



Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Bachelorarbeit

Anastasiya Vladimirova

**Analysis and Comparison of Various Clustering and
Dimensionality Reduction Algorithms on LIDAR-Ceilometer
Aerosol-Backscatter Data**

*Fakultät Technik und Informatik
Studiendepartment Informatik*

*Faculty of Engineering and Computer Science
Department of Computer Science*

Anastasiya Vladimirova

**Analysis and Comparison of Various Clustering and
Dimensionality Reduction Algorithms on LIDAR-Ceilometer
Aerosol-Backscatter Data**

Bachelorarbeit eingereicht im Rahmen der Bachelorprüfung

im Studiengang Bachelor of Science Angewandte Informatik
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr. Michael Neitzke
Zweitgutachter: Prof. Dr. Peer Stelldinger

Eingereicht am: 22. July 2021

Anastasiya Vladimirova

Thema der Arbeit

Analysis and Comparison of Various Clustering and Dimensionality Reduction Algorithms on LIDAR-Ceilometer Aerosol-Backscatter Data

Stichworte

Clustering, Rückstreudaten, Dimensionsreduktion, Hard-Clustering, Spektrales Clustering, Hierarchisches Clustering, Dichte-basiertes Clustering, PCA, UMAP, T-SNE, LIDAR-Ceilometer, Unüberwachtes Lernen, Hochdimensionales Clustering, Subspace-Clustering

Kurzzusammenfassung

Diese Studie bietet einen umfassenden Überblick über verfügbare Clustering-Methoden und analysiert deren Leistung auf den Merkmalsvektoren von LIDAR-Ceilometerdaten. Nach der Identifizierung relevanter Merkmale der Cluster und der Auswahl der entsprechenden Metriken fährt die Arbeit mit dem Vergleich verfügbarer Methoden zur Partitionierung von Daten fort. In Anbetracht der Tatsache, dass die Daten hochdimensional sind, werden auch die dafür spezialisierten Clustering-Methoden in die Untersuchung einbezogen. Die Studie beinhaltet weiterhin eine Faktorenanalyse der Daten und eine anschließende Reduktion dieser Daten auf niedrigere Dimensionen, die dann demselben Clustering-Analyseprozess unterzogen werden.

Anastasiya Vladimirova

Title of the paper

Analysis and Comparison of Various Clustering and Dimensionality Reduction Algorithms on LIDAR-Ceilometer Aerosol-Backscatter Data

Keywords

clustering, backscatter data, dimensionality reduction, hard-clustering, spectral clustering, hierarchical clustering, density-based clustering, PCA, UMAP, T-SNE, LIDAR-ceilometer, unsupervised learning, high-dimensional clustering, subspace clustering

Abstract

This study provides a comprehensive overview of available clustering methods and analyses their performance on the feature vectors of LIDAR-ceilometer data. After identifying relevant characteristics of the clusters and choosing the corresponding metrics, the work proceeds with comparing available methods for partitioning data. Considering that the data is high-dimensional, the specialized clustering methods are also included into examination. The study further incorporates factor analysis of the data and subsequent reducing these data to lower dimensions, which then undergo same clustering analysis process.

Contents

1. Introduction	1
1. Classical Clustering Methods	4
2. Background	5
2.1. Clustering Validation	5
2.1.1. Clustering Characteristics	7
2.1.2. Clustering Validity Metrics	8
2.1.3. Cluster Visual Representation	9
2.2. Clustering Technique Analysis	9
2.2.1. Centroid Clustering	9
2.2.2. Spectral Clustering	9
2.2.3. Hierarchical Clustering	11
2.2.4. Density-Based Clustering	12
2.3. Data Factor Analysis	14
2.3.1. Kaiser-Meyer-Olkin Criterium	15
2.3.2. Bartlett-Test	15
2.3.3. Result Analysis	15
2.4. Dimensionality Reduction	16
2.4.1. Principal Component Analysis	16
2.4.2. T-distributed Stochastic Neighbor Embedding	16
2.4.3. Uniform Manifold Approximation and Projection	17
2.4.4. Overview	18
3. Methods	19
3.1. Analysis Procedure	19
3.2. Data Description	20
3.3. Software Description	20
3.4. Parameter Estimation for DBSCAN	20
3.5. Interactive Cluster Visualization	21
4. Results	23
4.1. Clustering Metrics	23
4.1.1. Clustering on Encoded Data	23
4.1.2. Clustering on Reduced Data	24

