the degree of quantization [14] and can be *binary* (or dichotomous), *discrete* or *continuous*. Binary variables indicate absence or presence of a property. The continuous type is the most frequent one in biosciences, since most measurements correspond to continuous processes. We assume data inputs to be of numerical nature, where it can any of these types.

1.2.4 Data Formats

Independent of the scale and type of a variable, the information can be stored in different formats, which may influence the clustering results positively as well as negatively.

1.2.4.1 Raw Data

The measurements can be stored in a matrix X, were each row corresponds to a sample and each column represents a variable (or feature). Position X_{ij} corresponds to the measurement of the j'th feature of the i'th sample. In X the variables can be of different types and scales, therefore comparability of the values is difficult, since meta-information about the scales and types are

not included. Non-standardized features may necessitate preprocessing of the input data.

1.2.4.2 Object Proximities

Alternatively, instead of storing the raw data values, pairwise proximities (or similarities) for the samples can be assessed. Then the resulting symmetric similarity matrix S contains the similarity of samples i and j at position S_{ij} and S_{ji}. Applying clustering methods to S instead of X has the advantage, that the method does not need to deal with the problem of weighting, standardizing or considering the scales and types of the original variables. On the other hand, this approach may worse the clustering performance of some methods, when the similarity calculation is not performed accurately by for example choosing an inappropriate measure of similarity.

Distances and similarities carry the same information and usually refer to each other by a simple transformation function like

$$dist(x_i, x_j) = \max_{k,l} S_{kl} - S_{ij}$$

Distance Measures Manifold *distance measures* exist, assessing the distance of pairs of samples. Some of them are dedicated to numerical data measured in the ratio scale.

Metrics can be employed as distance measures, where a metric is a function $m : X \times X \to R$ satisfying the following conditions:

1. $m(x, y) \geq 0$
2. $m(x, y) = 0$ if and only if $x = y$
3. $m(x, y) = m(y, x)$
4. $m(x, z) \leq m(x, y) + m(y, z)$

The one and two dimensional instances of the *Minkowski Metric*, namely *Manhattan* and *Euclidean distance*, are frequently used. As discussed in [4], they meet some implicit assumptions, which require prior normalization of the data:

- Dimensions are equally weighted without normalization
- Dimensions are introduced linearly
- Dimensions are assumed to be independent

Non-metric distance measures violate at least one of the four conditions above, but may provide some flexibility when applied to noisy biological data.

Binary measures are non-metric and are limited to binary feature data. Some of them are based on contingency tables and count simultaneously occurring classes in the two sample vectors. We do not consider binary distance measures further in this work.

Advances have been made to standardize distances of variables based on the baseline distribution of the distance measure used. In [31] the baseline distributions for the Jaccard Index and Simple Matching Coefficient are provided and based on these, corrected distance measures are presented. We do not include these distance measures in ClustEval yet.

Other non-metric distance measures are based on *correlation* coefficients like the *Pearson* or *Spearman* correlation. While Pearson Correlation rates the distance of two variables based on their degree of linear relationship, Spearman Correlation generalizes this idea to any monotonic relationship.

1.2.5 Data Preprocessing

Several pitfalls exist which can obscure clustering structure of the original data and prevent clustering methods from its identification.

1.2.5.1 Feature Normalization and Weighting

Many clustering methods require proximity matrices as data input instead of the raw data. Some very prominent distance measures weight all features equally, for example Manhattan and Euclidean distance. Thus, if the data contains features of different numeric ranges, a single feature with larger numeric range will dominate the other features in the pairwise similarities. For an example see Figure 1.1. It depends on the data origin, whether this is desirable. The higher influence of the feature on the distances may correspond to its higher natural variance, but on the other hand it may be an artifact of data measurement, annotation or missing normalization. Thus, it is important to know where the data is coming from, how it was measured and to perform *normalization* of the features such that the values passed to the clustering method reflect the natural structure of the data. No universal argument exists, whether to normalize features to equal weight [40, 56]. A working and a counter example for normalization by standard variance can be found in Figure 1.2.

The most frequently encountered normalization techniques are presented below; detailed information on each method is given in [4, 40].

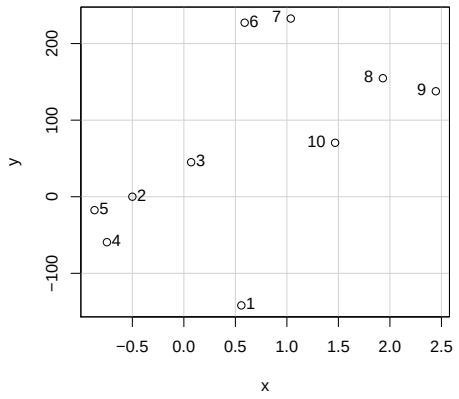


Figure 1.1: An example of a dominating dimension. Here dimension y has a larger numeric range than x and the resulting pairwise Euclidean distances will be dominated, i.e. mainly determined, by dimension y.

Mean Score: All feature variables X_i are standardized to $\tilde{X}_i$, such that $E[\tilde{X}_i] = 1$ and $E[\tilde{X}_i - \tilde{X}_j] = 0$.

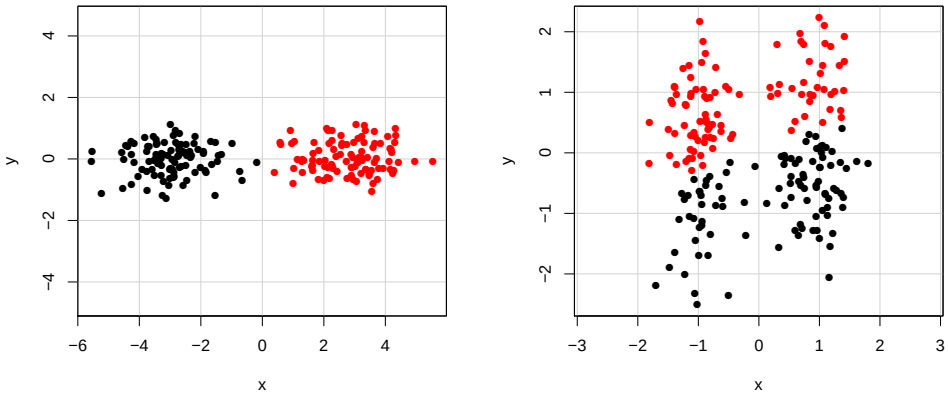
Range: The features are normalized by their range, such that $\tilde{X} = \frac{X}{max(X)-min(X)}$ and $\tilde{X} \in [0, 1]$

Variance: Feature X is normalized to $\tilde{X}$, with $\tilde{X} = \frac{X}{\sqrt{Var(X)}}$ such that $Var(\tilde{X}) = 1$

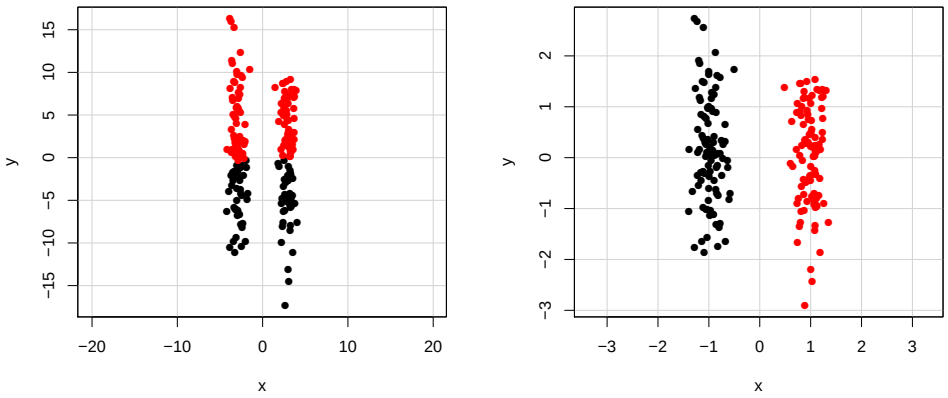
Z-score: A feature X is normalized by first subtracting its mean and then dividing by its standard variance: $\tilde{X} = \frac{X-\bar{X}}{\sqrt{Var(X)}}$

A comparison and detailed discussion of how well these normalization methods performed on validation data sets and whether or not normalization is reasonable is given in [40]. The same authors prefer the range normalization over the z-score.

Some clustering methods implicitly assign weights to the features, where the weight indicates the importance of that feature for the clustering process. The normalization techniques discussed before can be interpreted as a weighting of features, where the weight coefficients are based on some property of



(a) Normalization is disturbing the data and worsening the clustering result. Both gaussians had a standard variance of 1 in x direction, and of 0.5 in y direction. The distance between the two clusters in x direction causes the overall standard variance of dimension x to be much larger than that of y.



(b) Clustering improves after normalization by dividing by standard variation in each dimension. The gaussians had a standard variance of 0.5 in x direction and of 6 in y direction.

Figure 1.2: Standard variance normalization examples with 200 synthetically generated data points, distributed over two gaussians with 100 points each. We normalized the features by division of the standard variance. After normalization we applied K-Means with $k = 2$. The resulting clusters are denoted in black and red.

the features as for example its variance or range. This weighting can be interpreted as a transformation of the data into a new space, whose features are functions of the original features. These new features can be non-linear [4], which might lead to overfit of the clustering model and therefore being prone to noise in the data [56].

1.2.6 Cluster Tendency

Clustering is usually employed, when not much is known about the data and to get first insights into its structure. A clustering method will determine a clustering solution regardless of whether there is real natural grouping in the data or not.

Cluster tendency is a notion providing the likelihood of the given data set possessing natural grouping structure not induced by noise. Cluster tendency should be assessed before the data is actually clustered, to avoid unnecessary computational clustering tasks. Usually cluster tendency is equalized with testing the data for spatial randomness [31], while different kinds of tests for cluster tendency have been proposed.

Scan tests aim to find regions in the feature space with an statistically significant high number of sample points. In high dimensional problems, this kind of analysis is computationally infeasible.

Quadrant analysis masks out points within quadrants and compares the expected number of points within the quadrant to the found number of sample points. This approaches suffer from the same computational problems as the aforementioned one.

Second moment structure calculates the expected number of points in neighborhoods of certain sizes, against which one can compare the sample points of the data.

Interpoint distances can be used to test the observed distance distributions against an expected distance distribution of pairs of uniformly distributed points.

Nearest-neighbor distances of observed points to observed points can be calculated and compared to nearest-neighbor distances of observed points and newly uniformly sampled points. Methods based on this approach are also called *sparse sampling tests*. Depending on the null-hypothesis of randomness, different tests have been proposed, like for example based on the Hopkins [29] statistic. This test can also be employed to assess

cluster validity and is explained in more detail in the corresponding subsection 1.2.8.

1.2.7 Types of Clustering Methods

As discussed earlier, various definitions of similarity of objects exist, each implicating individual kind of groupings even for the same data. Many clustering methods have been proposed, each with its very individual idea of how to define and uncover groups and structure in data. Clustering methods can be classified based on different characteristics, for example their underlying model applied to the data or the general structure of the detected clusterings.

Models. Clustering methods are based on different underlying models which they apply to the input data, resulting in different input interpretations. *Graph theoretic* approaches model the input data as a graph and operate on the graph structure. This allows to apply and employ well defined algorithms and measures originating from graph theory. Examples are approaches based on Markov Graphs [58], Weighted Transitive Graph Projection (WTGP) [60] or Minimum Spanning Trees (MSTs) [66]. Examples of *probabilistic* clustering methods are based on prior probability estimation of Bayesian theory [27] or on permutation based clustering ensembles [6]. Clustering approaches can also be based on *mixture models*, i.e. fitting a different model to every cluster. An example is a mixture of Gaussian distributions [25].

Clustering Structure. In the literature clustering methods are mostly categorized into *hierarchical* and *partitional* approaches. Hierarchical methods build up a hierarchy for the data points, where each hierarchy level is connected to a splitting event. *Agglomerative hierarchical* approaches start with all data objects in a separate cluster and merging two clusters together on each level of the hierarchy. *Divisive hierarchical* clustering starts with a single cluster containing all data objects and on each level splitting up an existing cluster into two smaller ones. Hierarchical approaches assume a hierarchy in the data. This assumption is limiting the way those methods can cluster the data. *Partitional* clustering methods are less limited because they do not assume the data to have a hierarchical structure and can therefore detect groupings hierarchical approaches may not. On the other hand hierarchical approaches may perform superior when the data is indeed hierarchically structured.

1.2.8 Cluster Validity and Robustness

Cluster validity is the assessment of quality or appropriateness of a clustering for a given data set and is highly definition dependent. This is why there exist many cluster validity indices (or clustering quality measures) rewarding different optimization criteria.

In general cluster validity indices are being classified into two types. *Internal* cluster validity indices are solely based on the information of the data set that was also provided to the clustering algorithm. Since every clustering method optimizes a certain criterion, a specific cluster validity index can favor the results of some clustering methods over those of others, because the clustering method optimized exactly this criterion.

Probabilistic are internal indices which try to test the provided clusters of clustering methods for statistical significant.

External cluster validity indices incorporate new information about the data to judge the clustering quality. An example for such additional information are gold standards of how the data objects are classified correctly. While external cluster validity indices usually compare the clustering result to the real-world situation, internal measures focus their judgment on the distances or similarities in the data. Thus in some cases the two might not agree well, when the gold standard is not well reflected by the pairwise similarities and vice versa.

1.2.8.1 Internal Indices

In lifesciences there is often no gold standard given for the data set, which is why internal instead of external cluster validity indices have to be employed.

Some internal cluster validity indices might not be appropriate for certain types of cluster structure. When given the example spirals data set shown in Figure 2.7(e) a cluster validity index which rewards a maximized sum of pairwise distances between clusters and a minimal sum of pairwise distances within clusters (like for example the Silhouette Value) would penalize the gold standard classification. Since cluster structure can be almost arbitrary, there is no single cluster validity index which will work for every data set. A good strategy is to know the preferences of each index and to employ ensembles of indices when doing cluster studies. An extensive list of available internal cluster validity indices is provided in [39].

1.2.8.2 External Indices

Many external cluster validity indices are based on the four variables true positives (TP), true negatives (TN), false positives (FP) and false negatives (FN). For a given clustering and gold standard there are two ways of how to calculate these variables: A *pairwise* and a *mapping* approach. One can compare the relationship of pairs of objects in the gold standard and in the clustering:

TP: Element a and b are in the same class in the gold standard, and are clustered together in the same cluster.

TN: Element a and b are of different classes in the gold standard, and are clustered in different clusters.

FP: Element a and b are of different classes in the gold standard, but are clustered together in the same cluster.

FN: Element a and b are in the same class in the gold standard, but are clustered into different clusters.

The second approach first establishes a mapping between clusters of the clustering and classes in the gold standard. For each class k one selects the cluster $c(k)$, with the highest number of common elements. Then for each class one can define the four variables as follows:

TP(k): Element a is in class k and in cluster $c(k)$.

TN(k): Element a is not in class k and not in cluster $c(k)$.

FP(k): Element a is not in class k, but is in cluster $c(k)$.

FN(k): Element a is in class k, but not in cluster $c(k)$.

One can then take the average values of these to get TP,TN,FP and FN for the overall clustering.

1.2.8.3 Probabilistic Indices

Some cluster validity indices have been proposed, which are based on statistical hypothesis tests. They implicitly model and account for the noise and randomness in the data and express the quality of a clustering in terms of its likelihood to be significant.

Hopkins Statistic. The Hopkins statistic [29] is based on a random position hypothesis. Under the null hypothesis it assumes the data to be uniformly distributed without cluster structure. The null hypothesis is rejected, if cluster structure is statistically significant. Cluster structure here is defined via closest data object neighbors. If there is cluster structure present, the nearest neighbors are much closer, as if the data objects were uniformly distributed. Therefore, the null hypothesis is rejected, if the nearest neighbors of observed points to other observed points can be shown to be closer, than of observed points to uniformly generated data objects distributed over the same space. For a given clustering every cluster is statistically tested separately, and only if for no cluster the null hypothesis can be rejected, the clustering is significant.

1.3 Summary and Objectives

The relevance of cluster studies for exploratory data analysis together with its increasing complexity motivated the goals of this thesis. A cluster analysis can be summarized as a process consisting of the aspects discussed in this chapter:

- The origin of the data influences clustering results. The data input needs to be interpreted correctly to cluster it successfully. This step cannot be automatically dealt with, but needs to be kept in mind by the researcher.

- Preprocessing the data ensures comparability and standardization of the measurements and features.

- Analyzing the data set with respect to cluster tendency avoids unnecessary clustering tasks. Only if a grouping structure is present in the data clustering makes sense.

- Deciding for suited clustering methods with respective density parameters is important to achieve reasonable clusterings.

- Converting the data set to appropriate formats is necessary to apply clustering methods. If a conversion to pairwise similarities is necessary, a distance measure should be employed conserving natural groupings.

- Choosing suitable cluster validity indices to assess cluster validities for the generated clusterings.

- Analysis of clusterings may provide answers to the question, whether clustering methods produce comparable results. If they do, this adds further significance to the clusterings.

- Detecting correlations between data set properties and cluster validities of certain methods can promote the more general understanding of when and why clustering methods perform how.

From these tasks we deduced the objectives of this Master's Thesis, which we will discuss briefly in the following.

1) **Integrative data analysis.** We want to be able to assess relevant properties of data sets. The results should be presented in a table on a website either in numerical or if appropriate in visual form. The data properties of different data sets should allow the easy comparison of data sets based on these properties and its basic classification. The data analysis options should include *cluster tendency* measures.

   The framework should be extensible by new types of data analysis.

2) **Data preprocessing.** Our framework should provide infrastructure to perform automated configurable data preprocessing operations, like normalization of features. The preprocessing should be integrated into the clustering process, such that clustering methods can directly operate on the preprocessed data.

   The framework should be extensible by new data preprocessing algorithms.

3) **Format conversions.** Clustering methods support specific data formats and data sets are available in different formats. The framework should support the user in converting data sets to required formats, such that clustering methods can be applied automatically to any data set without the need to manually convert the data to the right formats. The same holds for the result formats. We want to include automatic conversion of clustering results, such that results of all clustering methods are available in the same format and are comparable. We define *standard input and result formats*. If the framework needs to parse information from either data input or clustering results it can rely on the standard formats.

   Alternative *distance measures* should be available to convert raw data to pairwise proximities.

   The framework should be extensible by new formats and distance measures.

4) **Clustering methods and density parameters.** We want our framework to provide a customizable infrastructure, to execute sets of clustering methods on sets of data sets. The *density parameter optimization* should only require little configuration and then happen automatically. The results of parameter optimizations should be easily accessible and visually represented on a website.

   The user should be able to extend the framework by new clustering methods and data sets.

5) **Cluster Validity Assessment.** Several cluster validity indices of different kinds should be available. Validities of generated clusterings should be automatically assessed by the framework, after the user has told the framework in the beginning, which indices he wants to evaluate. All achieved cluster validities should be stored in a database and visually presented on a website in tables and graphs, allowing for the easy comparative analysis of achieved validities of different clustering methods and data sets.

   The framework should be extensible by new cluster validity indices.

6) **Clustering Result Analysis.** The clusterings produced by the framework should be presented in visual and tabular form and details should be available for inspection by the user. Integrative comparison of sets of clusterings should be available, to provide a significance indication of these clusterings based on majority votes.

   Every clustering should be presented on the website in visual and tabular form.

   The framework should be extensible by new types of clustering result analysis.

7) **Relationships of data properties and cluster validities.** The framework should support the user in detecting correlations or even causalities between data properties and achieved cluster validities on the corresponding data set.

   The framework should be extensible by detection of new types of relationships.

8) **Open Source.** We want to publish all parts of our framework under the GNU General Public License, to enable the dedicated community to further contribute to the standardization process of cluster analyses.

The cluster study aspects covered by our framework are also visualized in Figure 1.3.

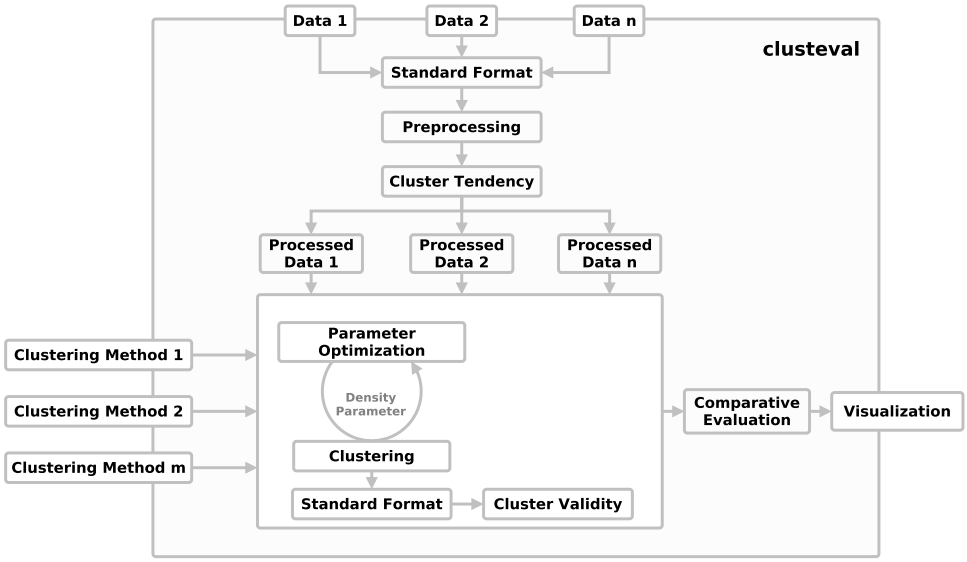


Figure 1.3: The aspects we want to provide in our framework ClustEval.

1.4 Related Work

Some related projects exist, which aim to solve at least part of our objectives. In this section we briefly discuss, which project fulfills which requirements. The descriptions do not reflect the complete functional range of each method, but rather describe the relevant part in the clustering context of this work. Afterwards, we present the results of the discussions in Table 1.1.

1.4.1 ELKI

ELKI (Environment for Developing KDD-Applications Supported by Index-Structures) [1] is an open-source data mining framework written in Java. It was published as a tool to unify several subspace clustering algorithms and allow their fair comparison for efficiency and effectiveness. Besides subspace clustering algorithms, ELKI contains few methods to perform standard clustering tasks, for example different variants of k-means and k-medoids. Different input sources are supported, like text files or databases, and it is capable of reading varying file input formats. Besides Euclidean distance other distance measures are included, e.g. Manhattan or cosine distance, which can be used to calculate pairwise similarities between input data objects. The frame-

work provides detection of outliers by several means. Basic visualizations of input data, resulting clusterings and outlier detection are integrated within ELKI. New algorithms, distance functions, parsers and visualizations can be incorporated. ELKI can generate data sets of uniform, normal and gamma distributions, with varying overlaps, rotations, translations and scalings of the clusters [1]. Its main focus is the execution and evaluation of single scenarios specifying a single method, data set and distance measure which supports the practitioner's work.