4.2. Visual Representation	25
4.2.1. Clustering on Encoded Data	25
4.2.2. Clustering on Reduced Data	28
4.3. 2D Visualization of Clusters	33
4.3.1. Clustering on Encoded Data	33
4.3.2. Clustering on Reduced Data	34
5. Discussion	37
II. High-Dimensional Clustering Methods	39
6. Background	40
7. Methods	42
7.1. Parameter Settings	42
7.2. Software	42
8. Results	43
8.1. Clustering Metrics	43
8.2. Visual Representation	43
8.3. 2D Visualization	46
9. Discussion	47
III. Clustering Original Data	48
10. Motivation	49
11. Results	50
11.1. Clustering Metrics	50
11.2. Visual Representation	50
11.3. 2D Visualization of Clusters	53
12. Conclusion	54

List of Tables

2.1. General Notation	8
3.1. DBSCAN Parameters	21
4.1. Metrics Result: Clustering on Encoded Data	24
4.2. Metrics Result: Clustering on UMAP Reduced Data	25
4.3. Metrics Result: Clustering on PCA Reduced Data	25
8.1. Metrics Result: High-Dimensional Clustering on Encoded Data	43
11.1. Metrics Result: Clustering on UMAP Reduced Original Data	50

List of Figures

1.1.	CHM 15k „NIMBUS“2 LIDAR-Ceilometer	2
1.2.	Wind profile from Hamburg 14.05.2021 21:00	2
2.1.	Difference between K-Means and spectral clustering	10
2.2.	Sorted k-distance plot by Sander u. a. (1998): the sorted distances of each object to the 3d neighbour.	14
3.1.	Sorted k-distance plots for eps estimation.	20
3.2.	Interactive cluster visualization tool	22
4.1.	Average representation of each cluster of clustering methods on encoded data.	26
4.2.	Data partitioning of clustering methods.	27
4.3.	Average representation of each cluster of clustering methods on UMAP reduced data.	29
4.4.	Average representation of each cluster of clustering methods on PCA reduced data.	30
4.5.	Data partitioning of clustering methods on UMAP reduced data.	31
4.6.	Data partitioning of clustering methods on PCA reduced data.	32
4.7.	UMAP visualization of clustering methods on encoded data.	33
4.8.	UMAP visualization of clustering methods on the UMAP reduced data.	35
4.9.	UMAP visualization of clustering methods on the PCA reduced data.	36
8.1.	Average representation of high-dimensional clustering methods and the clustering method from previous work on encoded data.	44
8.2.	Data partitioning of high-dimensional clustering methods and the clustering method from previous work on encoded data.	45
8.3.	UMAP visualization of of high-dimensional clustering methods and the clustering method from previous work on encoded data.	46
11.1.	Average representation of each cluster of clustering methods on original data reduced to 2D by UMAP.	51
11.2.	Data partitioning of clustering methods on original data reduced to 2D by UMAP.	52
11.3.	UMAP visualization of clustering methods on original data reduced to 2D by UMAP.	53

1. Introduction

Wind profile data is a vital part of the weather forecast. The particular class of instruments that measure aerosol backscatter profiles is called automatic lidars and ceilometers, Illingworth u. a. (2019). Automatic lidars and ceilometers (ALC) create wind profile by measure cloud base, cloud thickness and aerosol layers. They do it by shining a laser pulse into the sky and detecting the reflection from aerosol particles with the help of a photocell in the receiver. The return time of the single is then transferred into the spatial range with the speed of light.