1.4.2 jClust

jClust [47] is a Java application providing means to apply several clustering methods to a data set, which follows a predefined pairwise similarity-based input format. The available clustering algorithms include for example Affinity Propagation, k-means, Markov Clustering and Spectral Clustering. Additionally several postprocessing filters allow the modification of resulting clusterings, potentially improving the produced result. Basic two dimensional visualizations for the clusterings are provided by incorporation and extension of Medusa v1.5 [28].

1.4.3 KNIME Desktop

KNIME Desktop [5] is an open-source data-mining framework written in Java. It allows to create data mining processes, including data handling, preprocessing, classification and clustering. Its focus lies on supervised rather than on unsupervised learning methods. It contains for example k-means, Hierarchical Clustering and fuzzy c-Means clustering. It allows basic parameter optimization by including loops in the process, more advanced parameter optimizations require user contribution. It provides extensive data analysis and visualizations. It is comparable to RapidMiner in functional range and structure.

1.4.4 MLcomp

MLcomp (Machine Learning comparison) [2] is a website providing standardization for machine learning tasks, to promote transparency, reproducibility and collaboration of machine learning solutions. Its main focus lies on supervised rather than unsupervised learning methods. The existing website does not provide dedicated functionality for the clustering domain. The user can

[1] `http://elki.dbs.ifi.lmu.de/wiki/DataSetGenerator`
[2] `http://mlcomp.org`

upload new programs and data sets and define runs, which comprise execution of programs on data sets. Resulting classifications are ranked based on evaluated metrics. Each run result is listed together with its evaluation metrics (e.g. different error measures) and running time. The website allows for the quick comparison of different machine learning algorithms, intended to ease the work of the researcher to apply sets of algorithms to sets of data sets. MLcomp does not provide integrated data analysis, preprocessing, clustering visualizations, clustering postprocessing or parameter optimization.

1.4.5 RapidMiner

RapidMiner (formerly YALE) [38] is an extensive open-source data mining framework written in Java, providing tools for process design and data handling. It incorporates the WEKA library including machine learning tasks in the general framework. This enables RapidMiner to generate machine learning processes, also including clusterings of data sets using different clustering methods including DBScan, EM (Expectation Maximization), Hierarchical Clustering, k-means, k-medoids, SVM (Support Vector Machines), using varying distance measures, evaluating chosen cluster validity indices and writing the results in files or databases. It also provides basic parameter optimizations to find good density parameter values. The framework can be extended by new clustering methods, distance measures and validity indices or approaches to optimize parameters. Basic data and clustering visualizations are available.

1.4.6 WEKA

WEKA [24] is an open-source data mining software, containing functionalities to preprocess, classify or cluster data sets using different classifiers and clustering methods. Its main focus lies on supervised learning methods, and clustering functionality is rather limited. Included clustering methods are for example Hierarchical Clustering and k-means. Data and calculated clusterings can be visualized using different kinds of visualization models. WEKA can load data from different source types, including files, URLs or databases, where different file formats are supported. The framework provides parameter optimizations for classifiers, but not for clustering methods. WEKA's knowledge flow editor allows to setup clustering processes, i.e. preprocess, analyze and visualize data, apply sets of clustering methods to sets of data sets, visualize clustering results. Synthetic data sets can be generated using integrated data generators.

	ELKI	jClust	KNIME Desktop	MLcomp	RapidMiner	Weka
Supports Clustering	✓	✓	✓	(✓)	✓	✓
Clustering Performance Comparison	✗	✗	✗	(✓)	✗	✗
Clustering Visualizations	✓	✓	✗	✗	✗	✓
Parameter Optimization	✗	✗	(✓)	✗	(✓)	✗
Open Results Platform	✗	✗	✗	✓	✗	✗
Data Analysis	(✓)	✗	✓	✗	✓	✓
Data Preprocessing	✓	✗	✓	✗	✓	✓
Data Generation	✓	✗	(✓)	✗	✓	✗
Extensibility	✓	✗	✓	✓	✓	✓
Multi-Threaded	✗	✗	(✓)	✓	✗	✗
Open Source	✓	✓	✓	✓	✓	✓

Table 1.1: Requirement analysis of related projects. Checks are put in brackets if the corresponding functionality is not provided by default but has to be implemented or added by the user.

1.5 Requirement Analysis and Overview

This chapter motivated this work, introduced into the backgrounds of cluster analysis and lay the groundwork for later chapters. Based on these fundamentals we made design decisions later.

We compared several related projects and assessed, whether they fulfill our objectives. Although all projects are intended for the data mining domain, most are specifically focusing on supervised learning tasks (KNIME, MLcomp, RapidMiner and WEKA) and neglect the field of unsupervised learning (i.e. clustering). Those frameworks provide basic functionality needed in clustering tasks, like data parsing, automatized execution of methods on data sets, data preprocessing (filtering, normalization) and visualization, but few features specific to clustering are available, like for example cluster visualization, postprocessing or comprehensive clustering performance comparison of several methods. Also each project provides only few clustering methods and cluster validity indices. Table 1.1 summarizes our analysis.

This work is organized in five chapters, here we briefly discuss their contents.

Chapter 1: This first chapter contains the relevant backgrounds of cluster analysis and the resulting existing problems. We formulate our main objectives and with respect to those we analyze related projects. We summarize the results of the requirement analysis in a table.

Chapter 2: In the second chapter we present the design, structural details and integrated components of ClustEval. For each type, for example distance measures or cluster validity indices, we list the components ClustEval provides by default.

Chapter 3: The implementation chapter briefly describes technical details of our framework.

Chapter 4: Here we demonstrate the power of ClustEval by illustrating three application cases. For each case we provide all necessary details for reproduction of the cluster study results. Additionally the results including configurations and inputs can be found online at the ClustEval website [3].

Chapter 5: We conclude our work in the fifth chapter, provide an overview of features being still work in progress and presenting possible future work extensions.

[3] `http://clusteval.mmci.uni-saarland.de`

Chapter 2

Materials and Methods

2.1 The Framework Structure

This section describes the general structure of the framework. Here the types of all components used by ClustEval are described. Lists of available components of each type, for example distance measures or cluster validity indices, are given in the subsequent sections.

ClustEval consists of a backend and a frontend. The backend performs analyses and clustering tasks and the frontend visualizes the results that are stored in a database (see Figure 2.1). The backend is further separated into server and client, where the server component awaits commands from connecting clients. The clients can tell the server to shut down, or to start and stop tasks. The frontend is optional and can be activated or deactivated by the user. Without frontend, visualizations are unavailable but calculations and analyses can still be performed and are stored locally in the file system. Later activation of the frontend can still make use of previously calculated results.

2.1.1 Supervisor

As is visualized in Figure 2.2, the most important component of the backend server is the *supervisor*. The supervisor manages all processes within the server, can terminate them, or can start new processes. If a process terminates unexpectedly, the supervisor will restart the process.

2.1.2 Repository and Finders

The framework uses a central location, called *repository*, on the file system with a specific folder structure to store data inputs, configuration files, clus-

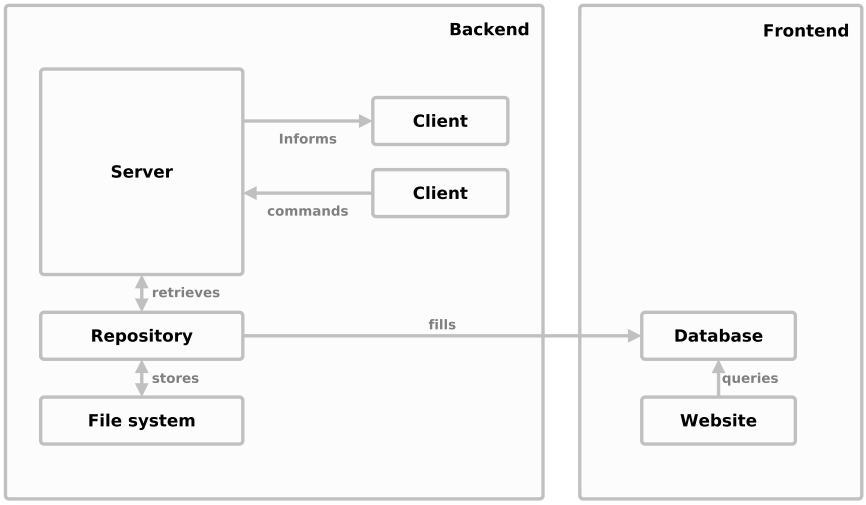


Figure 2.1: The backend and frontend separation of ClustEval allows both parts to be executed completely independent. The backend can execute runs even though no frontend is used and since relevant information are stored in the database, they can be visualized without a running backend.

tering methods and dynamically loadable modules like for example distance measures or cluster validity indices. These modules are encapsulated in jar files and are loaded by ClustEval during runtime. Dedicated processes, called *finders*, check the repository for new files or changes and update the status in memory and the database. For every parsed component in the repository meta-information are stored in an in-memory data structure.

2.1.3 Runs and Run Scheduler

The central task-concept of ClustEval are *runs*. There are execution runs and analysis runs. Execution runs are actually applying clustering methods to data sets and produce clustering results. Analysis runs can either analyze data sets, clustering results or both together.

Runs can be performed, resumed or terminated by the *run scheduler*. When a run is performed, a unique identification string is created to identify the corresponding results. During its execution a run can be terminated and using its unique ID it can be resumed later. All used input and configuration files are backed up to the results directory of the run. This way complete reproducibility is ensured. The run scheduler keeps track of the

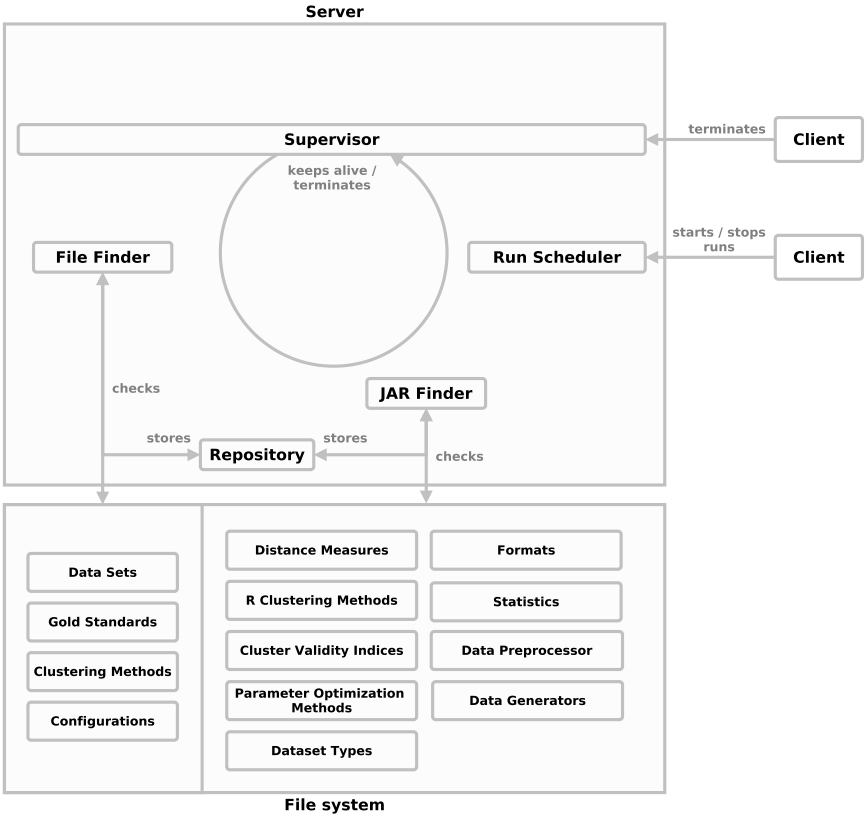


Figure 2.2: The supervisor controls all processes. Finder processes check the file system for new components or changes, and meta-information are stored in the repository in memory.

available processors and manages the runs appropriately in a queue. When a run is performed, several *run runnables* are created which are smaller parts of the overall process and are performed independently.

2.1.4 Run States

When the framework is started every run is in the *inactive* state. A run can be scheduled by a user. It is then in the *scheduled* state. As soon as there are enough resources available the scheduler dequeues the run and performs it in its own process. The run is then in the *running* state. After completion of the run, it is in the *finished* state. If the user stops the execution of the run

before it terminates, the run is in the *terminated* state. Clients can query the status of a run while it is running. The run will return its finished percentage.

2.1.5 Run Results

Every run produces results which are stored in dedicated directories in the repository. Every time a run is performed, a unique identification string is created which determines the directory name in which backups of all inputs and configurations as well as the generated results are stored.

2.1.6 Run Types

There are different types of runs. Execution runs perform clustering methods on data sets, namely clustering and parameter optimization runs. Analysis runs assess properties of either data sets, clusterings or both together. The types are briefly described in the following sections.

2.1.6.1 Clustering Runs

Clustering runs perform a set of clustering methods on a set of data inputs. For each pair only one pre-configured parameter set is evaluated and thus only one clustering is generated per pair. Before the data inputs are clustered, they are first converted to the standard input format and preprocessed. Afterwards clustering results are converted to a standard output format and then clustering qualities are assessed using a set of cluster validity indices. The whole process is conceptually comparable to parameter optimization runs visualized in Figure 2.3.

2.1.6.2 Parameter Optimization Runs

Parameter optimization runs are similar to clustering runs, except that for every pair of clustering method and data input several parameter sets are evaluated and for each a single clustering is generated. A parameter optimization run process is visualized in Figure 2.3.

Which parameter sets are evaluated in each iteration is determined by parameter optimization methods. Some methods base their current decision on the optimal parameter sets of previous iterations. These methods will restrict the search space to promising regions, which can accelerate determination of good parameters, but can also lead to local minima. Different parameter optimization methods exist and are available by default and new ones can be added dynamically.

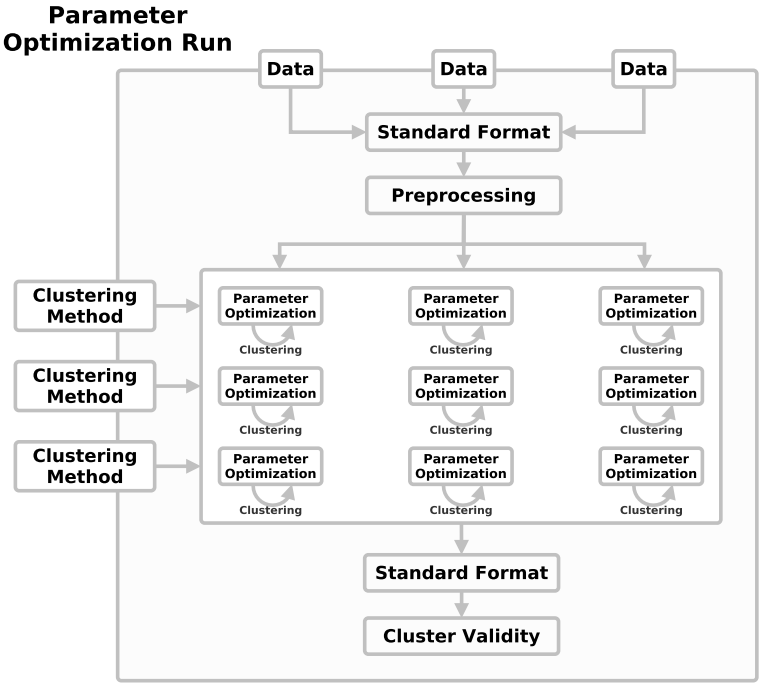


Figure 2.3: Parameter optimization runs evaluate several parameter sets and produce many clusterings for every pair of clustering method and data set.

2.1.6.3 Internal Parameter Optimization Runs

Some clustering methods provide an integrated parameter optimization, such that they automatically assess optimal density parameters. These methods then produce sets of clusterings during one execution. This is reflected by internal parameter optimization runs, which run the clustering method once and handle the iterative optimization output of the clustering method.

2.1.6.4 Data Analysis Run

A data analysis run assesses a set of data statistics for every data input. These inputs can either be data sets alone, or data sets together with gold standards. Some data statistics require a gold standard, such that those can only be employed if a gold standard is available. A data analysis run is shown in Figure 2.4 .

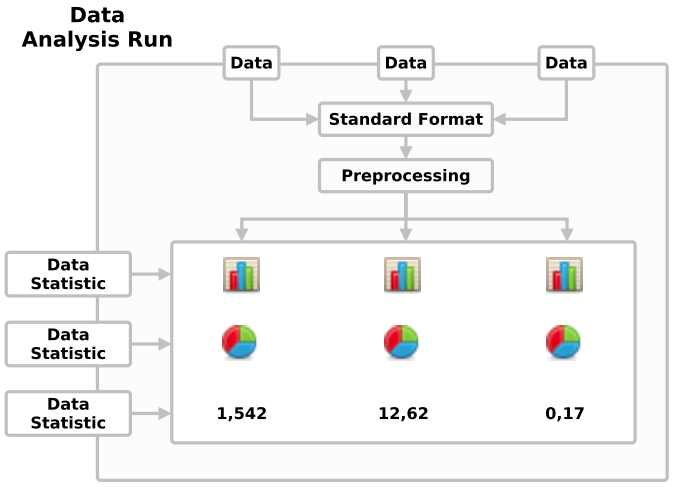


Figure 2.4: A data analysis run assesses sets of data statistics for each data input.

2.1.6.5 Run Analysis Run

This type of run analyzes the results of a set of clustering and parameter optimization runs. These results comprise a set of clusterings, which can be analyzed together. Run statistics are specific techniques to analyze these clusterings.

2.1.6.6 Run-Data Analysis Run

A run-data analysis run deduces correlations or relationships between data statistics (data analysis run results) and cluster validities (run analysis run results). Although this functionality is readily available, we do not present it in this thesis. It is part of future work projects.

2.1.7 Formats and Standards

Most clustering methods have specific supported input formats, thus input data needs to be converted to any of these formats first. Data inputs have their specific format too, thus the maximal number of required format converters to convert m data sets to the input formats of n clustering methods is $n \cdot m$.

This number can be reduced by introducing a *standard input format*. Then data formats are converted to the standard format first and in a second step to

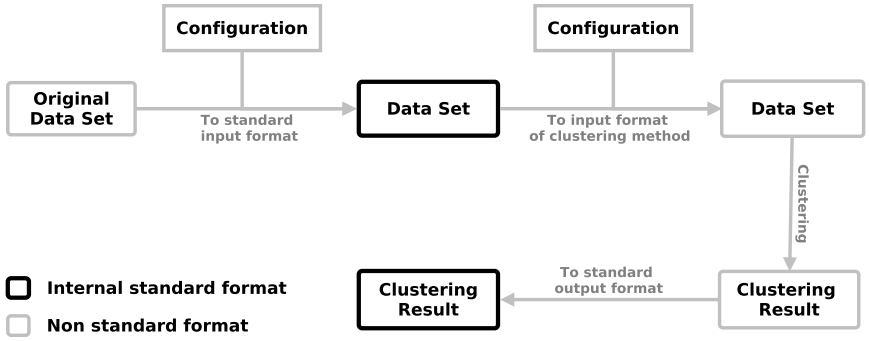


Figure 2.5: The format conversion flow within ClustEval.

the input format of the methods. This reduces the maximal number of required format converters to $n + m$, n converters converting from data formats to the standard format and additional m to convert from the standard format to the clustering method input formats.

The standard input format has a second advantage. All operations which work on the input data can rely on the functionality implemented for the standard format and do not need to do additional conversions or implementations because the input format is new. However, functionality requiring the original data instead of the standard format can still access the original data inputs.

Clustering methods output their results in individual formats as well, such that automatized analysis would require one implementation for each format. Therefore we introduced a standard result format, defining how clusterings together with the parameter sets are stored. Then all later operations based on the clustering results can rely on the standard format and no further conversions need to be done.

In Figure 2.5 the data flow within ClustEval is demonstrated, beginning with the original data input and ending with the clustering results in standard format.

2.1.7.1 Standard Input Format

The standard input format is a tab-separated similarity matrix including ids. It must neither be symmetric nor normalized between 0.0 and 1.0. Missing entries can be indicated by the string *NA*.

	id_1	id_2	...	id_n
id_1	NA	-0.4	...	5.3
id_2	0.2	7.0	...	NA
...	...	...	...	...
id_n	4.1	4.9	...	7.0

2.1.7.2 Standard Result Format

In the standard result format the clustering is stored together with the parameter set used to generate the clustering. The clustering is a string, in which clusters are separated by semi-colons and objects within the same cluster are comma separated. Each object identifier is followed by a colon and the fuzzy coefficient of how strongly this object belongs to the cluster.

p_x is the name of the x'th density parameter to optimize, v_{yz} is the value of density parameter p_z in iteration y, $|C_w|$ is the number of clusters of clustering of iteration w and c_{uv} is the v'th cluster of clustering C_u:

$\mathrm{p}_1, \mathrm{p}_2, ..., \mathrm{p}_k$	Clustering
$\mathrm{v}_{11}, \mathrm{v}_{12}, ..., \mathrm{v}_{1k}$	$\mathrm{c}_{11}; \ldots; \mathrm{c}_{1\lvert C_1\rvert}$
...	...
$\mathrm{v}_{i1}, \mathrm{v}_{i2}, ..., \mathrm{v}_{ik}$	$\mathrm{c}_{i1}; \ldots; \mathrm{c}_{i\lvert C_i\rvert}$

The format of cluster c_{uv} is defined as

$$c_{uv} = o_1 : f_1, \ldots, o_{|c_{uv}|} : f_{|c_{uv}|}$$

where o_i is the i'th object of the cluster c_{uv} and f_i is the fuzzy coefficient of object o_i and cluster c_{uv}

2.1.8 Configurations

ClustEval requires some components and inputs to be configured by the user. This can be provided in configuration files, specifying options and parameters of *data sets*, *gold standards* and *clustering methods*. These components can only be used within ClustEval if appropriate configuration files have been added. A configuration file will only be successfully parsed and will be available to be used, if all referenced components are available. For example, in a clustering method configuration its supported input formats and run result format has to be specified. If any of those formats are unknown to the backend, the configuration file will not be loaded.