Not only does it help to identify weather events and air quality, but it is also essential for aircraft because backscatter profiles provide information about fog and volcanic ash. These ground-based profiling instruments are spread around Europe and incorporated into the E-PROFILE network. E-PROFILE collects data from its members, including DWD - Deutscher Wetterdienst, and offers an interactive map of wind profiles gathered across all stations.

Usually, ceilometers' measurements are affected by a high ratio of noise; that is why multiple profiles are averaged to intensify the signal.

The high sensitivity of ceilometers might produce false readings, for example, when icy particles are reported as cloud height equal zero.

ALCs are capable of continuous 24-7 operation with high sampling rates. DWD utilizes 100 devices CHM 15k „NIMBUS“2 LIDAR-Ceilometer, which allows to measure the range up to 15 km. This model has a very high-sensitivity due to its single wavelength backscattering technique. The device is illustrated in figure 1.1.

The ceilometer stores all measured backscatter profiles in a day file of NetCDF format (Network Common Data File). For the detailed description of variables saved for each ceilometer measurements, please refer to the manual G. Lufft Mess- und Regeltechnik GmbH. The visualization of data is presented in figure 1.2.

All of the generous amounts of data by ALCs produced each day are stored without labels and can only be interpreted by weather experts. Machine learning can potentially be utilized to classify weather events in data produced by ceilometers.

In 2019, Niels Gandraß presented his work "Unsupervised Machine Learning: Feature Extraction for Classification of LIDAR-Ceilometer Aerosol-Backscatter Profiles" - Gandraß

1. Introduction



Figure 1.1.: CHM 15k „NIMBUS“2 LIDAR-Ceilometer

Source:

www.lufft.com/products/cloud-height-snow-depth-sensors-288/ceilometer-chm-15k-nimbus-2300/

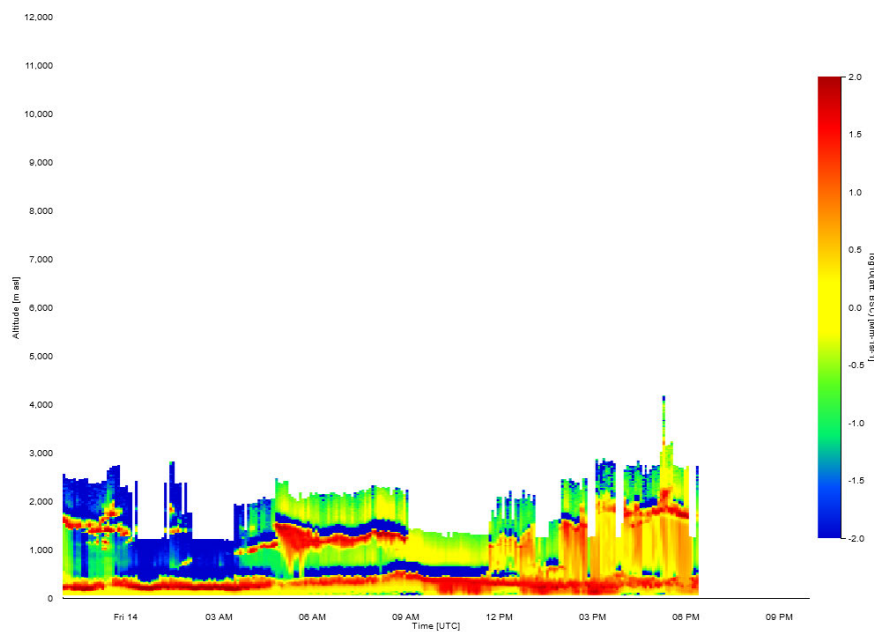


Figure 1.2.: Wind profile from Hamburg 14.05.2021 21:00

Source: e-profile.eu/

(2019), where he developed a system that creates a feature representation of each measurement and then proceeds with partitioning the dataset of these features into different groups that are meant to be representative of diverse weather events.