To provide some flexibility in combining different data sets with different gold standards and refer to them during runs, data inputs are further

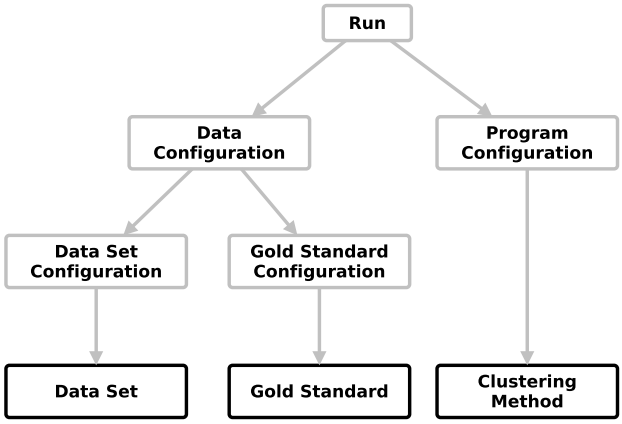


Figure 2.6: The configuration file hierarchy demonstrating how runs access information during execution.

abstracted within a hierarchy of configurations. This is visualized in Figure 2.6. A run does only directly access data configurations and program configurations. Via the data configuration the run can indirectly access data set configuration and gold standard configuration and via these data set and gold standard. Each of the configurations adds some settings to the referenced object, which is used throughout the process. For example if the data set contains numerical coordinates, the data set configurations contains information, which distance measure to use to convert the coordinates to pairwise similarities.

2.1.9 R Framework Integration

ClustEval supports communication with an Rserve server. Such a server provides the R functionality to remote machines via TCP/IP connections. In general, R can be used anywhere in ClustEval, while infrastructure is already provided for the usage of *clustering methods*, *distance measures*, *cluster validity indices* and *data generators* that are readily implemented in R.

2.1.10 Framework Extensibility

Many components can be dynamically added to and removed from a running backend. Since the backend server employs finders to check the file system for changes in regular intervals, new data sets, clustering methods, distance

Type	Component
Raw file	Clustering methods
	Data sets
	Gold standards
	Configuration files
JAR archive	R clustering methods
	Cluster validity indices
	Distance measures
	Data set types
	Formats
	Parameter optimization methods
	Data set preprocessor
	Statistics
	Data generators

Table 2.1: A table of all types of components, which can be added by the user. There are two types of extensions. Either a component can be added by (a) implementing java classes and placing them as a *jar* file in the repository or (b) placing a *file* directly in the repository.

measures and other components will be recognized and integrated into the framework automatically.

All possible extensions are shown in Figure 2.2 and in Table 2.1. For more detailed information about how to extend ClustEval please consult the technical documentation available on the ClustEval website.

2.2 Data Sets and Gold Standards

In the following we will describe the data sets and corresponding gold standards contained by default in ClustEval. A brief comparison can be found in Table 2.2.

2.2.1 Amidohydrolase Protein Sequence Similarities

The data set contains pairwise similarities of blasted sequences of 232 proteins belonging to the amidohydrolase superfamily. A gold standard is provided by [9] describing families within the given superfamily. According to the gold standard the amidrohydrolase superfamily contains 29 families.

Type	Name	Size	GS	Source
Gene expression	Golub [23]	n=38, p=999	Yes	URL
PPI	Brohee [8, 46]	n=1562	Yes	URL
Protein similarity	Brown [9]	n=232	Yes	URL
	Wittkop [61]	n=506	Yes	URL
Social network	Zachary [65]	n=34	Yes	URL
Synthetic	cassini250	n=250, p=2	Yes	
	cuboid250	n=250, p=3	Yes	
	spirals250	n=250, p=2	Yes	

Table 2.2: A list of all data sets contained in ClustEval by default. n is the number of samples and p the dimensionality. GS is the abbreviation for gold standard.

2.2.2 ASTRAL Protein Similarities

ASTRAL [7] is a database containing classifications of proteins partially derived from the SCOP database. Different releases of ASTRAL are available, containing data evolved over time. We included the data set ASTRAL95_1_161 derived from ASTRAL as described in [61]. This data set contains pairwise protein similarities generated by applying a similarity function to the blasted proteins contained in the above protein sets. The gold standard to the data set contains the SCOP classification of each protein.

2.2.3 Bone Marrow Gene Expression

This data set [23] has been made publicly available by the Broad Institute. It contains microarray gene expression levels of 999 genes for 38 samples of leukemia patients suffering from three different subtypes of acute leukemia. A gold standard is provided consisting of three cancer subtypes.

2.2.4 Protein-Protein Interactions MIPS

The MIPS Mammalian Protein-Protein Database is a database for protein-protein interactions of mammalian species [46]. We used the data set proposed in [8] consisting of a subset of 220 protein complexes of 1562 proteins.

2.2.5 Zachary Karate Club

The data set [65] contains similarities between 34 members of a karate club. The karate club split up into two groups, which is used as a gold standard.

Name (Abbreviation)	Parameter	Version	Availability
Affinity Propagation (AP)	*preference*	16/02/07	URL [20]
Hierarchical Clustering (HC)	k	2.15.1	R, *stats*
K-Means (KM)	k	2.15.1	R, *stats* [25]
Markov Clustering (MC)	inflation I	12-068	URL [13]
Partitioning Around Medoids (PAM)	k	1.14.2	R, *cluster* [33]
Spectral Clustering (SC)	k	0.9.14	R, *kernlab*
Transitivity Clustering (TC)	threshold T	1.0	URL [62]

Table 2.3: List of Clustering Methods. The k parameters correspond to the number of clusters. The R package is denoted in italic.

2.2.6 Synthetic data sets

We included three synthetically generated data sets, well suited for validation purposes. All three have 250 generated data points, where cuboid250 is three, cassini250 and spirals250 are two dimensional. For all data sets a gold standard was generated as well.

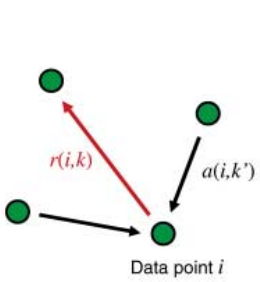
2.3 Clustering Methods

ClustEval contains a set of clustering methods by default, which are frequently used in the bioscience literature. We describe each method with respect to the criteria discussed in subsection 1.2.7 and sketch the main idea of how it uncovers groups of similar objects. If available, we will reference to literature, which demonstrates the appropriateness of the corresponding method for data of different origins. All available methods are also shown in Table 2.3 together with the abbreviations we will use throughout this thesis.

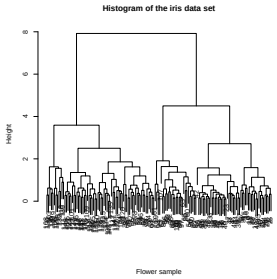
2.3.0.1 Affinity Propagation

Affinity propagation [20] is a *partitional* clustering method based on a graph model. Data points are represented as nodes and are connected with directed edges. The goal of affinity propagation is to detect representatives together with their corresponding clusters by sending flow along the edges between the nodes. The flow that is sent from node n_i to n_j depends on two factors.

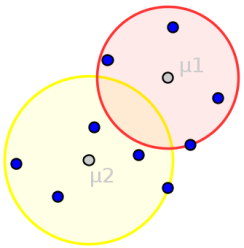
The *responsibility* $r(i, j)$ is sent from every point to potential representatives and is the likelihood of n_j being a good representative for node n_i, compared to other competing representatives. The *availability* $a(i, k)$ is sent



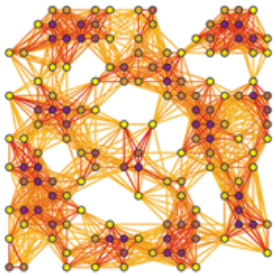
(a) The message sending process of affinity propagation. Image taken from [20].



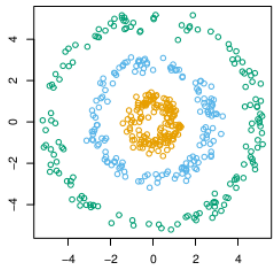
(b) A histogram of the iris data set [17].



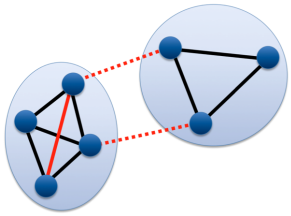
(c) K-means with $k = 2$. Sample points are assigned to their closest cluster mean.



(d) A Markov graph resulting from alternating expansion and inflation. Darker edges are stronger correspondence between two nodes.



(e) The non-convex spirals data set which can be clustered by spectral clustering. Image taken from [56].



(f) Illustration of edge operations employed by transitivity clustering. Image taken from `http://transclust.mmci.uni-saarland.de`.

Figure 2.7: Illustrations of the clustering methods integrated into ClustEval by default.

from a potential representative n_k to node n_i and is the likelihood of n_k containing n_i in its cluster, compared to competing other points.

Affinity propagation can be fine-tuned using the *preference* density parameter. It corresponds to the overall probability, of a node being a representative. Thus choosing a high preference leads to many clusters.

The authors of affinity propagation applied their method to cluster (1) images of faces to find groups of images of the same person and their best representative, (2) genes based on microarray gene expression levels, (3) sentences of a manuscript to detect the most central sentences, (4) cities to identify those ones efficiently accessed by airline travels [20].

2.3.0.2 Hierarchical Clustering

Hierarchical clustering approaches correspond to a group of clustering methods. Depending on whether they build up their dendrogram bottom-up or top-down, they are called agglomerative or divisive. In agglomerative clustering on each level of the hierarchy the two closest clusters are merged together. The definition of "closest clusters" is based on the chosen linkage function, *complete-*, *average-* or *single-linkage*. Divisive approaches split up the cluster with the largest diameter [33] or for which two subgroups exist maximizing the between-group dissimilarity [56].

Hierarchical clustering is being used very frequently and is one of the most used clustering methods overall, since the resulting dendrograms are intuitive and easy to interpret. The method assumes, that the data is structured in an hierarchical way, which is not always the case. On some data sets this may lead to suboptimal results, since hierarchical clustering will lock itself in local solutions which cannot be reversed in later steps (see [42]; as referenced by [41]).

A hierarchical clustering produces a dendrogram describing clusterings of all possible number of clusters from 1 to $n - 1$. The density parameter of hierarchical clustering can be seen as the number of clusters k corresponding to the level of the hierarchy, at which to cut it.

2.3.0.3 K-means

K-means [25] is a *partitional* clustering method which is based on a mixture model. Every cluster is implicitly assumed to be a gaussian distributed point cloud. The k-means algorithm is an iterative procedure keeping a set of k cluster representatives (means). All points are assigned to the cluster of the closest mean and means are reevaluated based on the new cluster-assignment.

The density parameter of k-means is the number of clusters k, corresponding to the number of means to maintain during the iterative procedure.

The clusters are assumed to be gaussians and the points are assigned to the cluster corresponding to the mean closest to the point (measured using Euclidean distance). This assumption is not appropriate for non-convex data sets, like for example the spirals data set shown in Figure 2.7(e).

2.3.0.4 Partitioning Around Medoids

K-medoid methods are a *partitional* approach similar to k-means, in that they maintain k cluster centers which are iteratively refined. Here the cluster centers are the medoids of the cluster objects. K-medoid methods can also be

applied to pairwise proximities and do not need raw numerical coordinates to calculate the medoids.

Partitioning Around Medoids [33] is a heuristic solving the k-medoids problem, thus partitioning the data objects into k partitions centered around medoids.

2.3.0.5 Markov Clustering

Markov Clustering [13] is a *partitional* and graph-model based approach. Data points are represented as nodes in a graph, with transitional edges connecting them. The edges are weighted with the pairwise similarities of the objects. From this graph a stochastic matrix is calculated, i.e. the similarities are converted to probabilities, such that all outgoing probabilities of any single node sums to one.

Then two alternating opposing operations called *expansion* and *inflation* are applied to the stochastic matrix. Expansion is a matrix multiplication operation and leads to an evenly spread of the edge probabilities, while inflation takes the element-wise-power of the matrix, which focuses the probabilities to favored edges and thins out edges between clusters.

The density parameter of Markov Clustering is the inflation parameter I which controls the power to which the elements of the stochastic matrix are raised. The larger I, the more clusters will be found.

Markov clustering has been applied to protein sequence similarities to detect protein families [16] and gene expression levels [57] by its authors.

2.3.0.6 Spectral Clustering

Spectral clustering is a class of *partitional* approaches and a generalization of traditional clustering methods in that they transform the input data based on the eigenvalue spectrum of a similarity matrix. In contrast to other approaches the similarity matrix here reflects local neighbor similarities and does not contain global similarities. They are calculated using a *kernel* function, a frequently used one is the gaussian-like radial basis kernel function.

This approach enables Spectral Clustering to cluster non-convex data, like in application case III. Many clustering methods as for example k-means have sever problems with non-convex data sets. This is one of the major advantages of spectral clustering over other more traditional approaches, since clusters in real-world data do not necessarily have to be convex.

Although Spectral Clustering methods can be performed on an already existing similarity matrix, they may then not perform as good on non-convex cluster data sets, since distance measure may obscure the grouping structure.

The density parameter of Spectral Clustering is the number of clusters k.

2.3.0.7 Transitivity Clustering

Transitivity Clustering is a *partitional* approach based on a *graph-model*. The given pairwise similarities are converted to a similarity graph. The threshold T is used to determine, which edges are removed from the graph, by subtracting T from the similarities and removing all edges with negative values. This possibly intransitive graph is then transformed to a transitive one by adding and removing edges. The idea is to find a solution where the sum of weights of added or removed edges is minimal. The transitive components of the overall graph are the resulting clusters.

Transitivity clustering has been applied by its authors to protein sequence similarities, gene expression data and [63] and protein-protein interactions [64].

2.4 Cluster Validity Indices

As described in subsection 1.2.8 cluster validity indices are used to assess the qualities of clusterings. We integrated a set of internal and external cluster validity indices into ClustEval. These will be described in the following in some more detail.

2.4.1 Internal Indices

Internal indices are solely based on the pairwise distances of the input data and do not integrate external information. Many of them try to maximize distances of elements of different clusters and to minimize distances of elements of the same clusters.

Davies Bouldin Index. The Davies Bouldin Index [11] relates the average distance of elements of each cluster to their respective centroids to the distance of the centroids of the two clusters.

$$DB = \frac{1}{n} \cdot \sum_{c_i \in C} \max_{c_i \neq c_j} \left(\frac{\sigma_i + \sigma_j}{d(\bar{c}_i, \bar{c}_j)} \right) \tag{2.1}$$

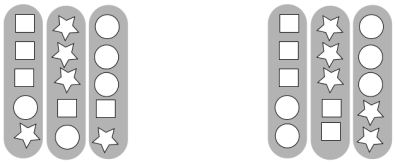
Name	Type	Formula	Range
Davies Bouldin Index	Internal	$\frac{1}{n} \cdot \sum_{c_i \in C} \max_{c_i \neq c_j} \left(\frac{\sigma_i + \sigma_j}{d(\bar{c_i}, \bar{c_j})} \right)$	[0,+∞]
Dunn Index	Internal	$\frac{\min\limits_{c_i \neq c_j \in C} \{d(c_i, c_j)\}}{\max\limits_{c_k \in C} \{d'(c_k)\}}$	[0,+∞]
F1-Score	External	$\frac{2 \cdot TP}{2 \cdot TP + FN + FP}$	[0,1]
F2-Score	External	$\frac{5 \cdot TP}{5 \cdot TP + 4 \cdot FN + FP}$	[0,1]
False Discovery Rate	External	$\frac{FP}{FP + TP}$	[0,1]
False Positive Rate	External	$\frac{FP}{FP + TN}$	[0,1]
Fowlkes-Mallows Index	External	$\sqrt{\frac{TP}{TP + FP} \cdot \frac{TP}{TP + FN}}$	[0,1]
Jaccard Index	External	$\frac{TP}{TP + FP + FN}$	[0,1]
Rand Index	External	$\frac{TP + TN}{TP + FP + FN + TN}$	[0,1]
Sensitivity (Recall)	External	$\frac{TP}{TP + FN}$	[0,1]
Specificity	External	$\frac{TN}{TN + FP}$	[0,1]
Silhouette Value	Internal	$\frac{1}{n} \sum_i s_i = \frac{1}{n} \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}$	[-1,1]
V-Measure	External	$(1 + \beta) \frac{h \cdot c}{\beta \cdot h + c}$	[0,1]

Table 2.4: List of Validity Indices

Dunn Index. The Dunn Index [15] assesses the goodness of a clustering, by measuring the maximal diameter of clusters and relating it to the minimal distance between clusters. This measure is quite conservative and prone to outliers, since it bases its calculation only on minimal and maximal distances.

$$D = \frac{\min\limits_{c_i \neq c_j \in C} \{d(c_i, c_j)\}}{\max\limits_{c_k \in C} \{d'(c_k)\}} \tag{2.2}$$

Silhouette Value. The (average) Silhouette Value [53] relates dissimilarities of elements of different clusters to dissimilarities of elements of the same cluster and tries to maximize their difference. It takes into account all pairwise dissimilarities and is therefore less conservative and less prone to outliers than the Davies Bouldin Index. The formula is given in 2.3, where $a(i)$ is the average dissimilarity of object i to other elements of its cluster, $b(i)$ is the average dissimilarity of object i to elements of other clusters, c_i is the cluster of object i and $d(i, j)$ is the



Solution A
F-Measure=0.5
V-Measure=0.14

Solution B
F-Measure=0.5
V-Measure=0.39

Figure 2.8: An example, where V-measure is more sensible to the number of different classes present in each cluster. Image taken from [51].

dissimilarity of object i and j.

$$S = \frac{1}{n}\sum_i s_i = \frac{1}{n}\frac{b(i) - a(i)}{\max\{a(i), b(i)\}} \tag{2.3}$$

$$a(i) = \frac{1}{|c_i|}\sum_{j \in c_i} d(i,j) \tag{2.4}$$

$$b(i) = \frac{1}{n - |c_i|}\sum_{j \in C \setminus c_i} d(i,j) \tag{2.5}$$

V-Measure. The V-Measure is defined as the harmonic mean of homogeneity h and completeness c of the clustering, as defined in [51]. Homogeneity h is maximized when each cluster contains elements of as few different classes as possible. Completeness c aims to put all elements of each class in single clusters. Both these measures can be expressed in terms of the mutual information and entropy measures originating from the field of information retrieval.

$$V_\beta \quad = \quad (1+\beta)\frac{h \cdot c}{\beta \cdot h + c} \tag{2.6}$$

2.4.2 External Indices

Many external cluster validity indices are based on true positives (TP), true negatives (TN), false positives (FP) and false negatives (FN). As explained in subsubsection 1.2.8.2 there are two ways of obtaining numbers for these variables: a *pairwise* and a *mapping* approach. For the following indices we will indicate which approach we used in its realization in ClustEval.

F-Score. The F-Score [50] is the harmonic mean of precision and recall, where $Precision = \frac{TP}{TP+FP}$ and $Recall = \frac{TP}{TP+FN}$ and β can be used to adjust the respective influences. This measure is based on the *mapping* approach to calculate TP,TN,FP and FN.

$$F_\beta = \frac{(1+\beta^2) \cdot Precision \cdot Recall}{\beta^2 \cdot Precision + Recall} \tag{2.7}$$

$$= \frac{(1+\beta^2) \cdot TP}{(1+\beta^2) \cdot TP + \beta^2 \cdot FN + FP} \tag{2.8}$$

False Discovery Rate. The FDR relates the number of false positives to the total number of positively predicted elements. This estimates a likelihood of an element being negative, if it is predicted positive. This measure is based on the *pairwise* approach to calculate TP,TN,FP and FN.

$$FDR = \frac{FP}{FP+TP} \tag{2.9}$$

False Positive Rate. The FPR relates the number of false positives to the total number of negative elements. This estimates a likelihood of an element being predicted positive, if it is negative. This measure is based on the *pairwise* approach to calculate TP,TN,FP and FN.

$$FPR = \frac{FP}{FP+TN} \tag{2.10}$$

Fowlkes-Mallows Index. The Fowlkes-Mallows Index [19] is the geometric mean of precision and recall (see F-Score). This measure is based on the *pairwise* approach to calculate TP,TN,FP and FN.

$$FM = \sqrt{\frac{TP}{TP+FP} \cdot \frac{TP}{TP+FN}} \tag{2.11}$$

Jaccard Index. The Jaccard Index [30] neglects the true negatives (TN) and relates the true positives to the number of pairs that either belong to the same class or are in the same cluster. This measure estimates a likelihood of an element being positive, if it is not correctly classified a negative element. This measure is based on the *pairwise* approach to calculate TP,TN,FP and FN.

$$J = \frac{TP}{TP+FP+FN} \tag{2.12}$$

Rand Index. The Rand Index [49] is the ratio of pairs of objects correctly clustered out of all possible pairs. This measure estimates the likelihood of an element being correctly classified. This measure is based on the *pairwise* approach to calculate TP,TN,FP and FN.

$$R = \frac{TP + TN}{TP + FP + FN + TN} \tag{2.13}$$

Sensitivity. The Sensitivity relates the number of true positives to the total number of true elements. This estimates a likelihood of an element being classified as positive, if it is a positive element. This measure is based on the *pairwise* approach to calculate TP,TN,FP and FN.

$$Sensitivity = \frac{TP}{TP + FN} \tag{2.14}$$

Specificity. The Specificity relates the number of true negatives to the total number of false elements. This estimates a likelihood of an element being negative, if it is indeed negative. This measure is based on the *pairwise* approach to calculate TP,TN,FP and FN.

$$Specificity = \frac{TN}{TN + FP} \tag{2.15}$$

2.5 Parameter Optimization Methods

Parameter optimization methods determine the density parameters that are evaluated during a parameter optimization. Density parameters are optimized iteratively and in each iteration another density parameter is evaluated. For some methods future density parameters depend on the cluster validities achieved with previous parameters.

Diverging parameter optimization methods account for the fact, that some clustering methods might not terminate. In this iteration no clustering is generated and no cluster validities are assessed. The process then continuous with the next iteration.

The most important parameter optimization methods are described below.

Divisive Parameter Optimization Method. This method takes the range of every parameter, divides it into uniform steps and assesses every combination of these parameter values. The result of previous iterations does not change the behavior in later iterations.

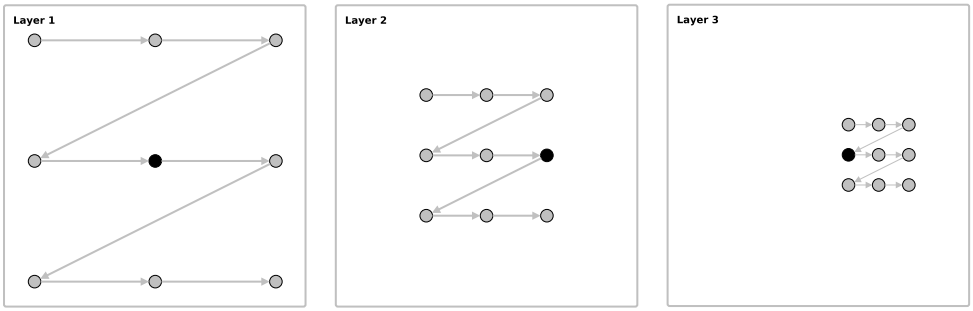


Figure 2.9: A layered divisive parameter optimization with a total number of 27 iterations distributed over three layers. Filled circles are parameter sets corresponding to the clustering with optimal validities.