Since the data does not contain labels, only unsupervised learning can be used, e.g., clustering.

The work of Niels Gandraß introduces the data of the ceilometer and describes calibration and normalization in great detail.

Considering that this work only deals with feature vectors created by the model of the previous project, it is sufficient to know that data goes through the following stages:

Step 1: The NetCDF data of each day is calibrated.

Step 2: Data is reduced in size (for example, eight measurements are combined along the height and four measurements along the time).

Step 3: Data is split into smaller slices.

Step 4: Autoencoder is trained on these slices to extract a feature vector.

Each slice contains 2048 backscatter variables, meaning that data is high-dimensional. Clustering algorithms perform poorly on such types of data; that is why autoencoder was used to reduce the data to manageable 32 dimensions. Autoencoder is a type of neural network that learns an efficient representation in an unsupervised manner, Kramer (1991). Autoencoder attempts to copy input to the output. It consists of two parts - encoder and decoder. An encoder maps the input object (in this case, slice of wind profile) to the code (representation), and a decoder maps the code to a reconstruction of the input. The great advantage of the autoencoder is its ability to capture nonlinear dependencies between an object's variables.

After the autoencoder is trained and each weather slice is represented by a code, or so-called feature vector, previous work proceeded with partitioning the dataset into groups. The chosen clustering method was K-Means. The results looked promising but left room for improvement.

This work aims to explore other clustering algorithms and compare how they perform on the feature vectors.

In case of successful partitioning, the labels of assigned clusters could be used to train the classifier and, with that, provide a machine learning application that could automatically analyze the weather data.

The study consists of two parts: the first part introduces the classical clustering algorithms and analyses their performance on LIDAR data, and the second one deals with high-dimensional methods. Both parts contain background, methodology and discussion sections. The work ends with a conclusion chapter.

Part I.

Classical Clustering Methods

2. Background

2.1. Clustering Validation

Clustering is a vital mining technique that identifies subgroups of data. Overall, clustering attempts to maximize the intra-cluster similarity and minimize inter-cluster similarity. There are at least 100 clustering methods, and it is not a trivial task to choose which method suits data the most. Usually, the choice of an appropriate algorithm is heavily influenced by the characteristics of data, e.g., dimensionality and structure of the data, and the goal of clustering, like classification or exploration. Although, for many real-life datasets, it is rather hard to make a prediction, which algorithm will show the best results. Therefore, it is common to try and compare a few methods, which is also the goal of this work. The arising question is how to determine which algorithm performed best. One of the possible options is showing the data that was assigned to each cluster. Since this approach relies on human expertise, it is not deterministic or reliable. Some clusters might have apparent separation, whereas the difference between others can be subtle. To make the assessment of quality less questionable and easier to interpret, people use various metrics that measure different aspects of clusters. These aspects are also referred to as desirable characteristic of clustering, among which are the following ones:

- Between cluster separation
- Within cluster consistency
- Good representation of data by centroids
- Similar size of clusters
- Convex shape

and other, all presented by Hennig u. a. (2015).

Before attempting at clustering and validating, it is important to identify relevant characteristics. These characteristics can then be measured by the corresponding cluster-quality

2. Background

measurement techniques. Most of the time, there are several requirements for a clustering algorithm, resulting in multiple metrics, that are used to measure the cluster quality. However, it is usually not possible to achieve great scores in all existing requirements.

The following paragraph introduces an example that aims to show the importance of the clustering aspects.

Before attempting clustering and validating, it is crucial to identify relevant characteristics. The corresponding cluster-quality measurement techniques can then measure these characteristics. Most of the time, there are several requirements for clustering algorithms, resulting in multiple metrics that are used to measure the cluster quality. However, it is usually not possible to achieve great scores in all existing requirements.

The following paragraph introduces an example that aims to show the importance of the clustering aspects.