Gap Statistic Parameter Optimization. The gap statistic [56] is a method to estimate the optimal number of clusters in a statistical way. This method can be used for every clustering method that uses a k (number of clusters) density parameter. This parameter optimization method only evaluates one parameter value for k, the one which is determined by the gap statistic as the optimal one.

Layered Divisive Parameter Optimization Method. This method differs from the previous one in that it takes a "layered" approach. The total number of iterations is distributed evenly over the layers, such that in each layer the same number of parameter sets is evaluated. Within each layer a divisive parameter optimization is performed, thus distributing parameter sets uniformly over the current parameter space. In the following layer the parameter space is halved in each dimension and centered around the best parameter set found in the last layer. This is continued until the total iteration number is reached. This approach focuses on regions, that performed better in earlier iterations. When the total number of iterations is I, then the number of layers is $L = \lfloor \sqrt{I} \rfloor$ and the number of iterations per parameter and layer is $\lfloor \frac{I}{L} \rfloor$. See Figure 2.9 for an illustration.

Similarity Distribution Quantile Method. This method works for clustering methods, which use similarity thresholds as density parameter, as for example Transitivity Clustering. It is a layered divisive method, but

Type	Name	Used by
Relative	APRowSimDataSetFormat	AP
	BLASTDataSetFormat	TC
	RowSimDataSetFormat	MCL,TC
	SimMatrixDataSetFormat	HC,KM,SC
	TransClustSimMatrixDataSetFormat	TC
Absolute	MatrixDataSetFormat	KM

Table 2.5: The available input formats and which clustering methods are using them. Relative input formats contain pairwise object similarities, absolute ones raw coordinates.

Name	Used by
APRunResultFormat	AP
MCLRunResultFormat	MC
TabSeparatedRunResultFormat	HC,KM,SC
TransClustRunResultFormat	TC

Table 2.6: The available output formats and which clustering methods are using them.

in the first iteration it uses quantiles of the similarity distribution of the input data.

2.6 Data and Clustering Formats

ClustEval supports a set of input and output formats. Converters are included, such that every clustering method in Table 2.3 can be applied to every data set listed in Table 2.2. All conversions are shown in Figure A.1. The framework has its internal standard input and output formats, because of the reasons discussed in subsection 2.1.7. An overview over all input and output formats can be found in Table 2.5 and Table 2.6. For more detailed information about each format please refer to the technical documentation of ClustEval.

APRowSimDataSetFormat. The input format of Affinity Propagation contains three tab-separated columns. The first two columns hold object identifier and the third column their pairwise similarity. Self similarities are omitted.

BLASTDataSetFormat. Data sets of this format consist of a BLAST m8 and a FASTA file.

MatrixDataSetFormat. This format contains raw numerical coordinates for every sample. The file consists of $p + 1$ columns, where the first column holds the object identifier and the remaining p columns hold the measurements of that sample. Missing values are not allowed.

RowSimDataSetFormat. This format has three tab-separated columns. The first two columns hold object identifier and the third column their pairwise similarity.

SimMatrixDataSetFormat. The standard input format has been described earlier in subsubsection 2.1.7.1.

TransClustSimMatrixDataSetFormat. This format is used by Transitivity Clustering. It contains in the first line the number of objects n, then in the following n lines the object identifier and in the remaining lines the upper half of the pairwise similarities of the n objects.

We included result formats, such that results of all integrated clustering methods can be converted to the standard output format.

APRunResultFormat. The output of Affinity Propagation contains a line for each object. Where line i consists of the cluster id object i belongs to.

MCLRunResultFormat. The output format of Markov Clustering contains a line for each cluster. Where a cluster is a tab-separated list of object ids belonging to that cluster.

TabSeparatedRunResultFormat. The standard output format has been described earlier in subsubsection 2.1.7.2.

TransClustRunResultFormat. The output of Transitivity Clustering contains three columns. The first column holds the threshold used to cluster, the second column holds the internally assessed F2-Score and the third column holds the the clustering as a semi-colon separated list of clusters, where each cluster holds a comma-separated list of objects. The output does not contain fuzzy coefficients but may contain the same object in several clusters.

Name	Type	Formula	Range
Manhattan Distance	Metric	$\sum_i \lvert x_i - y_i \rvert$	$[0, \infty]$
Euclidean Distance	Metric	$\sqrt{\sum_i \lvert x_i - y_i \rvert^2}$	$[0, \infty]$
Pearson Correlation	Non-Metric	$1 - \left\lvert \frac{cov(X,Y)}{\sigma_X \cdot \sigma_Y} \right\rvert$	$[-1, 1]$
Spearman Correlation	Non-Metric	$1 - \left\lvert \frac{\sum_i (x_i - \mu_x) \cdot (y_i - \mu_y)}{\sqrt{\sum_i (x_i - \mu_x)^2 \cdot \sum_i (y_i - \mu_y)^2}} \right\rvert$	$[-1, 1]$

Table 2.7: Metric and non-metric distance measures.

2.7 Distance Measures

In Section 1.2.4.2 we described, that raw numerical measurements contained in a data set can be converted to pairwise similarities by using distance measures. Those integrated into ClustEval will be presented in the following. A brief summary can be found in Table 2.7.

Euclidean Distance. The Euclidean distance takes the square root of the summed squares of distances between components of two vectors.

Manhattan Distance. This distance measure sums up the absolute distances between components of two vectors.

Pearson Correlation. The Pearson correlation determines for linear relationships between two variables. The correlations lie between [-1,+1], where -1 means linearly anti-correlated and +1 means linearly correlated. Taking the inverse of the absolute value can therefore be interpreted as dissimilarities.

Spearman Correlation. The Spearman correlation determines monotone relationships between two variables. The correlations lie between [-1,+1], where -1 means linearly anti-correlated and +1 means linearly correlated. Taking the inverse of the absolute value can therefore be interpreted as dissimilarities.

2.8 Data Statistics

Data statistics are techniques integrated in ClustEval, corresponding to properties of data sets which can be automatically analyzed by a data analysis run. In the following we give a list of all data statistics which are included

in ClustEval by default. Table 2.8 gives a brief overview and classifies each statistic based on its main motivation.

The graph-based statistics operate on a similarity graph built from the similarity matrix input data.

Class Size Distribution. Calculates the number of samples of the data set per class of the gold standard. This statistic can only be assessed when a gold standard is available.

Clustering Coefficient (unweighted). This graph-measure describes, the tendency of a graph to be clustered into separated groups. The original unweighted measure counts how many of those triplets of nodes connected by at least two edges, also have three edges.

Clustering Coefficient (weighted). A generalization of the original clustering coefficient for weighted graphs has been proposed by Barrat et al.

Graph Adhesion. The minimal sum of edge weights of removed edges, to get a resulting graph which is not strongly connected. That means, two nodes exist, which are not connected by a path.

Graph Cohesion. The minimal number of nodes to be removed, to get a resulting graph which is not strongly connected. That means, two nodes exist, which are not connected by a path.

Graph Density. The proportion of number of existing and total possible edges.

Graph Diversity Average. The average of normalized Shannon entropies of the weights of each node.

$$D = \frac{1}{n} \sum D(i) = \frac{1}{n} \sum \frac{H(i)}{log(k[i])} \tag{2.16}$$

Graph Min-Cut. The minimal sum of edge weights of removed edges, to partition a graph into two disconnected subgraphs.

Hopkins Statistic. Assessing the cluster tendency of a data set based on nearest-neighbor distances. Creates random data and calculates the probability, that the observed points are clustered.

Intra/Inter Overlap. The overlap of the intra and the inter similarity distribution can be used as an indicated whether an easy separation of clusters is possible. The overlap can be assessed differently. This statistic considers the similarity distributions discretized into buckets. Let the intra distribution be d_{intra} and the inter distribution d_{inter}, then the overlap is defined as

$$Overlap = \frac{1}{\sum_i d_{intra}(i) + \sum d_{inter}(i)} \cdot \sum_i \min\{d_{intra}(i), d_{inter}(i)\} \tag{2.17}$$

Intra-Inter Similarity Distribution. Requires a gold standard for the data set and calculates the distributions for pairs of objects belonging to the same cluster and to different clusters respectively. This statistic is visualized on the website as seen in Figure 2.10(a).

Matrix Rank. The rank of a matrix can be described as the number of linearly independent rows or columns. A similarity matrix having a low rank can be assumed to contain a lot of redundant information.

Node Degree Distribution. Calculates the number of nodes with degree k for unweighted graphs. Since we are dealing with weighted graphs the node degrees are defined as the sums of adjacent edge weights. This statistic is visualized on the website as seen in Figure 2.10(b).

Number of Samples. The number of samples alone can affect the validities of some clustering methods and can also bias some cluster validity indices.

Similarity Distribution. Calculates the distribution of similarities for pairs of objects in the data set. This statistic is visualized on the website.

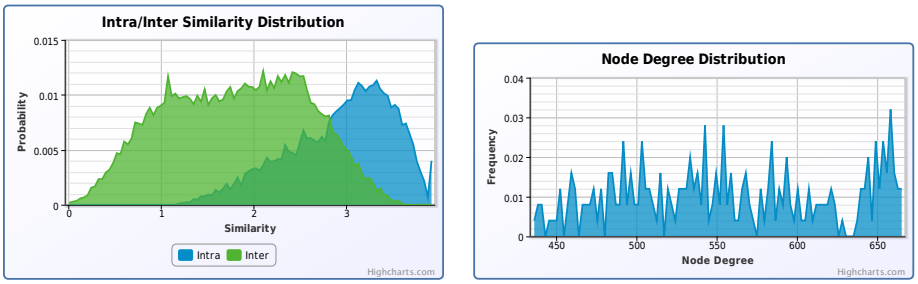
2.9 Run Statistics

Clustering results can be analyzed in a combined fashion. Run Statistics are techniques integrated into ClustEval, to analyze clustering results. Here we list all run statistics available in ClustEval by default.

Co-occurrence Matrix (best clusterings). Counts how often a pair of objects is clustered together in the best clustering of each clustering

Name	Type	Presentation
Class Size Distribution	General	Graphically
Clustering Coefficient (unweighted)	Cluster Tendency	Numerically
Clustering Coefficient (weighted)	Cluster Tendency	Numerically
Graph Adhesion	Cluster Tendency	Numerically
Graph Cohesion	Cluster Tendency	Numerically
Graph Density	General	Numerically
Graph Diversity Average	General	Numerically
Graph Min-Cut	Cluster Tendency	Numerically
Hopkins Statistic	Cluster Tendency	Numerically
Intra-Inter Overlap	Cluster Tendency	Numerically
Intra-Inter Similarity Distribution	General	Graphically
Matrix Rank	General	Numerically
Node Degree Distribution	General	Graphically
Number of Samples	General	Numerically
Similarity Distribution	General	Graphically

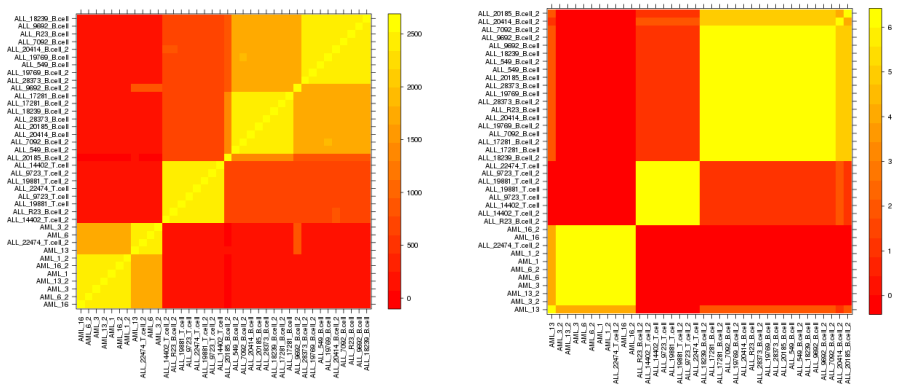
Table 2.8: The data statistics available in ClustEval by default. Some data statistics are presented numerically and other graphically on the website.



(a) Visualization of an intra vs. inter similarity distribution on the website.

(b) Visualization of a node degree distribution on the website.

Figure 2.10: Examples of data statistic visualizations on the website.



(a) Visualization of a co-occurrence matrix (all clusterings) on the website.

(b) Visualization of a co-occurrence matrix (best clusterings) on the website.

Figure 2.11: Examples of run statistic visualizations on the website.

method on a data input. These sets of clusterings originate from parameter optimizations, when the same clustering method is applied to the same data set using different density parameter values. This statistic is visualized on the website as seen in Figure 2.11(b).

Co-occurrence Matrix (all clusterings). Counts how often a pair of objects is clustered together in all clusterings of each clustering method on a data input. These sets of clusterings originate from parameter optimizations, when the same clustering method is applied to the same data set using different density parameter values. This statistic is visualized on the website as seen in Figure 2.11(a).

2.10 Run-Data Statistics

Run-data statistics are techniques integrated into ClustEval, that can be used to analyze relationships between properties of a data set and achieved cluster validities. Currently ClustEval provides three different models which can be trained on the data statistics of data inputs, trying to predict the respectively achieved cluster validities of the clustering methods. Each data statistic is a column of the input matrix X, and each row of this matrix corresponds to one data set. The best qualities assessed using a single cluster validity index

on the same set of data sets can be put into a validity vector y. For every clustering method and cluster validity index this vector contains different best achieved qualities.

Linear Regression Model. This statistic fits a linear regression model on the data statistics of a data input which can be represented as a single double value and tries to predict the achieved cluster validities of that data set. Linear regression models find a vector of coefficients β_i for a linear prediction model of the form

$$X \cdot \beta \approx y \tag{2.18}$$

solving the optimization problem

$$\min \sum_i (y_i - X_i \cdot \beta) \tag{2.19}$$

For every clustering method and cluster validity index one linear regression model is trained.

Linear Lasso Regression Model. A version of a linear regression model which tries to minimize the complexity of the trained model, by shrinking the β coefficients. Lasso is penalizing the coefficients β_i linearly, where λ can be used to control the degree of penalization:

$$\min \sum_i (y_i - X_i \cdot \beta) + \lambda \sum_i |\beta_i| \tag{2.20}$$

Linear Ridge Regression Model. Similar to the LASSO regression, but penalizing the coefficients quadratically: Tries to minimize the number of features in the model

$$\min \sum_i (y_i - X_i \cdot \beta) + \lambda \sum_i \beta_i^2 \tag{2.21}$$

2.11 Data Generators

We integrated several generators for synthetic data which are well suited for evaluation purposes of machine learning tasks in general and clustering methods in particular. Figure A.2 visualizes the data generators available in ClustEval by default.

Cassini. The cassini data set consists of three groups, two banana-shaped clusters (non-convex) bending around a circular cluster in between them.

Circles. One class forms a circle in the middle and the second class fills the cube around the circle.

Cuboid. The cuboid data set consists 3 cuboids and one cube in between in a three dimensional space. The cuboids are placed at the edges of a boundary cube.

Gaussian 2D. This data set contains a set of overlapping gaussians, where the number of gaussians and the degree of overlap can be varied.

Hypercube Corners. Gaussians are placed at the corners of a hypercube. In this data set the clusters are well separated.

Ringnorm. The data set consists of two overlapping gaussians.

Simplex. Gaussians are placed on the corners of a simplex.

Spirals. This data set contains data points distributed over two entangled spirals. This data set is a solvable but challenging clustering task, since the clusters are non-convex.

2.12 Data Preprocessors

A set of data preprocessors is available in ClustEval by default. For more detailed information about how and when to use data preprocessors be referred to subsection 1.2.5.

Range normalization. The features are normalized to a common range between [0,1].

Variance normalization. The features are normalized to a common variance of 1.

2.13 Website

The website shows information about all available objects in a repository. Several backends using different repositories can use the same website, and the repository can be changed on the website. Then only information of this particular repository are displayed.

The website consists of a set of sections, which can be selected in the menu. In the following we will briefly describe the purpose of every section. Additionally, a technical documentation is available on the ClustEval website with more detailed information.

Figure 2.12: Comparison graph on the ClustEval website over all best achieved F2-Scores of all clustering methods on all application case data sets.

Dataset	Affinity Propagation	Hierarchical Clustering	K-Means	Markov Clustering	Partitioning Around Medoids	Spectral Clustering	Transitivity Clustering
sfld/sfld_brown_et_al_amidohydrolases_protein_similarities_for_beh.txt	0.712	**0.993**	–	0.948	0.888	–	0.987
synthetic/spirals250	0.181	0.833	0.833	0.833	–	**1.0**	0.833
tcga/all_emc_spearman.txt	0.293	0.943	–	0.806	0.927	0.933	**0.984**

Figure 2.13: Comparison table on the ClustEval website over all best achieved F2-Scores of all clustering methods on all application case data sets.

2.13.1 Overview

The intention of this section is to give the user a general overview over all calculated clustering results and cluster validities of all clustering methods applied to all data sets. The comparison presented here is limited to one cluster validity index at a time, therefore a cluster validity index can be selected.

Furthermore the site contains a graph and a table (see Figure 2.12 and Figure 2.13). The graph is a bar chart showing the optimal cluster validities achieved of all clustering methods on all data sets measured with the selected cluster validity index.

The table contains the same information than the graph, but in numerical format. For each pair of clustering method and data set the optimal achieved clustering quality is shown as a number. In each row of the table the highest quality is denoted in *bold text*.

By default the table and graph focus on the comparison of different clustering methods on the same data set. Then rows in the table and groups in the graph correspond to a data set. To compare the performance of a single clustering method on different data sets, one can click the *invert button* on the top.

2.13.2 Clustering Methods

For each available clustering method a paragraph is shown. If for a certain clustering method a description including publication and link and a demonstrating image is available, these are shown here. By clicking on *details* for a certain clustering method, one sees more detailed information about that method including its performance on different data sets and lists of clusterings and cluster validities achieved using certain parameter sets. This site also provides a download link for that clustering method (if available).

2.13.3 Data Sets

Here a list of all available data sets is shown. By clicking on a certain data set one sees more detailed information about the data set including a download link, performance of different clustering methods on this data set and lists of parameter sets and cluster validities.

2.13.4 Measures

A list of cluster validity indices is shown. By clicking on an index, one is forwarded to a description page where detailed information about the measure is shown (if available).

2.13.5 Submit

This page allows visitors of a website to contribute to ClustEval. The visitor can fill out one of the available forms and an email will be sent to the site administrator. There is one form for data set submission and a second one for clustering method submission. This way, new data sets and clustering methods can be added to the framework. Then the site administrator can

apply the clustering method to the data sets of the framework or can apply existing clustering methods to the new data set.

2.13.6 Admin

The admin section allows more detailed insights into the objects stored in the repository and into the individual run results. Here all configurations are listed as well as runs and run results. By clicking on a run result, more detailed information about it are shown.

2.13.6.1 Clustering Run Results

A table containing the cluster validities for each pair of clustering method and data input is shown.

2.13.6.2 Parameter Optimization Run Results

For parameter optimization run results graphs and tables show the cluster validities achieved using different parameter sets and in different iterations (see Figure 2.14).

2.13.6.3 Data Analysis Run Results

The data statistics are shown in a table or are visualized in a graph if available for the assessed data statistics.

2.13.6.4 Run Analysis Run Results

The run statistics are shown in a table or are visualized in a graph if available for the assessed run statistics.

2.13.6.5 Run-Data Analysis Run Results

Each run-data statistic is represented textually on the website.

2.13.7 Help

The help section explains how to download and install the framework. This section is likely to be extended in the future.

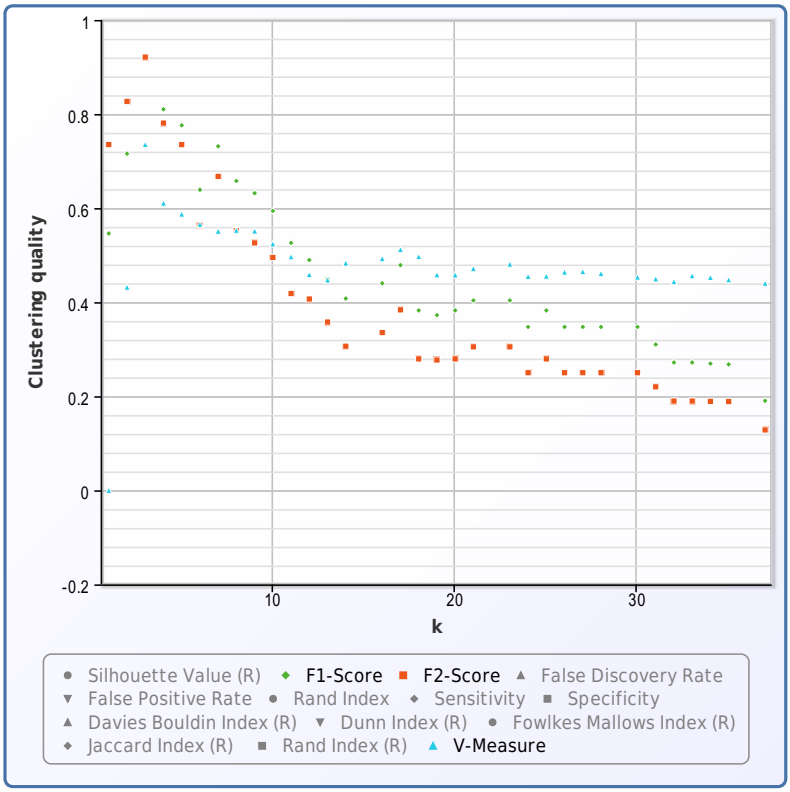


Figure 2.14: Parameter optimization run result visualizations on the ClustEval website.

Chapter 3

Implementations

In this chapter we explain some implementation specific details. However, many aspects are discussed in the technical documentation in more detail. Here we only aim to give a brief overview of the general implementation details and an understanding of how ClustEval works on a lower level.

3.1 Backend, Java and Rserve

The backend is written in Java and requires a JRE 1.6 to be executed. We used a set of libraries which we bundle with ClustEval. These are listed in Table 3.1. The backend consists of a server and client component, which correspond to two independent jar files. Java Remote Method Invocation (RMI) is used to enable inter-process communication between the server and the client.

A lot of machine learning algorithms and approaches are already implemented in the R framework. To be able to integrate this functionality into ClustEval, we use Rserve to communicate with R over TCP/IP. The usage of R is optional, but a lot of functionality of ClustEval will not be available. In Table 3.2 all R packages are listed, which are used by components of ClustEval which are included by default.