A dataset published by Franck *u. a.* (2004) contains genetic information about 236 Australasian tetragonula bees, and the goal of clustering this dataset is identifying species. The approach to this problem is discussed in Hennig (2017). The first step would be identifying relevant clustering characteristics. The definition of biological species requires close to no genetic exchange between species, meaning that one of the aspects to measure would be cluster separation. Small within-cluster distances is another characteristic that is relevant because organisms of the same species must share many genes. On the other hand, the shape of the within-cluster density, representation of clusters by centroids, entropy, and other possible characteristics are not of concern for the overall goal. After all the relevant aspects are identified, one can proceed with choosing corresponding metrics. When the appropriate metrics are found, clustering algorithms can be performed and assessed.

The same paradigm will be utilized with the LIDAR dataset.

Hennig *u. a.* (2015) brings attention to the problem that to the date of publishing (2015), there was no existing research on investigating the characteristics of methods and how different clustering algorithms could be used to achieve set requirements. The literature research did not produce any new papers on the topic. Therefore multiple clustering algorithms will be introduced in this work, and their quality will be assessed with the help of several chosen metrics.

There are various metrics available for different aspects of clustering. For an overview, please refer to Halkidi *u. a.* (2015). Another paper by Hennig (2017) proposes standardized metrics that can be combined in different ways to be better suited for individual purposes of clustering.

The following section briefly introduces the aforementioned clustering characteristics and identifies the relevant ones for the LIDAR dataset. The section after that describes the metrics that are suited for the chosen aspects of clustering.

2.1.1. Clustering Characteristics

Hennig u. a. (2015) provides at least 14 different possible, desirable aspects of clustering. The relevant aspects can be derived from the overall goal of clustering. We are interested in identifying patterns in data that can potentially be used for classifying weather events. That leads us to the first measurable aspect of the results: clear separating gaps between clusters.

Two approaches are utilized to assess the between cluster separation: measuring distances between cluster centers or measuring the pairwise minimum distances between objects in different clusters.

Another primary aim in most cluster analysis studies is within-cluster similarity. It will also be considered here as one of the critical aspects since weather events should be sufficiently similar to each other in each class.

Another aspect is within-cluster gaps. A gap, in this case, is a split of a cluster into two subgroups so that the minimum dissimilarity between these subgroups is still considerable. It measures the within-cluster similarity from a different perspective.

Characteristics, such as representation of objects by centroids and representation of dissimilarity structure by clustering, are mainly used for information reduction, which is not part of the goal for the LIDAR-dataset.

Among other aspects, there are a few that can also be disregarded for the LIDAR dataset. There are no expectations about the shape of clusters, and there is no limit to variables that should be used for characterizing clusters. Variables can be either dependent or independent in each cluster, and this aspect is also of no interest to us.

Another valuable characteristic of clustering is its stability. Similar data should produce similar partitioning. It is an isolated aspect of the clustering since both desirable and not desirable partitioning might be stable. To measure that, resampling or bootstrapping is used alongside some type of classification. Usually, the new data that was not in the cluster is classified using centroids and then compared with the results of the separate clustering run. Since we are not aiming for a good cluster centroid representation and hence can not use centroids as a classification method, we refrain from measuring stability for the clusters produced by methods in this work.

2.1.2. Clustering Validity Metrics

Popular metrics, like the ones presented by Davies und Bouldin (1979), Dunn (1974) and Kaufman (1990), measure between cluster separation and within-cluster similarity simultaneously.

Hennig (2017) provides individual measurements for each aspect, which helps in the assessment of individual characteristics of each clustering method.

Below are the indices for each relevant aspect of clustering. For notation refer to table 2.1.

Table 2.1.: General Notation

Symbol	Meaning
$D = \{x_1, \dots, x_n\}$	set of objects
$C = \{C_1, \dots, C_K\}$	clustering set
K	amount of clusters
n	amount of objects
$n_j = C_j $	amount of objects in this cluster

For within-cluster similarity:

$$I_{withindis}(C) = \frac{1}{n} \sum_{j=1}^K \frac{2}{n_j - 1} \sum_{x \neq y \in C_j} d(x, y), \quad (2.1)$$

where $d(x, y)$ is a distance function. To normalize the index within $[0, 1]$ the value of $I_{withindis}$ is divided by the maximum distance between two points in the dataset and subtracted from one. The values closer to 1 are better.

For between-cluster separation:

$$I_{p-sep}(C) = \frac{1}{\sum_{j=1}^K \lfloor pn_j \rfloor} \sum_{j=1}^K \sum_{i=1}^{\lfloor pn_j \rfloor} d_{j(i)}, \quad (2.2)$$

where $d_{j(i)}$ is a fraction(p) of nodes distances between node i in cluster j and all other nodes in other clusters sorted in an ascending order. $\lfloor pn_j \rfloor$ is an amount of these distances, taken into calculation for cluster j .

This index allows tolerating odd, very small distances of clusters if in total separation between clusters is good. To achieve that, this index takes the portion p , for example, $p = 0.1$, of a cluster that is closest to the other cluster into the measurement. Whereas usual approaches, like in Dunn (1974), consider only the minimal distance between two clusters, which is not appropriate for more than two clusters since it only considers two closest clusters, and this representation might not be typical for the entire result.

2.1.3. Cluster Visual Representation

Even though visual assessment is not a reliable technique of identifying cluster quality, it is still helpful to represent the results in an easily interpretable image.

Each clustering result will be accompanied by an image with ten random LIDAR measurements assigned to each cluster.

Additionally, each method will have an average cluster representation: all LIDAR-images will be averaged across all assigned LIDAR-measurements for each pixel for each cluster.

To appreciate the data distribution, cluster data will be reduced to two-dimensional points and visualized with the help of UMAP method (for detail, please refer to section 2.4.3).

In total, for each clustering technique and hyperparameters, there will be three plots.

2.2. Clustering Technique Analysis

The following chapter is a survey on existing techniques for clustering data. Each clustering algorithm is assessed regarding how well it might perform on the LIDAR data.

2.2.1. Centroid Clustering

The well-known type of clustering, K-means, also used in previous work, is a typical example of a centroid-based clustering. It is the most widely used algorithm, as reported by Dubey und Choubey (2017). Given a set number K of clusters, this approach tries to find K centroid objects that would best represent other data points in each cluster. This approach is formalized as minimizing the sum of square distances for each cluster, though finding global optimum is computationally hard. The main limitation of this technique is that it finds only ellipsoid-shaped clusters. The result of clustering depends on the initial position of centroids; that is why K-Means attempts to initialize centroids multiple times. A survey by Dubey und Choubey (2017) references several papers that attempt to optimize K-Means, make it more efficient or even take away the limitation of needing to specify the number of clusters in advance. Nevertheless, this algorithm will not be considered further due to its inherent assumption about cluster shape.

2.2.2. Spectral Clustering

First introduced by Donath und Hoffman (1973), spectral clustering is essentially a graph partitioning technique, and it deals with similarity data. The similarity data can be built from the original data, using various distance metrics. The spectral clustering comes from