3.1.1 Installation

The backend does not need to be installed. It can be immediately used after downloading two jar files from the ClustEval website: one jar file for the backend server and the second jar file for the backend client. The server is started once and can be told an absolute path to the repository to use during runtime. After the server has been started it will scan the file-system for all

Name	Version
Apache Commons CLI	1.2
Apache Commons Configuration	1.8
Apache Commons IO	2.3
Apache Commons Lang	2.6 & 3.1
Apache Commons Logging	1.1.1
JAnsi	1.9
JLine	2.8 Snapshot
JUnit	4
JUnit Addons	1.4
log4j	1.2.16
Logback	1.0.9
Parallel Colt	0.9.4
MySQL Connector Java	5.1.21
Rserve	1.7
TransClust	1.0
Wiutils	1.0

Table 3.1: The Java libraries used and redistributed by the backend.

kinds of objects and will notify on the command line, as soon as initialization has been finished. Objects like for example data sets or clustering methods that are added during runtime will aoso be reported on the command line. Additionally, all log messages are written into a log file for later reproducibility.

Then the backend client can be started with the details of the server (at least IP-address and port number, if the default settings are not appropriate) to connect to the server and set up commands, for example start a run. More information like for example available command line parameters can be found in the technical documentation.

3.2 Frontend, Ruby on Rails and MySQL

The frontend consists of a MySQL database and a Ruby on Rails website. Ruby on Rails provides so called gems, which are small extensions providing modular functionality to a website. To realize our visualizations we made use of Highcharts and some other gems, which we list in Table 3.3. These gems will be automatically installed during the installation procedure of our Ruby on Rails application.

R Package	Used by
base [48]	Matrix Rank
cluster [37]	Silhouette Value
clv [45]	Davies-Bouldin Index
	Dunn Index
	Fowlkes-Mallows Index
	Jaccard Index
	Rand Index
fields [21]	Hopkin's Statistic
igraph [10]	Clustering Coefficient
	Graph Adhesion
	Graph Cohesion
	Graph Density
	Graph Diversity Average
	Graph Min-Cut
lars [26]	Lasso Regression
kernlab [32]	Spectral Clustering
MASS [59]	Ridge Regression
mlbench [35]	Cassini Data Generator
	Circle Data Generator
	Cuboid Data Generator
	Gaussian 2D Data Generator
	Hypercube Data Generator
	Ringnorm Data Generator
	Simplex Data Generator
	Spirals Data Generator
stats [48]	Hierarchical Clustering
	K-means
	Linear Regression
	Pearson Correlation
	Spearman Correlation

Table 3.2: The R packages used by components of the backend.

Ruby Gem	**Functionality**
authlogic	User-Management
cells	Used to realize modular components reused throughout the website
coffee-rails	CoffeeScript support in Ruby on Rails
composite_primary_keys	Adds support for database primary keys
dynamic_form	Adds helpers for error handling
execjs	Allows the execution of javascript runtime
foreigner	Adds foreign-key support to Ruby on Rails
immigrant	Allows dump of foreign-keys into migration files
jquery-rails	Use jQuery to dynamically load data via AJAX
jquery-datatables-rails	Dynamic tables with filtering support
jquery-datatables-tabletools-rails	Export of datatables
jquery-ui-rails	Provide jQuery UI to theme objects
json	Add support for the standardized data format
mathjax-rails	Parses and visualizes LaTeX code dynamically on the website.
mysql2	Add support for MySQL databases
rails	Basis of website
rails3-generators	A compatibility gem providing generators for older gems
rails_sql_view	Adds support for MySQL views
raphael-rails	Cross-browser vector graphics
sass-rails	Support for the Sass stylesheet language
simple-navigation	Used to build the menu navigation
therubyracer	A javascript engine for Ruby
uglifier	A compressor and encrypter for javascript files

Table 3.3: The Ruby gems used by the frontend.

3.2.1 Requirements and Installation

Here we briefly describe the installation requirements of the frontend. More details on how to install the frontend can be found in the technical documentation, where we also provide a step-by-step tutorial how to install ClustEval on a Debian 6 server.

To successfully install the frontend on a new machine the minimal requirements are:

1. A database, preferably MySQL 5
2. Ruby 1.9

Other databases are in general also supported by Ruby on Rails, but using another database will require adaptations in our Ruby on Rails application.

Chapter 4

Results & Discussion

In this chapter we present results achieved by using ClustEval to perform cluster studies. First, for each data set we demonstrate results of automatized data analysis and a brief interpretation of the results. Second, we perform parameter optimizations using the integrated clustering methods and compare the cluster validities and performances of the methods on different data inputs. Except the density parameters we aim to optimize, some clustering methods have important additional parameters which we list in Table 4.1 for reproducibility. Table 4.2 shows the settings used for all parameter optimizations presented in this chapter. These preferences are not varied between the three application cases.

All results presented here are also available on the ClustEval website [1] in the repository named *repositoryThesisWiwie*.

In this chapter we indicate the clustering methods by AP (Affinity Propagation), HC (Hierarchical Clustering), KM (K-Means), MCL (Markov Clustering), PAM (Partitioning Around Medoids), SC (Spectral Clustering) and TC (Transitivity Clustering).

4.1 Application Case I - Protein Taxonomy by Clustering Protein Sequence Similarities

4.1.1 Motivation

Proteins have been classified based on their sequences and structures in hand-curated databases. Since automated sequence and structure comparison approaches have been proposed, the classification of proteins can at least be

[1] `http://clusteval.mmci.uni-saarland.de`

Clustering Method	**Parameter**	**Value**
AP	maxits	2000 (2000-5000)
	convits	200 (200-500)
	dampfact	0.9 (0.7-0.99)
HC	Linkage Method	Complete
KM	iterations	10
	number starts	1
	Algorithm	Hartigan and Wong [25]
MC	–	–
PAM	–	–
SC	Kernel	Radial Basis
	Clustering Method	K-means
	Iterations K-means	200
TC	–	–

Table 4.1: Default values for additional not-optimized parameters of clustering methods. These parameters are set to the given value and are not varied during parameter optimizations, except those of AP. These parameters are changed, if AP does not converge with a *preference* parameter value. The value ranges are given in brackets behind the default value. Parameters not listed are set to their default values.

Setting	**Value**
Optimization Method	Layered Divisive
Requested Iterations	1001
Optimization Criterion	F2-Score
Evaluated Validity Indices	Davies-Bouldin Index, Dunn Index, F1-Score, F2-Score, False Discovery Rate, False Positive Rate, Fowlkes-Mallows Index, Jaccard Index, Rand Index, Sensitivity, Silhouette Value, Specificity, V-Measure

Table 4.2: The settings for the parameter optimizations in the three application cases of this thesis.

Data Statistic	Value
Intra-/Inter Overlap	0.002959893443836022
Graph Cohesion	0.0
Graph Adhesion	0.0
Graph Min-Cut	0.0
Clustering Coefficient (unweighted)	0.8238582466431129
Clustering Coefficient (weighted)	0.8732868870608063
Graph Density	0.3782467532467532
Graph Diversity Average	0.8584200869285257
Matrix Rank	231.0

Table 4.3: The data statistics for the Brown et al. data set.

semi-automatized, with an automated preliminary classification and a subsequent expert verification. We aim to verify whether clustering can be employed to autonomously classify proteins into groups based on sequence similarities for a set of proteins belonging to the common superfamily amidohydrolase.

4.1.2 Data Input

We clustered the protein sequence similarity data set by Brown et al. [9] described in subsection 2.2.1. The data set contains protein sequence similarities of 232 proteins belonging to the amidrohydrolase superfamily. Throughout this section we will refer to the data set as "Brown et al. data set".

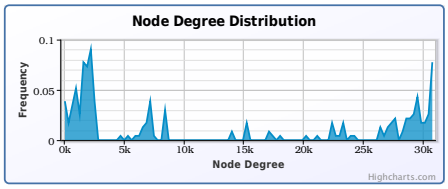
The data format are pairwise similarities and no raw numerical coordinates are available. Thus, KM can not be applied to this data set.

4.1.3 Data Analysis

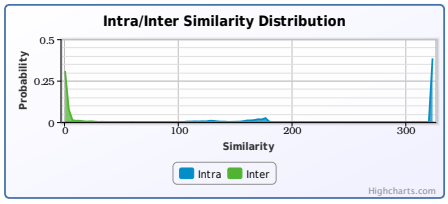
To get a feeling for the data set we analyzed it using a data analysis run. We evaluated the data statistics listed in Table 4.3. Some of them are visualized on the website as seen in Figure 4.1.

The overlap of the intra and inter similarity distribution is small with less than 1%. The intra-inter similarity distribution shows two large peaks at the minimal and maximal similarity range. Pairs of objects in the same clusters show a very high similarity, except for a rather small peak in the middle, and pairs of different clusters show a very low similarity close to 0.0.

The node degree distribution shows an interesting shape, containing two peaks representing many nodes with very low similarities to other nodes, and many nodes with very high similarities to their neighbors. This might reflect



(a) Visualization of the data statistic *Node Degree Distribution* as seen on the website.



(b) Visualization of the data statistic *Intra-vs-Inter Similarity Distribution* as seen on the website.

Figure 4.1: Visualizations of data statistics of the Brown et al. data set on the ClustEval website.

the class distribution of the gold standard, which contains 11 classes with only 1 element and the remaining objects distributed over 18 classes.

The pairwise similarity input data contained only significant BLAST hits, which explains the low graph density of 38%. Comparing the low graph density and the high unweighted graph coefficient suggests that missing edges are not uniformly distributed over all, but are focused on a few nodes. This probably reflects the protein families of superfamily amidohydrolase.

The weighted clustering coefficient shows a rather high cluster tendency of the data set, which is supported by graph cohesion, adhesion and min-Cut being 0.0, indicating that the input similarity graph is not tightly connected.

All together these analysis results indicate that the pairwise similarities of the protein sequences preserve and reflect natural grouping of the given proteins.

4.1.4 Clustering Methods

We applied AP, HC, MC, PAM, SC and TC to the data. SC was not able to transform the input similarity matrix into the eigenvector space and could therefore not cluster the data set.

Thus we employ two graph-flow methods, AP and MC, another graph-theory-based method, TC, an iterative numerical-distance based heuristic, PAM, and a hierarchical method, HC.

4.1.5 Parameter Optimizations

To give each clustering method equal chances to achieve high quality clusterings, we configured equal number of parameter optimization iterations of 1001

Clustering Method	$\mathbf{I_{eff}}$	Parameter Range
AP	992	$preference \in [0.0, 323.306]$
HC	80	$k \in [1, 232]$
MC	991	$I \in [1.1, 10.0]$
PAM	77	$k \in [1, 231]$
SC	63	$k \in [2, 232]$
TC	991	$T \in [0.0, 323.306]$

Table 4.4: The number of iterations and parameter ranges for each clustering method on the Brown et al. data set.

Clustering Method	Parameter Value(s)	F2-Score
AP	$preference = 99.957$	0.712
HC	$k = 25$	0.993
MC	$I \in [1.684, 1.717]$	0.948
PAM	$k = 10$	0.888
TC	$T \in [40.266, 40.542]$	0.987

Table 4.5: The parameter values which led to the highest F2-Scores for each clustering method on the Brown et al. data set.

for each method. To optimize the density parameters we used the *Layered Divisive Parameter Optimization Method* described in Section 2.5. The number of layers is evaluated to $L = \lfloor\sqrt{1001}\rfloor = 31$ and thus the number of values per parameter and layer is $\lfloor\frac{1001}{31}\rfloor = 32$. This makes a theoretical number of iterations per clustering method of $I_{theo} = 31 \cdot 32 = 992$. Parameter sets are not evaluated multiple times, thus the effective number of total iterations I_{eff} can be further decreased, this is especially the case for discrete density parameters. Further settings as for example the evaluated validity indices and optimization criterion can be found in Table 4.2. In Table 4.4 the number of effective iterations as well as respective parameter value ranges are shown for each clustering method.

4.1.6 Cluster Validities

The achieved cluster validities on the Brown et al. data set are represented on the website in graphical (see Figure 4.2) and tabular formats (see Figure B.3). We show the results additionally in Table B.1.

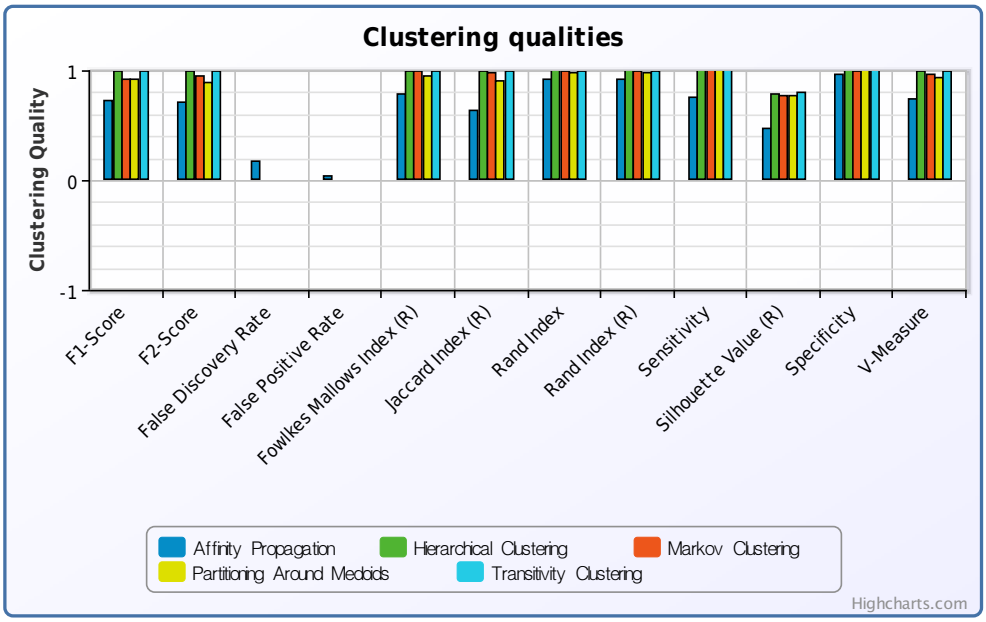


Figure 4.2: Visual representation of achieved cluster validities of the clustering methods on the Brown et al. data set as seen on the ClustEval website. Quality indices without bounded ranges are not shown in this graph, but only in the table. The artifacts in the graph legend are due to the export module of Highcharts

The ranking of the clustering methods is the same for most cluster validity indices. HC and TC constitute the best performers, followed by MC, PAM and then AP.

When looking at the optimization criterion F2-Score this order is confirmed. The website visualizes the parameter optimization in two ways, the achieved cluster validities in certain iterations or using parameter values (see Figure 4.3). These visualizations can been used to confine the region of better parameter values via eye-inspection. The parameter values which lead to the highest F2-Scores are listed in Table 4.5.

We use the website to inspect the best clusterings of the three best performers HC, TC and MC. The clusterings of HC and TC both contain 25 clusters, while that of MC contains 16 clusters. As an example the tabular and graphical visualizations of the best HC clustering as seen on the ClustEval website can be found in Figure B.4 and Figure B.5.

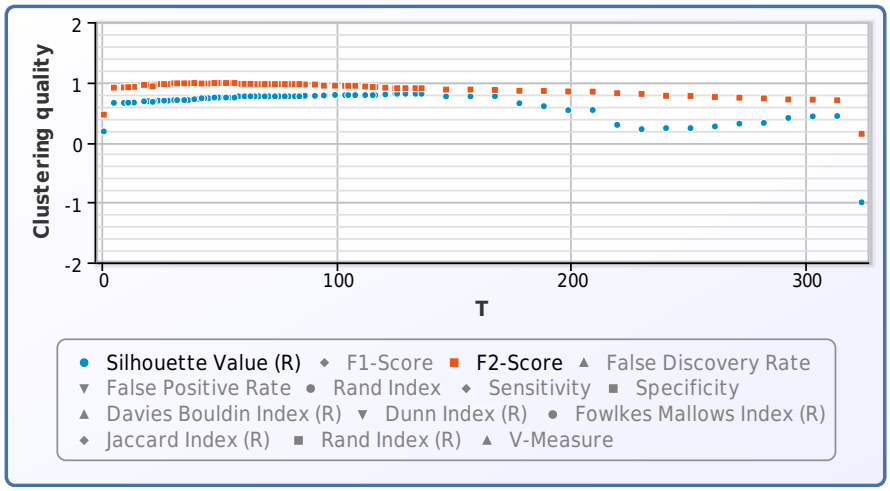


Figure 4.3: Achieved cluster validities of TC on the Brown et al. data set using different density parameter values (threshold T). This visualization is taken from the ClustEval website. Cluster validity indices can be hidden and activated by clicking on the legend.

4.1.7 Setup, Files and Configurations

In this section we want to give the details of how we achieved the data analysis and parameter optimizations, to ease reproducibility.

First of all we placed the Brown et al. data set, gold standard and all used clustering methods in the repository. The data files and clustering methods used will be included in ClustEval by default, thus in the future these do not need to be added manually.

The configuration hierarchy of ClustEval is explained in subsection 2.1.8. Accordingly, the configurations used by the parameter optimization run are visualized in Figure B.1. The analysis run was used to analyze the Brown et al. and other data sets at once. In Figure B.2 the configurations relevant for this data set are shown. We created two run files, one for the parameter optimization and one for the analysis run. For each clustering method a configuration is required as well as a data, data set and gold standard configuration for the data files. The contents of all configuration files are available in Appendix B.

4.1.8 Summary

The data analysis suggests a natural grouping structure to be present in the Brown et al. data set: High clustering coefficient, low graph adhesion, cohesion, min-cut and low intra-inter similarity distribution overlap.

This is confirmed by HC, TC and MC performing very well and clustering the data similar to the gold standard families belonging to amidohydrolase superfamily. HC and TC return clusterings with 6 groups more, than specified in the gold standard, while MC returns 3 groups less than the 19 families. This variance is not affecting the cluster validities to a high degree, because all three methods group the 2 largest families (*urease* (100 proteins) and *AMP deaminase* (28 proteins)) correctly together.

In general, it seems that graph-flow-based methods are less successful on the Brown et al. data set. The good performance of HC suggests, that the clustering structure in the data input is indeed of hierarchical nature.

4.2 Application Case II - Cancer Classification by Clustering Gene Expression Levels

4.2.1 Motivation

In the past, cancer types or subtypes have been identified based on invasive methods and manual expert examination. Usually, these classifications were based on phenotypic aspects like for example tumor shape, consistency or aggressiveness in growth. The availability of molecular data of cancer tissue of different types allows to support this classification, providing at least semi-automatized classification followed by expert validation.

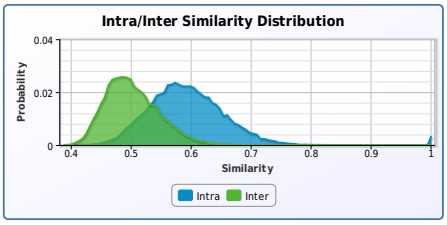
Here we aim to verify, whether clustering can be used to group cancer samples based on molecular data and automatically distinguish between cancer types.

4.2.2 Data Input

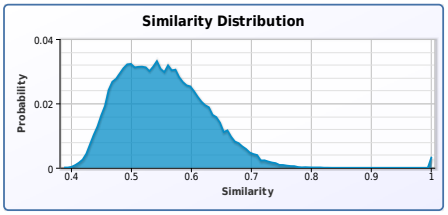
The Cancer Genome Atlas (TCGA) [2] is a project maintaining a database storing molecular information of cancer cells, including gene expression, DNA methylation or copy number aberration. It includes data for many different cancer types, allowing their comparison on a molecular level.

A data set has been derived integrating gene expression levels, DNA methylation and copy number aberration of the three different cancer types, namely

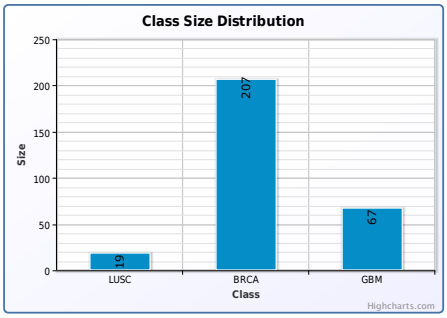
[2] `http://cancergenome.nih.gov/`



(a) The *Similarity Distribution* of the TCGA data set.



(b) The *Intra-vs-Inter Similarity Distribution* of the TCGA data set.



(c) The *Class Size Distribution* of the TCGA data set.

Figure 4.4: Visualizations of data statistics of the TCGA data set on the ClustEval website.

Breast Invasive Carcinoma (BRCA, 207 samples), *Glioblastoma Multiforme* (GBM, 67 samples) and *Lung Squamous Cell Carcinoma* (LUSC, 19 samples) [55]. For each type of molecular information the authors calculated pairwise similarities between the samples using Spearman correlation. This resulted in three similarities for every pair of samples, which were then combined by taking their arithmetic mean.

4.2.3 Data Analysis

We analyzed the TCGA data set using the same data statistics as in the first application case. The results are shown in Figure 4.4 and Table 4.6. The intra vs. inter similarity distribution reveals two large peaks with an overlap of 0.19. This is worse than for the Brown et al. data set.

Graph cohesion and adhesion of 292 indicate a strongly connected similarity graph, which can be explained by the graph density of 1.0. Every pair of nodes is connected by an edge, therefore the number of edges or nodes

Data Statistic	Value
Intra-/Inter Overlap	0.19142913720602453
Graph Cohesion	292.0
Graph Adhesion	292.0
Graph Min-Cut	141.58946962090988
Clustering Coefficient (unweighted)	1.0
Clustering Coefficient (weighted)	0.9982847341337912
Graph Density	1.0
Graph Diversity Average	0.9989428209555552
Matrix Rank	293.0

Table 4.6: The data statistics for the TCGA data set.

Clustering Method	$\mathbf{I_{eff}}$	Parameter Range
AP	621	$preference \in [0.388, 1.0]$
AP	68	$k \in [1, 293]$
MCL	680	$I \in [1.1, 10.0]$
PAM	69	$k \in [1, 292]$
SC	68	$k \in [2, 293]$
TC	991	$T \in [0.388, 1.0]$

Table 4.7: The number of iterations and parameter ranges for each clustering method on the TCGA data set.

to be removed to separate two nodes is high. Eye-inspection of the similarity distribution suggests the average similarity to be around 0.55, thus the graph min-cut of 141.6 indicates, that separating the graph into two connected components costs on average around 250 edges. This is confirmed by high unweighted and weighted clustering coefficients.

4.2.4 Clustering Methods and Parameter Optimizations

We applied AP, HC, MC, PAM, SC and TC to the data. For each clustering method we performed parameter optimizations with the same settings as for the first application case, except for the density parameter ranges and effective number of iterations, Table 4.7.