2. Background

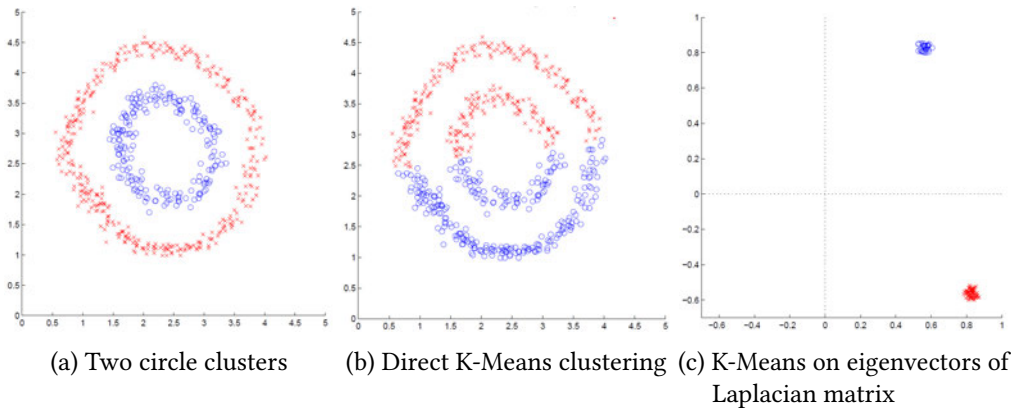


Figure 2.1.: Difference between K-Means and spectral clustering

a family of methods based on the spectral decomposition of the Laplacian matrix. The K principal eigenvectors of the Laplacian matrix map the objects into K dimensions. The obtained K -dimensional data is used to derive K clusters, usually by applying K-Means.

Spectral clustering not only can identify clusters of arbitrary shape but also finds the global optimal solution.

The described approach to spectral clustering is a generalization. There are numerous variations of the algorithms proposed by authors. Possible diversion points include, but are not limited to different similarity graphs, types of graph Laplacian, type of eigenvectors for clustering (first or last K -eigenvectors), and type of clustering method on eigenvectors. An extensive survey can be found in the work of Von Luxburg (2007).

Section 2.2.1 notes that K-Means fails to identify clusters of non-convex shapes. To clarify why this method works when applied on eigenvectors of Laplacian matrix, Ng u. a. (2001) proposes the following example. The dataset is composed of two circles, as presented in the figure 2.1. When K-Means is applied directly to the data, it yields two clusters that are not of the desired shape. Whereas the data mapped with eigenvectors consists of classical convex shapes that can be successfully clustered with K-Means.

One of the recent popular takes on spectral clustering is spectral clustering with collaborative representation, Wang u. a. (2015). Whereas the traditional similarity matrix consists of pairwise distances, this approach considers each data object as a linear representation of all other objects. The authors argue that pairwise metric is very sensitive to outliers, and the collaborative representation does not suffer from this drawback.

The approach is summarized as follows:

Step 1: Each data object $y = X\alpha$, where X are the other data points and α is the representation coefficient matrix of y over X .

Step 2: Calculate distance between representations using cosine similarity.

Step 3: Proceed with spectral clustering

Authors report improved accuracy and efficiency on different models for clustering tests.

A similar approach by Chen und Cai (2011) utilizes representation of data points, but instead of using all available points, this approach creates a landmark representation. Landmarks are several "characteristical" points in the dataset. They can be chosen randomly or with the help of K-Means, and the amount of landmarks is significantly smaller than the size of the dataset. That makes computation much faster.

The algorithm scales linearly with the number of datapoints and is reported by the authors to be as good as the state-of-the-art methods in 2011. The optimization parameter for this algorithm is the number of landmarks. The results are also heavily influenced by a distance measure: cosine, cityblock, and others.

Original spectral clustering, as well as both representative approaches, will be tested on LIDAR-dataset.

2.2.3. Hierarchical Clustering

A hierarchical algorithm creates a nested sequence of partitions presented in a tree structure, also referred to as a dendrogram. There are several levels of clusters; each big cluster includes subclusters, which can also be further divided into smaller ones. These partitions can be built in top-down or bottom-up approaches, also called divisive and agglomerative clustering.

Divisive clustering starts with all elements in one cluster and then performs some chosen technique for cutting the data, then the process repeats for each cluster in an iterative matter. The algorithm stops when each object is its own cluster or when the terminating condition is met. For each iteration holds: once the splitting is done, it can not be reversed.

In contrast to that, agglomerative clustering starts with all objects each being one cluster of one object and then merges clusters that are closer to each other. The process is repeated till the end condition is met, or all objects belong to one cluster.