Clustering Method	**Parameter Value(s)**	**F2-Score**
AP	$preference \in [0.388, 0.395]$	0.293
AP	$k = 2$	0.943
MCL	$I \in [1.1, 10.0]$	0.806
PAM	$k = 2$	0.927
SC	$k = 2$	0.933
TC	$T \in [0.529, 0.533]$	0.984

Table 4.8: The parameter values which led to the highest F2-Scores for each clustering method on the TCGA data set.

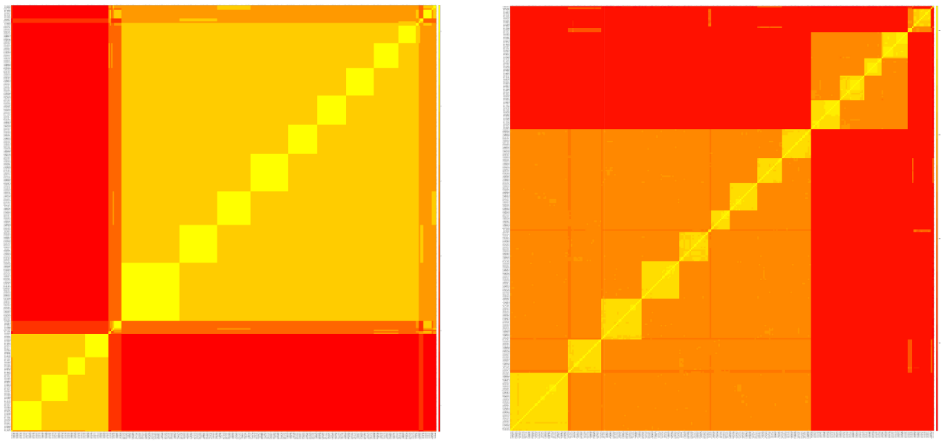
4.2.5 Cluster Validities and Run Analysis

Clustering Method	Davies Bouldin Index	Dunn Index	F1-Score	F2-Score	False Discovery Rate	False Positive Rate	Fowlkes Mallows Index	Jaccard Index	Rand Index	Sensitivity	Silhouette Value	Specificity	V-Measure
AP	0.917	2.051	0.389	0.293	0.0	0.0	0.362	0.133	0.518	0.133	0.057	1.0	0.433
HC	0.905	2.375	0.949	0.943	0.0	0.0	0.926	0.858	0.914	1.0	0.184	1.0	0.828
MC	–	–	0.678	0.806	0.446	1.0	0.744	0.554	0.554	1.0	-1.0	0.0	0.0
PAM	0.919	2.365	0.9	0.927	0.0	0.0	0.909	0.829	0.889	1.0	0.174	1.0	0.701
SC	1.037	2.394	0.909	0.933	0.012	0.0	0.933	0.872	0.92	0.988	0.179	1.0	0.759
TC	0.839	2.363	0.986	0.984	0.0	0.0	0.977	0.956	0.975	1.0	0.164	1.0	0.919

Table 4.9: The achieved cluster validities on the TCGA data set. A table of the same kind as shown in Figure B.3 is presented on the ClustEval website.

The uneven class size distribution of 207 (BRCA), 67 (GBM) and 19 (LUSC) samples (see Figure 4.4(c)) leads to a high cluster validity baseline of some indices, for example the F-Scores. Putting the samples in only 1 or 2 clusters is rated with high qualities, because the smaller LUSC cluster is less important. This explains the high F2-Scores of HC and PAM, although they only cluster the data set into 2 clusters.

The best clustering of HC merges the two classes LUSC and BRCA into



(a) The co-occurrence matrix of the best clustering of each clustering method (6 in total)

(b) The co-occurrence matrix of all clusterings.

Figure 4.5: The results of the run analysis of all clusterings of the TCGA data set.

a single cluster and puts all GBM samples into the second cluster. The best clusterings of SC and PAM follow the same pattern, but additionally have some confusions between GBM and the other classes. The best clustering of TC contains 5 clusters, where two are singletons containing one BRCA sample each, and three LUSC samples are misclassified to the BRCA cluster.

We created a run analysis run to analyze the clusterings in a combined fashion and to see whether the majority of clusterings can help us to distinguish between the classes better. The resulting cooccurrence-matrices of the respective best clustering of each method and of all clusterings can be seen in Figures 4.5(a) and 4.5(b). The co-occurrence matrix of the 6 best clusterings reveals two large clusters, and two very little ones. In contrast, the co-occurrence matrix of all clusterings shows three clusters, corresponding to the cancer subtypes in size. We verified at random whether the groupings make sense. Most samples contained in the smallest cluster correspond to the LUSC class. The other clusters correspond to GBM and BRCA respectively. This analysis indicates that doing ensemble analyses and combining many clusterings can help revealing natural grouping structure in the data set if the judgment of cluster validity indices on single clusterings can be misleading.

4.2.6 Setup, Files and Configurations

First, we placed the TCGA data set and gold standard in the repository.

We reused the program configurations of the first application case. We performed a data-analysis, run-analysis and parameter optimization run. For each of the runs we created a configuration file. For the TCGA data set we created a data, data set and gold standard configuration. All configuration files can be found in the Appendix D.

The configuration hierarchies for the analysis and parameter optimization are presented in Figure D.2 and D.1.

4.2.7 Summary

The data analysis revealed two potential problems: First, the uneven class size distribution causes some cluster validity indices to produce qualities difficult to compare and interpret. Second, intra and inter similarity distribution show an overlap of approximately 0.19. High graph cohesion, adhesion and min-cut and at the same time high clustering coefficients suggest, that the classes in the data could be difficult to separate.

This is partly confirmed by our parameter optimizations. The clustering methods have problems separating the two classes BRCA and LUSC from each other, while the GBM class is clearly separated from the others by most methods. This suggests, that the two cancer types BRCA and LUSC are more similar to each other than they are to the GBM type.

The results of our run analysis, two co-occurrence matrices based on different clusterings of the parameter optimization run, suggest a majority vote clustering into 3 instead of 2 clusters.

4.3 Application Case III - Synthetic Data Generation for Clustering Method Validation

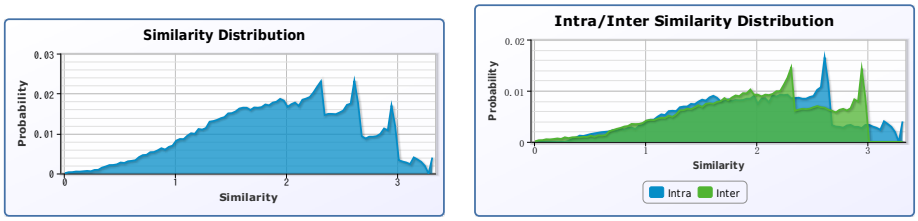
4.3.1 Data Input

Using the generator *SpiralsDataSetGenerator* we created a synthetic data set with 250 two-dimensional data objects, distributed over two entangled spirals. The data format are raw numerical coordinates, which are converted to euclidean pairwise similarities by ClustEval.

Clustering methods supporting raw numerical coordinates as input, namely KM and SC, will be applied directly to these, the other methods will operate on the pairwise similarities.

Data Statistic	Value
Intra-/Inter Overlap	0.418816
Graph Cohesion	248.0
Graph Adhesion	248.0
Graph Min-Cut	110.54602792446511
Clustering Coefficient (unweighted)	0.99998547130599
Clustering Coefficient (weighted)	0.9979634769579863
Graph Density	0.9999678714859437
Graph Diversity Average	0.9908242921576782
Matrix Rank	250.0

Table 4.10: The data statistics for the synthetic spirals250 data set.



(a) The similarity distribution of the spirals250 data set.

(b) The *Intra vs. Inter Similarity Distribution* of the spirals250 data set.

Figure 4.6: Data statistic visualizations of the spirals250 data set on the ClustEval website.

4.3.2 Motivation

Here we demonstrate the usefulness of synthetic data sets to evaluate the performance of clustering methods under specific conditions. Since the data set is non-convex, it is almost impossible to resolve the natural grouping for clustering methods, that are solely based on an approach maximizing between-cluster distances and minimizing within-cluster distances in the original input space. Therefore, methods that either transform the data into a more suitable space, like SC, or that base their cluster identification on local neighborhood similarities will be able to identify the cluster structure.

4.3.3 Data Analysis

The intra vs. inter similarity distribution indicates few separation between intra and inter pair similarities, Table 4.10 and Figure 4.6. This is confirmed

Clustering Method	$\mathbf{I_{eff}}$	Parameter Range
AP	991	$preference \in [0.0, 3.312]$
AP	64	$k \in [1, 250]$
KM	64	$k \in [1, 249]$
MCL	680	$I \in [1.1, 10.0]$
SC	64	$k \in [2, 250]$
TC	752	$T \in [0.0, 3.312]$

Table 4.11: The number of iterations and parameter ranges for each clustering method on the spirals250 data set.

by the intra-inter similarity distribution overlap of 0.42. This relates to the fact that the two spiral clusters are non-convex and entangled, such that for many pairs of objects of the same cluster, objects of the other spiral lie in between. Thus, clustering methods solely based on the pairwise euclidean distances in the original space will not be able to recover the grouping structure.

Graph cohesion and adhesion being 248 indicate a similarity graph, where separation of every pair of two objects requires removal of either 248 nodes or 248 edges. This can be explained with the similarity graph not missing any edges, leading to a density of 1.0 (the difference is due to numerical rounding errors). The minimal cut of 110.55 also shows, that it is costly to separate the graph into two connected components. By looking at the similarity distribution the average similarity is approximately 2.0, thus to separate the graph into two parts it requires to remove approximately 55 edges.

The clustering coefficients, unweighted as well as weighted, are almost 1.0. Combined with the previous analysis results this confirms that the whole graph is tightly connected.

4.3.4 Clustering Methods and Parameter Optimizations

We applied AP, HC, KM, MC, SC and TC to the data. For each clustering method we performed parameter optimizations with the same settings as for the first application case, except for the density parameter ranges and effective number of iterations. Those can be found in Table 4.11.

Clustering Method	**Parameter Value(s)**	**F2-Score**
AP	$preference \in [0.593, 0.620]$	0.181
AP	$k = 1$	0.833
KM	$k = 1$	0.833
MCL	$I \in [1.1, 7.99]$	0.833
SC	$k = 2$	1.0
TC	$T \in [0.0, 0.962]$	0.833

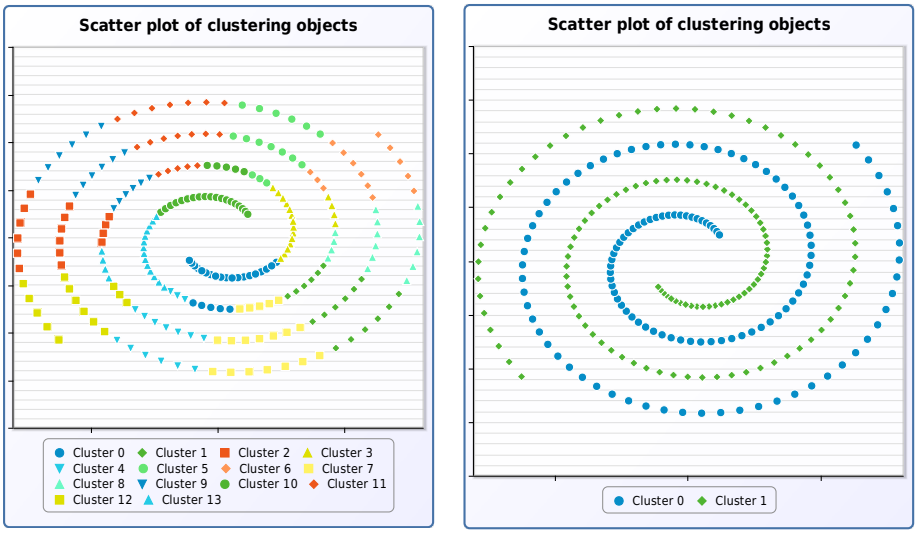
Table 4.12: The parameter values which led to the highest F2-Scores for each clustering method on the spirals250 data set.

4.3.5 Cluster Validities

All cluster validities are shown in Table 4.13, the best achieved F2-Scores for each clustering method in Table 4.12. AP, KM, MCL and TC put all objects into one cluster resulting in a F2-Score of 0.833. This example demonstrates the baseline problem of some cluster validity indices. Although a value of 0.833 seems quite high, the corresponding clustering does not make much sense. This is a characteristic of the F-Scores, which is tackled by some other indices, as for example the V-Measure. The V-Measure qualities of those clusterings are much lower.

SC was able to group the data objects perfectly accordingly to the two spirals (see Figure 4.7(b)). This is achieved by the transformation into the eigenvector space. AP splits the data set into 14 clusters without reflecting the spirals structure.

The Silhouette Value is an example of a validity index not well suited for the provided cluster structure. The perfect separation of SC into the two spirals is rated with a lower Silhouette Value than other solutions.



(a) Visual representation of the clustering with best F2-Score of AP on the spirals250 data set as seen on the ClustEval website.

(b) Visual representation of the clustering with best F2-Score of SC on the spirals250 data set as seen on the ClustEval website.

Figure 4.7: IsoMDS visualizations of clusterings of the synthetic spirals250 data set on the ClustEval website.

Clustering Method	Davies Bouldin Index	Dunn Index	F1-Score	F2-Score	False Discovery Rate	False Positive Rate	Fowlkes Mallows Index	Jaccard Index	Rand Index	Sensitivity	Silhouette Value	Specificity	V-Measure
AP	0.563	3.34	0.255	0.181	0.0	0.0	0.21	0.078	0.521	0.084	0.404	1.0	0.305
HC	0.645	3.724	0.667	0.833	0.0	0.0	0.706	0.498	0.515	1.0	0.388	1.0	0.284
KM	0.639	3.74	0.667	0.833	0.0	0.0	0.706	0.498	0.516	1.0	0.398	1.0	0.28
MC	0.657	3.042	0.667	0.833	0.502	0.5	0.706	0.498	0.498	1.0	0.329	0.5	0.0
SC	0.994	3.662	1.0	1.0	0.0	0.0	1.0	1.0	1.0	1.0	0.364	1.0	1.0
TC	0.598	3.499	0.667	0.833	0.0	0.0	0.706	0.498	0.522	1.0	0.396	1.0	0.304

Table 4.13: The achieved cluster validities on the spirals250 data set. This table is not taken from the ClustEval website.

4.3.6 Setup, Files and Configurations

After creation of a new data set, the generator automatically places the data set and the optional gold standard in the appropriate folders of the repository.

We used the same program configurations as in the second application case, except the one for K-Medoids. Additionally, we created a new program configuration for KM. For each of the two runs, one analysis and one parameter optimization run, we created a configuration file each. For the spirals250 data set we created a data, data set and gold standard configuration. The reused program configurations can be found in Appendix B and Appendix C, all new configuration files in the Appendix D.

The configuration hierarchies for the analysis and parameter optimization are presented in Figure D.2 and Figure D.1.

4.3.7 Summary

The data set was intended as a challenge for clustering methods, since it is an extreme example for non-convex data sets. This was also revealed in an automatic way by the data analysis. The intra-vs-inter similarity distribution, its large overlap and the high min-cut value suggested that the pairwise similarities of the data set shows only little grouping structure.

As a consequence, most clustering methods were not able to cluster the data objects into meaningful groups, but instead either put all objects into a single cluster, like AP, KM, MCL and TC, or in non-sense clusters like AP.

SC is an exception, in that it successfully identifies the two spirals as individual clusters by transforming the raw numerical coordinates into another space. Points lying on the same spiral before, are then located in a single convex cluster after the transformation, such that euclidean distances are meaningful again in the transformed space.

4.4 Discussion

In the introduction we motivated clustering as an important tool in exploratory data analysis. It can be employed when little is known about the given data and researchers need to obtain insights into the data's structure. Often, this is achieved by splitting the data into groups of similar objects. These groups can then be analyzed separately in a deeper way. As mentioned earlier, clustering techniques solve this grouping task. These methods automatize the grouping process researchers had to do manually in the past. But, the increasing number of clustering methods and other factors strongly influence obtained

clustering results, which makes it more difficult and time-consuming to arrive at the right conclusions.

In the three application cases presented before, we demonstrated how our framework ClustEval can be used to ease and automatize many tasks in cluster studies. Data analysis is carried out autonomously and is presented in ways, that allow for a fast interpretation. Also, clustering results are visualized and presented in tables or graphs making comparative analyses easier compared to manual solutions before. ClustEval allows to cluster, even for non-experts with few knowledge about the individual clustering methods or other components used during the clustering process.

Although our framework automatizes many functionalities required in cluster studies, the interpretation of the results is still left to the researcher. ClustEval provides run-data statistics, trying to perform automatic reasoning and detection of potential causalities between input properties and achieved cluster validities, but these are still to be optimized.

Chapter 5

Conclusion

We developed ClustEval, an extensible, open-source framework providing a wide functional repertoire relevant for clusterings, which on the one hand supports researchers in carrying out cluster studies and on the other hand provides interested practitioners with a website as a place to go when there is need for standardized clustering results of previous cluster studies.

ClustEval eases and standardizes clustering tasks by integrating analysis of inputs, data preprocessing, implicit format conversions and autonomous optimization of density parameters by iterative clustering of inputs using sets of clustering methods. Results are stored locally in standard formats and are presented and visualized on a website automatically.

Our framework is extensible such that the user can add new functionality without much hassle and employ it in future cluster studies. By default, ClustEval already contains many important clustering methods, data sets, distance measures, cluster validity indices, and statistics for analysis of data inputs, clustering results or both together. It makes massive use of parallelism when executing clustering tasks and can use many server CPU cores at the same time to accelerate computations. The integration of R into ClustEval allows the easy extension of the framework by other clustering methods, distance measures or statistics already available in the R framework.

ClustEval's website makes it easy to compare performances of clustering methods on the same data set, or of one clustering method on different data inputs. Graphical visualizations complement the interpretation of numerical results. New clustering methods or data sets can also be submitted to an existing website by contacting its maintainers via a contact form.

While related projects like ELKI, KNIME, MLcomp or RapidMiner are capable of automatizing machine learning processes in general, our project

	ClustEval	ELKI	jClust	KNIME Desktop	MLcomp	RapidMiner	Weka
Supports Clustering	✓	✓	✓	✓	(✓)	✓	✓
Clustering Performance Comparison	✓	✗	✗	✗	(✓)	✗	✗
Clustering Visualizations	✓	✓	✓	✗	✗	✗	✓
Parameter Optimization	✓	✗	✗	(✓)	✗	(✓)	✗
Open Results Platform	✓	✗	✗	✗	✓	✗	✗
Data Analysis	✓	(✓)	✗	✓	✗	✓	✓
Data Preprocessing	✓	✓	✗	✓	✗	✓	✓
Data Generation	✓	✓	✗	(✓)	✗	✓	✗
Extensibility	✓	✓	✗	✓	✓	✓	✓
Multi-Threaded	✓	✗	✗	(✓)	✓	✗	✗
Open Source	✓	✓	✓	✓	✓	✓	✓

Table 5.1: Requirement analysis of related projects and ClustEval. Checks are put in brackets if the corresponding functionality is not provided by default but has to be implemented or added by the user.

focuses on clustering in particular and therefore provides computations and visualizations specific to this area. Also, most methods are rather limited to the local computations and do not provide an open results platform for standardized comparison of previous research.

ClustEval is available online at `http://clusteval.mmci.uni-saarland.de`. It can be downloaded and installed, or even modified and extended to the user's needs since it is published under the GNU GPL. A technical documentation is available to guide through the installation of ClustEval on a own server and to provide an introduction into extension of the framework.

Chapter 6

Outlook

In this chapter we present features that are currently under development and others, which could be contributed to ClustEval in the future. Since ClustEval is published open-source under the GNU-GPL, we are positive that some of these features might be contributed by the engaged research community.

6.1 Work in Progress

6.1.1 Run-Data Analysis Runs

We integrated run-data statistics into ClustEval, which allow to search for correlations between data properties and cluster validities. Currently this comprises linear regression models trained on the data statistics, trying to predict the validities achieved by clustering methods. These models are intended to predict the achieved cluster validities of a certain method by only using the properties of the data input. Our studies so far indicate, that we do not have enough clustering results and data sets to get meaningful predictions. The models we trained exhibit a very high degree of variance, and a very low agreement between each other. We compared results of linear regression, as well as lasso and ridge regression methods. The most predictive data statistics were different for the models. Thus we plan to repeat these studies at a time point, when more data and clustering results are available in ClustEval which will make predictions more reliable.

6.1.2 Extending ClustEval for Graph Matching

We are currently extending ClustEval, such that it can also be used for graph matching algorithms. The core of ClustEval has been implemented in a gen-

eralized way, such that this extension is little time intense. This includes integration of graph matching methods, addition and support of graph matching inputs (several input files for each graph), implementation of specific validity indices like Graph Edit Distance (GED) or Edge Correctness (EC) and support of some formats and conversions.

6.2 Future Work

During our work on ClustEval we encountered several interesting ideas we would have liked to integrate into our framework but for which we did not find the time to realize them.

Robustness Analysis. So far our cluster studies were restricted to the analysis of the best performances of clustering methods on data sets. We would like to integrate robustness analysis for clustering methods into the framework. The general idea is to modify a given data input in specific ways and to varying degrees a certain number of times. Then the robustness of the clustering methods on the varied inputs can be measured by comparing the resulting clusterings or the best achieved cluster validity indices.

Complex Optimization Criterion. Currently ClustEval optimizes density parameters by maximizing or minimizing a single cluster validity index. This could be extended to functions, combining several cluster validity indices in a weighted way into a single value. Integrating internal and external validity indices at the same time would allow to find a trade-off between intrinsic distance based and external gold-standard oriented optimization.

Feature Selection. Many prediction methods, and also clustering, suffer from problems as $p \gg n$, the number of samples is much smaller than the dimensionality of the feature space (see [56] for more information on this problem). This problem can be tackled by reducing the feature space to its most important features. This functionality could be added to ClustEval in form of data preprocessors, which take a input data set with raw numerical values and generate new data sets containing only the most relevant features.

Missing Values. Data inputs, irrespective of whether containing pairwise proximities or raw numerical values, may have missing values. Several ways have been proposed [12] to deal with missing values, like for

example *delete corresponding features or samples* or *nearest neighbor inference*. Currently ClustEval does not explicitly handle missing values. In case of pairwise proximities these are just replaced by a default value (0.0) and for raw numerical input, missing values are not supported at all. This functionality could be added to ClustEval as data preprocessors which take the input data, process it in some way and generate a data set with imputed missing values.

Parameterized Statistics. The framework provides many data, run and run-data statistics to analyze inputs, clusterings or both together. Currently these statistics are nonparametric, thus they cannot be fine-tuned. Possible options could for example be the number of buckets for the similarity or node-degree distributions. Also the number of iterations considered in the Hopkins Statistic could be configured.

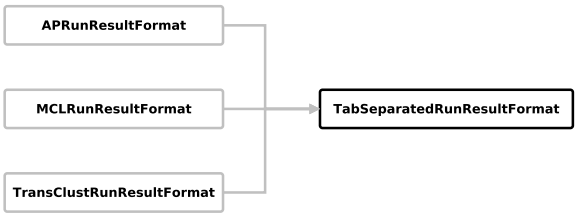
Basis transformations. As seen in the third application case, clustering methods may perform better on a data set, when it is first converted into another basis. Spectral Clustering for example transforms the data points into the eigenvector space of the affinity matrix. This approach could be integrated into ClustEval in general, by providing data preprocessors which take a data set with numerical raw coordinates and transform it into another basis.

Cluster Validity Index Correction. Cluster validity indices are not standardized. This means each of them has its individual expection value, some of them higher than others. Thus, qualities calculated by some index might to be interpreted more conservatively than qualities of another index. Corrected Indices have been proposed equalizing cluster validity indices in terms of their expectation values. In the future, corrected indices could be integrated into ClustEval, but also a framework could be provided which automatically adjusts the qualities by assessing baseline distributions of the indices implicitly.

Appendix A

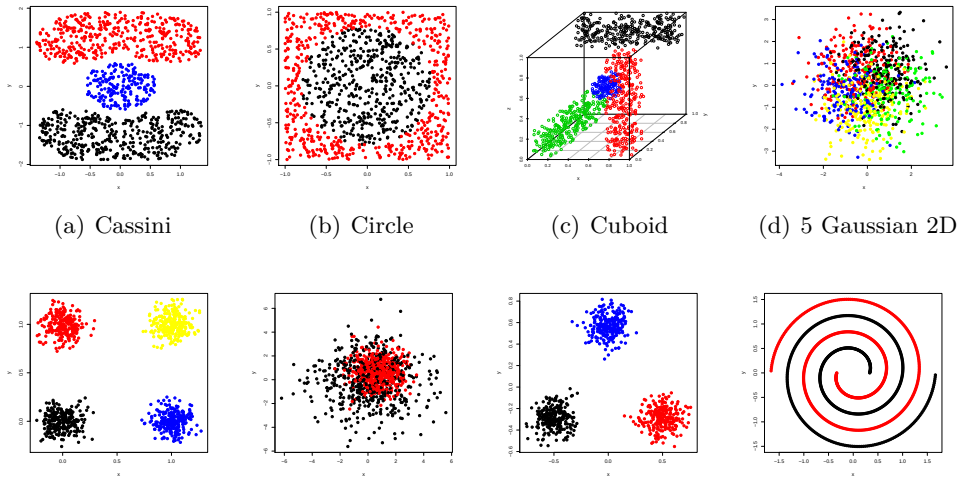
General

(a) Data input format conversions provided by ClustEval.



(b) Result format conversions provided by ClustEval.

Figure A.1: Format conversions included in ClustEval available by default.



(a) Cassini (b) Circle (c) Cuboid (d) 5 Gaussian 2D

(e) Hypercube (f) Ringnorm (g) Simplex (h) Spirals

Figure A.2: Visualizations of available types of synthetic data sets which can be created using the generators of ClustEval.

Appendix B

Application Case I

B.1 Additional Graphs

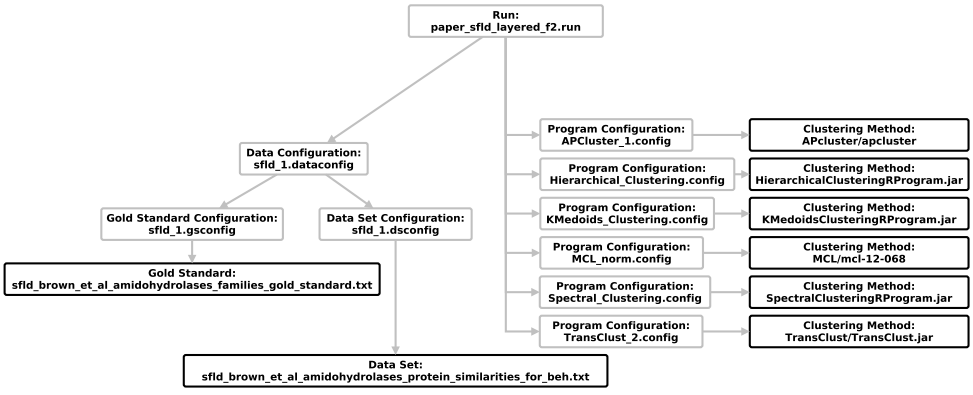


Figure B.1: A visualization of the configurations, files and executables used in the parameter optimization run we created to cluster the Brown et al. data set.

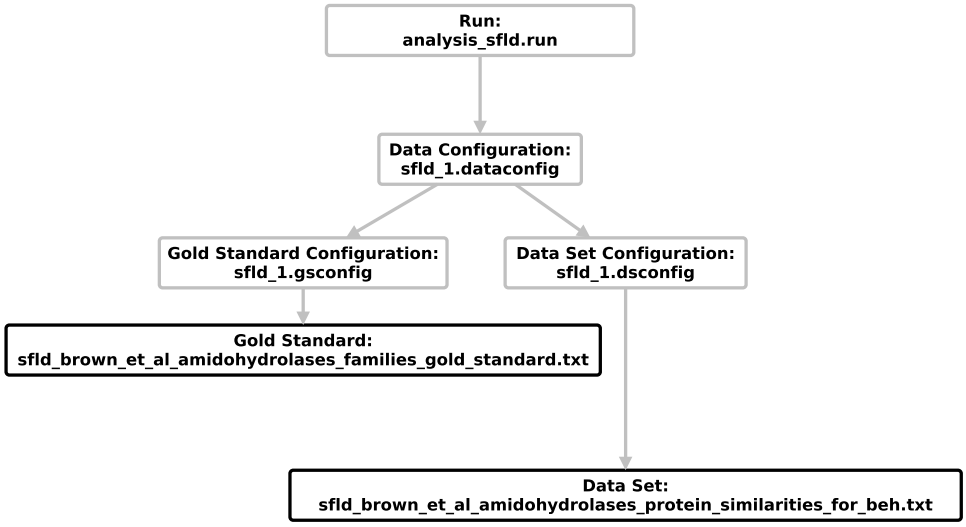


Figure B.2: A visualization of the configurations and files used in the analysis run for the Brown et al. data set.

Clustering Method	Davies Bouldin Index	Dunn Index	F1-Score	F2-Score	False Discovery Rate	False Positive Rate	Fowlkes Mallows Index	Jaccard Index	Rand Index	Sensitivity	Silhouette Value	Specificity	V-Measure
AP	0.763	1.368	0.722	0.712	0.169	0.038	0.781	0.64	0.911	0.749	0.472	0.962	0.741
HC	0.661	3.288	0.987	0.993	0.0	0.0	0.999	0.999	1.0	1.0	0.785	1.0	0.994
MC	0.634	2.769	0.923	0.948	0.012	0.003	0.988	0.976	0.995	1.0	0.772	0.997	0.955
PAM	0.754	4.858	0.912	0.888	0.0	0.0	0.953	0.909	0.979	1.0	0.764	1.0	0.938
SC	–	–	–	–	–	–	–	–	–	–	–	–	–
TC	0.583	10.086	0.986	0.987	0.0	0.0	0.998	0.996	0.999	1.0	0.805	1.0	0.991

Table B.1: The achieved cluster validities on the Brown et al. data set. This table is not taken from the ClustEval website.

Program	Davies Bouldin Index (R)	Dunn Index (R)	F1-Score	F2-Score	False Discovery Rate	False Positive Rate	Fowlkes Mallows Index (R)	Jaccard Index (R)	Rand Index	Rand Index (R)	Sensitivity	Silhouette Value (Java)	Silhouette Value (R)	Specificity	V-Measure
Affinity Propagation	0.763	1.368	0.722	0.712	0.169	0.038	0.781	0.64	0.911	0.911	0.749	--	0.472	0.962	0.741
Fuzzy Clustering (R)	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--
Hierarchical Clustering	0.661	3.288	0.987	0.993	0.0	0.0	0.999	0.999	1.0	1.0	1.0	--	0.785	1.0	0.994
K-Means	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--
Markov Clustering	0.634	2.769	0.923	0.948	0.012	0.003	0.988	0.976	0.995	0.995	1.0	--	0.772	0.997	0.955
Partitioning Around Medoids	0.754	4.858	0.912	0.888	0.0	0.0	0.953	0.909	0.979	0.979	1.0	--	0.764	1.0	0.938
Spectral Clustering	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--
Transitivity Clustering	0.583	10.086	0.986	0.987	0.0	0.0	0.998	0.996	0.999	0.999	1.0	--	0.805	1.0	0.991

Figure B.3: The tabular representation of the optimal achieved cluster validities on the Brown et al. data set as seen on the ClustEval website.

Cluster ID	Objects
0	gi400237 (1.0),
1	gi3287737 (1.0), gi13022196 (1.0), gi12644702 (1.0),
2	gi627281 (1.0), gi171366 (1.0), gi6322218 (1.0), gi728827 (1.0),
3	gi348315 (1.0), gi6141557 (1.0),
4	gi3287876 (1.0), gi3287879 (1.0), gi13194736 (1.0),
5	gi16078516 (1.0),
6	gi24115394 (1.0), gi16132149 (1.0), gi15804904 (1.0), gi16763320 (1.0), gi16767756 (1.0),
7	gi16130785 (1.0), gi13929094 (1.0), gi15803420 (1.0), gi15833010 (1.0), gi6753960 (1.0), gi7533042 (1.0), gi6330926 (1.0), gi20270696 (1.0), gi4758426 (1.0), gi887833 (1.0), gi11837778 (1.0),
8	gi171053 (1.0), gi2134756 (1.0), gi5922016 (1.0), gi5922018 (1.0), gi11359582 (1.0), gi399033 (1.0), gi4557311 (1.0), gi644509 (1.0), gi16945394 (1.0), gi18859679 (1.0), gi608499 (1.0), gi21294189 (1.0), gi21311925 (1.0), gi7638159 (1.0), gi7484807 (1.0), gi6753048 (1.0), gi178547 (1.0), gi6323606 (1.0), gi20302047 (1.0), gi19111945 (1.0), gi2494043 (1.0), gi2494044 (1.0), gi18404701 (1.0), gi4502079 (1.0), gi995562 (1.0), gi13928736 (1.0), gi21264318 (1.0), gi13938134 (1.0),
9	gi15643577 (1.0),
10	gi15212234 (1.0), gi17977844 (1.0), gi14719485 (1.0), gi13786719 (1.0), gi2392286 (1.0), gi12084365 (1.0), gi2098312 (1.0),
11	gi9837270 (1.0),
12	gi15802037 (1.0), gi28373486 (1.0), gi16129581 (1.0), gi6680636 (1.0), gi3892028 (1.0), gi20141178 (1.0), gi24113013 (1.0), gi1518868 (1.0), gi4557249 (1.0), gi18426812 (1.0),
13	gi7531230 (1.0),
14	gi12313641 (1.0), gi14599415 (1.0), gi14599413 (1.0), gi23101955 (1.0), gi4545295 (1.0), gi23020534 (1.0), gi2624847 (1.0), gi23502230 (1.0), gi224797 (1.0), gi22976226 (1.0), gi167228 (1.0), gi7320802 (1.0), gi21637179 (1.0), gi9967069 (1.0), gi1718044 (1.0), gi23000615 (1.0), gi3659633 (1.0), gi465008 (1.0), gi19070377 (1.0), gi2708800 (1.0), gi19310948 (1.0), gi15611138 (1.0), gi16080717 (1.0), gi4249601 (1.0), gi22125140 (1.0), gi17402589 (1.0), gi22831365 (1.0), gi3915207 (1.0), gi7839375 (1.0), gi16974911 (1.0), gi15889672 (1.0), gi14278353 (1.0), gi13357995 (1.0), gi15220459 (1.0), gi15600061 (1.0), gi23501177 (1.0), gi16329675 (1.0), gi10336584 (1.0), gi2507522 (1.0), gi23058236 (1.0), gi19338960 (1.0), gi14579320 (1.0), gi5913962 (1.0), gi16122873 (1.0), gi16272483 (1.0), gi1174891 (1.0), gi23027998 (1.0), gi22297549 (1.0), gi914897 (1.0), gi13474128 (1.0), gi17231162 (1.0), gi16974909 (1.0), gi21637192 (1.0), gi22988607 (1.0), gi15966223 (1.0), gi225714 (1.0), gi14192678 (1.0), gi21637244 (1.0), gi17987935 (1.0), gi43635 (1.0), gi7245287 (1.0), gi731075 (1.0), gi2340847 (1.0), gi23126033 (1.0), gi23136289 (1.0), gi227457 (1.0), gi1073151 (1.0), gi886331 (1.0), gi17546751 (1.0), gi15608987 (1.0), gi137071 (1.0), gi137070 (1.0), gi137073 (1.0), gi15925280 (1.0), gi137074 (1.0), gi19856524 (1.0), gi23039910 (1.0), gi586164 (1.0), gi14599161 (1.0), gi21637205 (1.0), gi421269 (1.0), gi21637202 (1.0), gi15807978 (1.0), gi21219743 (1.0), gi7272374 (1.0), gi7245484 (1.0), gi15800666 (1.0), gi6633958 (1.0), gi13605430 (1.0), gi3688063 (1.0), gi13959023 (1.0), gi1174889 (1.0), gi1174887 (1.0), gi2501620 (1.0), gi21637208 (1.0), gi586167 (1.0), gi14599435 (1.0), gi17986930 (1.0), gi15612817 (1.0), gi2501623 (1.0),
15	gi191364 (1.0), gi17532699 (1.0), gi279582 (1.0), gi400909 (1.0), gi7331 (1.0), gi21294381 (1.0), gi3024509 (1.0), gi22832393 (1.0), gi131696 (1.0), gi3915831 (1.0), gi2506025 (1.0), gi191173 (1.0), gi8509 (1.0),
16	gi8895751 (1.0), gi15833269 (1.0), gi15803675 (1.0),
17	gi16078614 (1.0), gi3122654 (1.0), gi1346930 (1.0),
18	gi15801179 (1.0), gi16764519 (1.0), gi16129025 (1.0), gi279570 (1.0), gi1651519 (1.0), gi24112473 (1.0), gi16760039 (1.0),
19	gi22218649 (1.0), gi3122243 (1.0), gi9931313 (1.0),
20	gi3182927 (1.0), gi17432321 (1.0),
21	gi15800066 (1.0), gi15595634 (1.0), gi1246354 (1.0), gi16120505 (1.0), gi16128322 (1.0), gi16766629 (1.0), gi16762096 (1.0), gi1657533 (1.0), gi15602430 (1.0),
22	gi21362600 (1.0), gi3334230 (1.0), gi15595638 (1.0), gi22988246 (1.0), gi21730171 (1.0), gi3122248 (1.0), gi18478562 (1.0), gi3122249 (1.0), gi23062845 (1.0), gi23200220 (1.0),
23	gi16080988 (1.0),
24	gi11890745 (1.0), gi4033703 (1.0), gi6226558 (1.0),

Figure B.4: Tabular representation of a clustering of the Brown et al. data set on the ClustEval website.

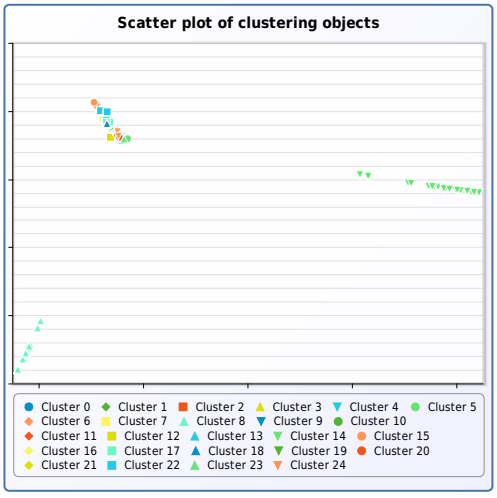


Figure B.5: 2-dimensional IsoMDS visualization of the high-dimensional data objects of the Brown et al. data set as seen on the ClustEval website.

B.2 Configuration Files

B.2.1 Run: paper_sfld_layered_f2.run

```
programConfig = TransClust_2,Hierarchical_Clustering,↩
    ↪ Spectral_Clustering,APcluster_1,MCL_norm,↩
    ↪ KMedoids_Clustering
dataConfig = sfld_1
qualityMeasures = SilhouetteValueRClusteringQualityMeasure,↩
    ↪ TransClustFClusteringQualityMeasure,↩
    ↪ TransClustF2ClusteringQualityMeasure,↩
    ↪ FDRClusteringQualityMeasure,FPRClusteringQualityMeasure,↩
    ↪ RandIndexClusteringQualityMeasure,↩
    ↪ SensitivityClusteringQualityMeasure,↩
    ↪ SpecificityClusteringQualityMeasure,↩
    ↪ DaviesBouldinIndexRClusteringQualityMeasure,↩
    ↪ DunnIndexRClusteringQualityMeasure,↩
    ↪ FowlkesMallowsIndexRClusteringQualityMeasure,↩
    ↪ JaccardIndexRClusteringQualityMeasure,↩
    ↪ RandIndexRClusteringQualityMeasure,↩
    ↪ VMeasureClusteringQualityMeasure
mode = parameter_optimization
optimizationMethod = LayeredDivisiveParameterOptimizationMethod
optimizationCriterion = TransClustF2ClusteringQualityMeasure
optimizationIterations = 1001

[TransClust_2]
optimizationParameters = T

[MCL_norm]
optimizationParameters = I

[APcluster_1]
optimizationParameters = preference,maxits,convits,dampfact
optimizationMethod = APParameterOptimizationMethod

[Hierarchical_Clustering]
optimizationParameters = k

[Spectral_Clustering]
optimizationParameters = k

[Kmedoids_Clustering]
optimizationParameters = k
```

B.2.2 Data Configuration: sfld_1.dataconfig

```
datasetConfig = sfld_1
goldstandardConfig = sfld_1
```

B.2.3 Dataset Configuration: sfld_1.dsconfig

```
datasetName = sfld
datasetFile = ↩
    ↪ sfld_brown_et_al_amidohydrolases_protein_similarities_for_beh↩
    ↪ .txt
```

B.2.4 Gold standard Configuration: sfld_1.gsconfig

```
goldstandardName = sfld
goldstandardFile = ↩
    ↪ sfld_brown_et_al_amidohydrolases_families_gold_standard.↩
    ↪ txt
```

B.2.5 Program Configuration: APcluster_1.config

```
program = APcluster/apcluster
parameters = preference,maxits,convits,dampfact
optimizationParameters = preference,maxits,convits,dampfact
compatibleDataSetFormats = APRowSimDataSetFormat
outputFormat = APRunResultFormat
alias = Affinity Propagation

[invocationFormat]
invocationFormat = %e% %i% %preference% %o% maxits=%maxits% ↩
    ↪ convits=%convits% dampfact=%dampfact%

[maxits]
desc = Max iterations
type = 1
def = 2000
minValue = 2000
maxValue = 5000

[convits]
desc = Cluster Center duration
type = 1
def = 200
minValue = 200
maxValue = 500

[dampfact]
type = 2
def = 0.9
minValue = 0.7
maxValue = 0.99

[preference]
desc = Preference
```

```
type = 2
def = $(meanSimilarity)
minValue = $(minSimilarity)
maxValue = $(maxSimilarity)
```

B.2.6 Program Configuration: Hierarchical_Clustering.config

```
type = HierarchicalClusteringRProgram
parameters = k
optimizationParameters = k

[k]
desc = Number clusters
type = 1
def = 1
minValue = 1
maxValue = $(numberOfElements)
```

B.2.7 Program Configuration: KMedoids_Clustering.config

```
type = KMedoidsClusteringRProgram
parameters = k
optimizationParameters = k

[k]
desc = Number clusters
type = 1
def = 1
minValue = 1
maxValue = $(numberOfElements)-1
```

B.2.8 Program Configuration: MCL_norm.config

```
program = MCL/mcl-12-068
parameters = I
optimizationParameters = I
compatibleDataSetFormats = RowSimDataSetFormat
outputFormat = MCLRunResultFormat
alias = Markov Clustering
expectsNormalizedDataSet = true

[invocationFormat]
invocationFormat = %e% %i% --abc -o %o% -I %I%

[I]
desc = Inflation
type = 2
def = 2.0
```

```
16 minValue = 1.1
17 maxValue = 10.0
```

B.2.9 Program Configuration: Spectral_Clustering.config

```
 1 type = SpectralClusteringRProgram
 2 parameters = k
 3 optimizationParameters = k
 4
 5 [k]
 6 desc = Number clusters
 7 type = 1
 8 def = 2
 9 minValue = 2
10 maxValue = $(numberOfElements)
```

B.2.10 Program Configuration: TransClust_2.config

```
 1 program = TransClust/TransClust.jar
 2 parameters = T,mode,minT,maxT,tss
 3 optimizationParameters = T
 4 compatibleDataSetFormats = RowSimDataSetFormat
 5 outputFormat = TransClustRunResultFormat
 6 alias = Transitivity Clustering
 7
 8 [invocationFormat]
 9 invocationFormat = java -jar %e% -i %i% -sim %i% -gs %gs% -o %o↩
    ↪ % -minT %T% -maxT %T% -tss %tss% -mode %mode% -verbose
10 invocationFormatWithoutGoldStandard = java -jar %e% -i %i% -sim↩
    ↪  %i% -o %o% -minT %T% -maxT %T% -tss %tss% -mode %mode% ↩
    ↪ -verbose
11 invocationFormatParameterOptimization = java -jar %e% -i %i% -↩
    ↪ sim %i% -gs %gs% -o %o% -minT %minT% -maxT %maxT% -tss %↩
    ↪ tss% -mode %mode% -verbose
12 invocationFormatParameterOptimizationWithoutGoldStandard = java↩
    ↪  -jar %e% -i %i% -sim %i% -o %o% -minT %minT% -maxT %↩
    ↪ maxT% -tss %tss% -mode %mode% -verbose
13
14 [T]
15 desc = Threshold
16 type = 2
17 def = $(meanSimilarity)
18 minValue = $(minSimilarity)
19 maxValue = $(maxSimilarity)
20
21 [minT]
22 desc = minimal Threshold
23 type = 2
24 def = $(minSimilarity)
```

```
25 minValue = $(minSimilarity)
26 maxValue = $(maxSimilarity)
27
28 [maxT]
29 desc = maximal Threshold
30 type = 2
31 def = $(maxSimilarity)
32 minValue = $(minSimilarity)
33 maxValue = $(maxSimilarity)
34
35 [tss]
36 desc = Threshold stepsize
37 type = 2
38 def = 0.01
39 minValue = 0.0
40 maxValue = 1.0
41
42 [mode]
43 type = 1
44 def = 2
```

Appendix C

Application Case II

C.1 Additional Graphs

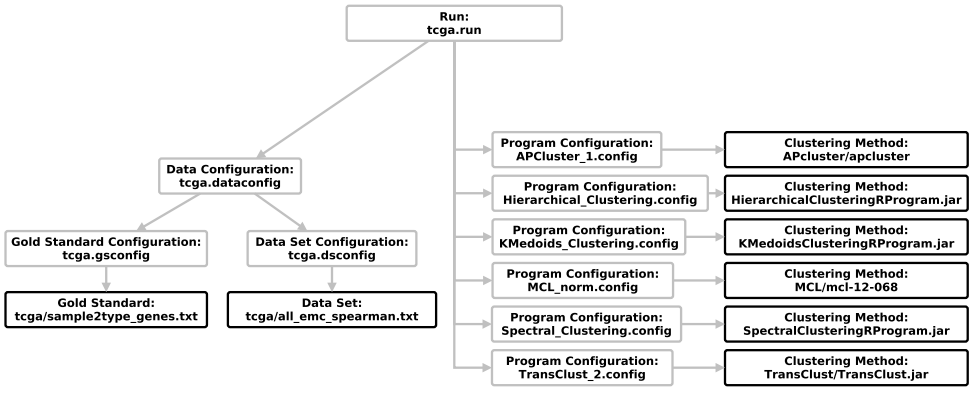


Figure C.1: A visualization of the configurations, files and executables used in the parameter optimization run we created to cluster the TCGA data set.

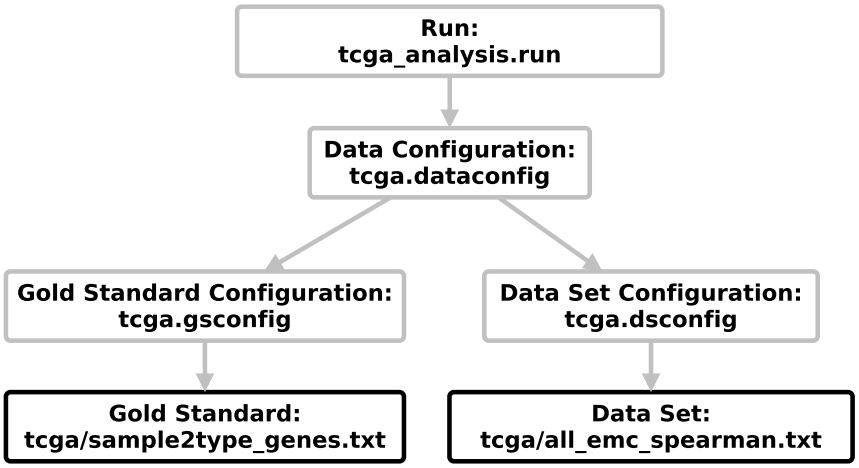


Figure C.2: A visualization of the configurations and files used in the data analysis run for the TCGA data set.

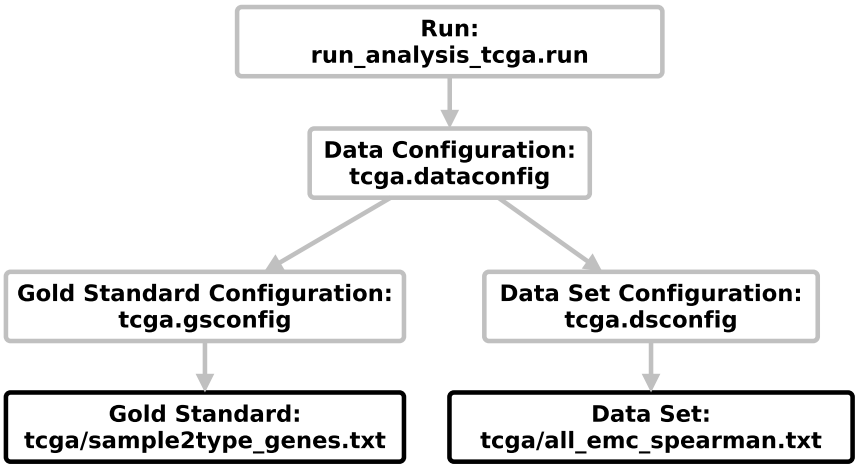


Figure C.3: A visualization of the configurations and files used in the run analysis run for the TCGA data set.

Appendix D

Application Case III

D.1 Additional Graphs

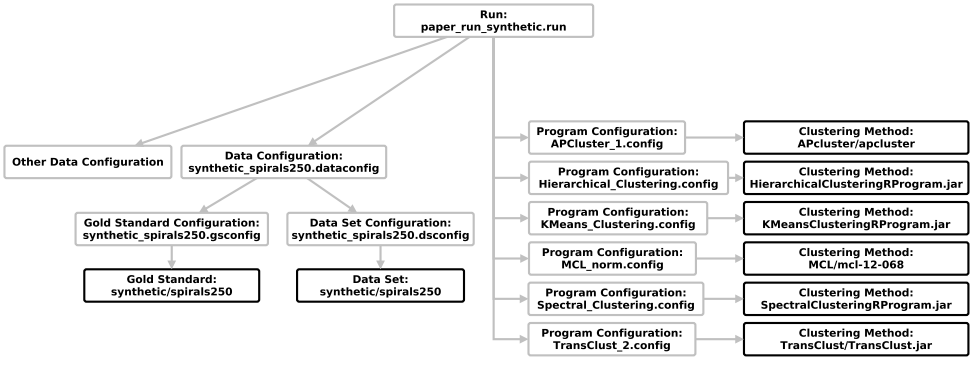


Figure D.1: A visualization of the configurations, files and executables used in the parameter optimization run we created to cluster the spirals250 data set.

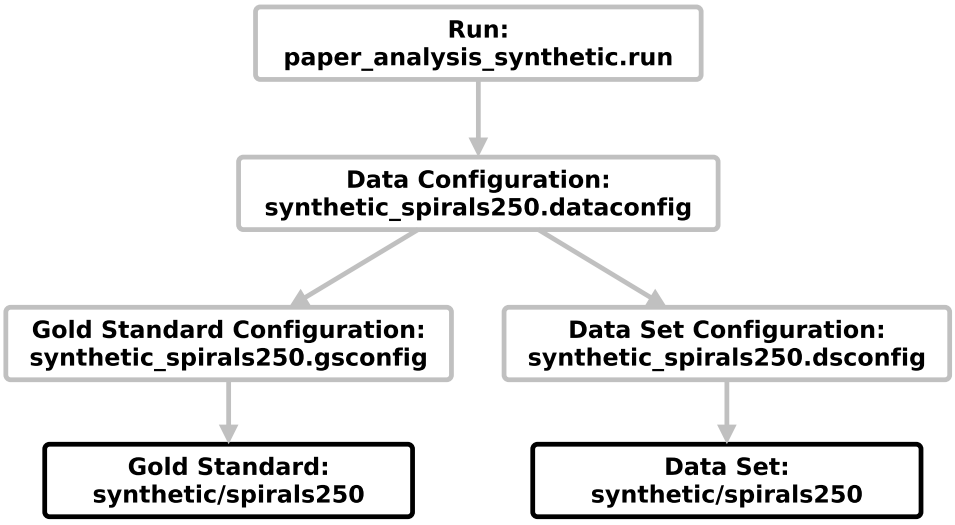


Figure D.2: A visualization of the configurations and files used in the analysis run for the spirals250 data set.

D.2 Configuration Files

D.2.1 Run: paper_run_synthetic.run

```
1 programConfig = APcluster_1,TransClust_2,MCL_norm,↩
    ↪ KMeans_Clustering,Hierarchical_Clustering,↩
    ↪ Spectral_Clustering
2 dataConfig = synthetic_spirals250
3 qualityMeasures = SilhouetteValueRClusteringQualityMeasure,↩
    ↪ TransClustFClusteringQualityMeasure,↩
    ↪ TransClustF2ClusteringQualityMeasure,↩
    ↪ FDRClusteringQualityMeasure,FPRClusteringQualityMeasure,↩
    ↪ RandIndexClusteringQualityMeasure,↩
    ↪ SensitivityClusteringQualityMeasure,↩
    ↪ SpecificityClusteringQualityMeasure,↩
    ↪ DaviesBouldinIndexRClusteringQualityMeasure,↩
    ↪ DunnIndexRClusteringQualityMeasure,↩
    ↪ FowlkesMallowsIndexRClusteringQualityMeasure,↩
    ↪ JaccardIndexRClusteringQualityMeasure,↩
    ↪ RandIndexRClusteringQualityMeasure,↩
    ↪ VMeasureClusteringQualityMeasure
4 mode = parameter_optimization
5 optimizationMethod = LayeredDivisiveParameterOptimizationMethod
```

```
optimizationCriterion = TransClustFClusteringQualityMeasure
optimizationIterations = 1001

[TransClust_2]
optimizationParameters = T

[MCL_norm]
optimizationParameters = I

[APcluster_1]
optimizationParameters = preference,dampfact,maxits,convits
optimizationMethod = APParameterOptimizationMethod

[Hierarchical_Clustering]
optimizationParameters = k

[Spectral_Clustering]
optimizationParameters = k
```

D.2.2 Data Configuration: synthetic_spirals250.dataconfig

```
datasetConfig = synthetic_spirals250
goldstandardConfig = synthetic_spirals250
```

D.2.3 Data Set Configuration: synthetic_spirals250.dsconfig

```
datasetName = synthetic
datasetFile = spirals250
firstRowHeader = 0
```

D.2.4 Gold Standard Configuration: synthetic_spirals250.gsconfig

```
goldstandardName = synthetic
goldstandardFile = spirals250
```

D.2.5 Program Configuration: KMeans_Clustering.config

```
type = KMeansClusteringRProgram
parameters = k
optimizationParameters = k

[k]
desc = Number clusters
type = 1
def = 1
minValue = 1
maxValue = $(numberOfElements)-1
```

Bibliography

[1] Elke Achtert, Hans-Peter Kriegel, and Arthur Zimek. Elki: A software system for evaluation of subspace clustering algorithms. In Bertram Ludäscher and Nikos Mamoulis, editors, *Scientific and Statistical Database Management*, volume 5069 of *Lecture Notes in Computer Science*, pages 580–585. Springer Berlin Heidelberg, 2008.

[2] U. Alon, N. Barkai, D. A. Notterman, K. Gish, S. Ybarra, D. Mack, and A. J. Levine. Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *Proc Natl Acad Sci U S A*, 96(12):6745–6750, Jun 1999.

[3] Stephen F. Altschul, Warren Gish, Webb Miller, Eugene W. Myers, and David J. Lipman. Basic local alignment search tool. *Journal of Molecular Biology*, 215(3):403 – 410, 1990.

[4] M. R. Anderberg. *Cluster Analysis for Applications*. Academic Press, 1973.

[5] Michael R. Berthold, Nicolas Cebron, Fabian Dill, Thomas R. Gabriel, Tobias Kötter, Thorsten Meinl, Peter Ohl, Christoph Sieb, Kilian Thiel, and Bernd Wiswedel. KNIME: The Konstanz Information Miner. In *Studies in Classification, Data Analysis, and Knowledge Organization (GfKL 2007)*. Springer, 2007.

[6] Victor L. Brailovsky. A probabilistic approach to clustering. *Pattern Recognition Letters*, 12(4):193 – 198, 1991.

[7] Steven E Brenner, Patrice Koehl, and Michael Levitt. The astral compendium for protein structure and sequence analysis. *Nucleic Acids Research*, 28(1):254–256, 2000.

[8] Sylvain Brohée and Jacques van Helden. Evaluation of clustering algorithms for protein-protein interaction networks. *BMC Bioinformatics*, 7:488, 2006.

[9] Shoshana D Brown, John A Gerlt, Jennifer L Seffernick, and Patricia C Babbitt. A gold standard set of mechanistically diverse enzyme superfamilies. *Genome Biol*, 7(1):R8, 2006.

[10] Gabor Csardi and Tamas Nepusz. The igraph software package for complex network research. *InterJournal*, Complex Systems:1695, 2006.

[11] David L. Davies and Donald W. Bouldin. A cluster separation measure. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, PAMI-1(2):224–227, 1979.

[12] John K Dixon. Pattern recognition with partly missing data. *Systems, Man and Cybernetics, IEEE Transactions on*, 9(10):617–621, 1979.

[13] Stijn Dongen. A cluster algorithm for graphs. Technical report, Amsterdam, The Netherlands, The Netherlands, 2000.

[14] Richard C. Dubes. How many clusters are best? - an experiment. *Pattern Recognition*, 20(6):645 – 663, 1987.

[15] Joseph C Dunn. Well-separated clusters and optimal fuzzy partitions. *Journal of cybernetics*, 4(1):95–104, 1974.

[16] A. J. Enright, S. Van Dongen, and C. A. Ouzounis. An efficient algorithm for large-scale detection of protein families. *Nucleic Acids Res*, 30(7):1575–1584, Apr 2002.

[17] Ronald A Fisher. The use of multiple measurements in taxonomic problems. *Annals of eugenics*, 7(2):179–188, 1936.

[18] R. J. Focht and S. L. Adams. Tissue specificity of type i collagen gene expression is determined at both transcriptional and post-transcriptional levels. *Mol Cell Biol*, 4(9):1843–1852, Sep 1984.

[19] Edward B Fowlkes and Colin L Mallows. A method for comparing two hierarchical clusterings. *Journal of the American statistical association*, 78(383):553–569, 1983.

[20] Brendan J Frey and Delbert Dueck. Clustering by passing messages between data points. *Science*, 315(5814):972–976, Feb 2007.

[21] Reinhard Furrer, Douglas Nychka, and Stephen Sain. *fields: Tools for spatial data*, 2012. R package version 6.6.3.

[22] R. H. Getzenberg. Nuclear matrix and the regulation of gene expression: tissue specificity. *J Cell Biochem*, 55(1):22–31, May 1994.

[23] T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, M. A. Caligiuri, C. D. Bloomfield, and E. S. Lander. Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *Science*, 286(5439):531–537, Oct 1999.

[24] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. The weka data mining software: an update. *SIGKDD Explor. Newsl.*, 11(1):10–18, November 2009.

[25] J. A. Hartigan and M. A. Wong. A k-means clustering algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 28(1):pp. 100–108, 1979.

[26] Trevor Hastie and Brad Efron. *lars: Least Angle Regression, Lasso and Forward Stagewise*, 2012. R package version 1.1.

[27] Katherine A. Heller and Zoubin Ghahramani. Bayesian hierarchical clustering. In *Proceedings of the 22nd international conference on Machine learning*, ICML '05, pages 297–304, New York, NY, USA, 2005. ACM.

[28] Sean D. Hooper and Peer Bork. Medusa: a simple tool for interaction graph analysis. *Bioinformatics*, 21(24):4432–4433, Dec 2005.

[29] Brian Hopkins and JG Skellam. A new method for determining the type of distribution of plant individuals. *Annals of Botany*, 18(2):213–227, 1954.

[30] Paul Jaccard. *Etude comparative de la distribution florale dans une portion des Alpes et du Jura*. Impr. Corbaz, 1901.

[31] Anil K. Jain and Richard C. Dubes. *Algorithms for clustering data*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1988.

[32] Alexandros Karatzoglou, Alex Smola, Kurt Hornik, and Achim Zeileis. kernlab – an S4 package for kernel methods in R. *Journal of Statistical Software*, 11(9):1–20, 2004.

[33] Leonard Kaufman and Peter J. Rousseeuw. *Finding Groups in Data: An Introduction to Cluster Analysis (Wiley Series in Probability and Statistics)*. Wiley-Interscience, March 2005.

[34] Nevan J Krogan, Gerard Cagney, Haiyuan Yu, Gouqing Zhong, Xinghua Guo, Alexandr Ignatchenko, Joyce Li, Shuye Pu, Nira Datta, Aaron P Tikuisis, Thanuja Punna, José M Peregrín-Alvarez, Michael Shales, Xin Zhang, Michael Davey, Mark D Robinson, Alberto Paccanaro, James E Bray, Anthony Sheung, Bryan Beattie, Dawn P Richards, Veronica Canadien, Atanas Lalev, Frank Mena, Peter Wong, Andrei Starostine, Myra M Canete, James Vlasblom, Samuel Wu, Chris Orsi, Sean R Collins, Shamanta Chandran, Robin Haw, Jennifer J Rilstone, Kiran Gandi, Natalie J Thompson, Gabe Musso, Peter St Onge, Shaun Ghanny, Mandy H Y Lam, Gareth Butland, Amin M Altaf-Ul, Shigehiko Kanaya, Ali Shilatifard, Erin O'Shea, Jonathan S Weissman, C. James Ingles, Timothy R Hughes, John Parkinson, Mark Gerstein, Shoshana J Wodak, Andrew Emili, and Jack F Greenblatt. Global landscape of protein complexes in the yeast saccharomyces cerevisiae. *Nature*, 440(7084):637–643, Mar 2006.

[35] Friedrich Leisch and Evgenia Dimitriadou. *mlbench: Machine Learning Benchmark Problems*, 2010. R package version 2.1-1.

[36] D. J. Lockhart and E. A. Winzeler. Genomics, gene expression and dna arrays. *Nature*, 405(6788):827–836, Jun 2000.

[37] Martin Maechler, Peter Rousseeuw, Anja Struyf, Mia Hubert, and Kurt Hornik. *cluster: Cluster Analysis Basics and Extensions*, 2012. R package version 1.14.3 — For new features, see the 'Changelog' file (in the package source).

[38] Ingo Mierswa, Michael Wurst, Ralf Klinkenberg, Martin Scholz, and Timm Euler. Yale: Rapid prototyping for complex data mining tasks. In Lyle Ungar, Mark Craven, Dimitrios Gunopulos, and Tina Eliassi-Rad, editors, *KDD '06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 935–940, New York, NY, USA, August 2006. ACM.

[39] Glenn W Milligan. A monte carlo study of thirty internal criterion measures for cluster analysis. *Psychometrika*, 46(2):187–199, 1981.

[40] GlennW. Milligan and MarthaC. Cooper. A study of standardization of variables in cluster analysis. *Journal of Classification*, 5(2):181–204, 1988.

[41] Byron JT Morgan and Andrew PG Ray. Non-uniqueness and inversions in cluster analysis. *Applied Statistics*, pages 117–134, 1995.

[42] F. Murtagh. *Multidimensional clustering algorithms*. 1985.

[43] A. G. Murzin, S. E. Brenner, T. Hubbard, and C. Chothia. Scop: a structural classification of proteins database for the investigation of sequences and structures. *J Mol Biol*, 247(4):536–540, Apr 1995.

[44] S. B. Needleman and C. D. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *J Mol Biol*, 48(3):443–453, Mar 1970.

[45] Lukasz Nieweglowski. *clv: Cluster Validation Techniques*, 2009. R package version 0.3-2.

[46] Philipp Pagel, Stefan Kovac, Matthias Oesterheld, Barbara Brauner, Irmtraud Dunger-Kaltenbach, Goar Frishman, Corinna Montrone, Pekka Mark, Volker Stümpflen, Hans-Werner Mewes, Andreas Ruepp, and Dmitrij Frishman. The mips mammalian protein-protein interaction database. *Bioinformatics*, 21(6):832–834, Mar 2005.

[47] Georgios A Pavlopoulos, Charalampos N Moschopoulos, Sean D Hooper, Reinhard Schneider, and Sophia Kossida. jclust: a clustering and visualization toolbox. *Bioinformatics*, 25(15):1994–1996, Aug 2009.

[48] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2012. ISBN 3-900051-07-0.

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

[50] C. J. Van Rijsbergen. *Information Retrieval*. Butterworth-Heinemann, Newton, MA, USA, 2nd edition, 1979.

[51] Andrew Rosenberg and Julia Hirschberg. V-measure: A conditional entropy-based external cluster evaluation measure. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 410–420, 2007.

[52] D. T. Ross, U. Scherf, M. B. Eisen, C. M. Perou, C. Rees, P. Spellman, V. Iyer, S. S. Jeffrey, M. Van de Rijn, M. Waltham, A. Pergamenschikov, J. C. Lee, D. Lashkari, D. Shalon, T. G. Myers, J. N. Weinstein, D. Botstein, and P. O. Brown. Systematic variation in gene expression patterns in human cancer cell lines. *Nat Genet*, 24(3):227–235, Mar 2000.

[53] Peter J Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20:53–65, 1987.

[54] Mihaela E. Sardiu, Laurence Florens, and Michael P. Washburn. Evaluation of clustering algorithms for protein complex and protein interaction network assembly. *J Proteome Res*, 8(6):2944–2952, Jun 2009.

[55] Nora Speicher. Towards the identification of cancer subtypes by integrative clustering of molecular data. Master's thesis, Universität des Saarlandes, December 2012.

[56] Robert Tibshirani, Guenther Walther, and Trevor Hastie. Estimating the number of clusters in a data set via the gap statistic. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(2):411–423, 2001.

[57] Stijn van Dongen and Cei Abreu-Goodger. Using mcl to extract clusters from networks. In *Bacterial Molecular Networks*, pages 281–295. Springer, 2012.

[58] Stijn Marinus van Dongen. Graph clustering by flow simulation. 2000.

[59] W. N. Venables and B. D. Ripley. *Modern Applied Statistics with S*. Springer, New York, fourth edition, 2002. ISBN 0-387-95457-0.

[60] Tobias Wittkop. Clustering biological data by unraveling hidden transitive substructures. 2010.

[61] Tobias Wittkop, Jan Baumbach, Francisco P Lobo, and Sven Rahmann. Large scale clustering of protein sequences with force -a layout based heuristic for weighted cluster editing. *BMC Bioinformatics*, 8:396, 2007.

[62] Tobias Wittkop, Dorothea Emig, Sita Lange, Sven Rahmann, Mario Albrecht, John H Morris, Sebastian Böcker, Jens Stoye, and Jan Baumbach. Partitioning biological data with transitivity clustering. *Nat Methods*, 7(6):419–420, Jun 2010.

[63] Tobias Wittkop, Dorothea Emig, Anke Truss, Mario Albrecht, Sebastian Böcker, and Jan Baumbach. Comprehensive cluster analysis with transitivity clustering. *Nat Protoc*, 6(3):285–295, Mar 2011.

[64] Tobias Wittkop, Sven Rahmann, Richard Röttger, Sebastian Böcker, and Jan Baumbach. Extension and robustness of transitivity clustering for protein–protein interaction network analysis. *Internet Mathematics*, 7(4):255–273, 2011.

[65] Wayne W Zachary. An information flow model for conflict and fission in small groups. *Journal of anthropological research*, pages 452–473, 1977.

[66] C.T. Zahn. Graph-theoretical methods for detecting and describing gestalt clusters. *Computers, IEEE Transactions on*, C-20(1):68–86, 1971.