It is much more computationally intensive to find optimal splits in divisive clustering than to find optimal merges in agglomerative clustering: $2^n - 1$ possible splits for the first level cluster with all n objects vs. $\binom{n}{2}$ with agglomerative clustering, Hennig und Meila (2015). Therefore, agglomerative clustering is more widely used, whereas divisive clusters are a good fit for clustering in which only a single variable is used for splitting.

On the other hand, Roux (2018) reports that divisive algorithms appear to score slightly better in the Silhouette index compared to agglomerative algorithms when used on a real-life dataset including multiple variables in consideration.

Agglomerative clustering's results depend on the chosen approach to calculate the dissimilarity between the merged cluster and all the others. The need for these different methods comes from the fact that a distance between two clusters could be formulated in many ways.

Most famous methods include:

- Single linkage, also known as nearest neighbour: distance between two clusters is the distance between their two closest objects.
- Complete linkage or farthest neighbour: distance between two clusters is the distance between their two farthest objects.
- Average linkage, or UMGMA: distance between two clusters is the arithmetic mean of all the distances between the objects of one and the objects of the other.
- Ward, or MISSQ: distance between two clusters is the magnitude by which the summed square in their joint cluster will be greater than the combined summed square in these two clusters.

Single, complete, and average linkage can be utilized with different distance measures, whereas ward will only work correctly when used with Euclidean distance. Single and complete linkage is faster than the other ways, but ward and average linkage usually provide more meaningful results, Abboud u. a. (2019).

There is no good heuristic about choosing the appropriate linkage method, so all four introduced methods should be tested for agglomerative clustering.

2.2.4. Density-Based Clustering

Density-based methods attempt to identify clusters of high density separated by regions of low density. All the data points found in regions of low-density are treated as noise. Density-based clustering does not require a number of clusters. One of the main advantages of this technique is the ability to identify clusters of different shapes.

This clustering approach does not make an assumption about the underlying density or variance of the data; it can therefore identify clusters that cannot be well described as groups of low within-cluster dissimilarity.

A popular algorithm DBSCAN, proposed by Ester u. a. (1996), utilizes a spatial index structure to perform scalable clustering. The two required parameters are r - distance threshold and k -

density threshold. The density of each point x is the amount k_i of neighbor points in radius r . If a point's k_i is bigger than k , this point is a core point, and a cluster is formed; otherwise it's a border point and the algorithm searches for the next point. The algorithm visits all points in this matter. Radius r is also referred to as *eps*, and the density threshold k as *MinPts*. The choice of these parameters heavily influences the quality of the result. DBSCAN also has the disadvantage of performing poorly if the density varies.

DENCLUE, offered by Hinneburg u. a. (1998), and DBCLASD by Xu u. a. (1998) were introduced as an alternative to DBSCAN.

DENCLUE consists of influence and density function and is essentially a generalization over DBSCAN. The influence function evaluates the impact of a data point on its neighborhood. The density function is a sum of influence of all data points. DENCLUE is also reported to be a much more efficient algorithm than DBSCAN. Hinneburg und Gabriel (2007) presented DENCLUE 2.0, a faster version of the method that still converges exactly towards a local maximum and only sacrifices a small part of accuracy.

DBSCLAD has three important notations for defining a subset of data a cluster C :

- Nearest neighbor distance set of C has the expected distribution with a required confidence level.
- C is maximal.
- C is connected.

A comparison study by Nagpal und Mann (2011) reports that DENCLUE indeed is the fastest method. All three algorithms can identify arbitrary shapes. DBCLASD has the advantage of requiring no parameter, whereas parameter estimation for DENCLUE is difficult. According to the study DENCLUE and DBCLASD handle noise much better than DBSCAN, though cluster quality is best in DENCLUE and DBSCAN.

DBSCAN and DBCLASD will be considered for LIDAR-dataset.

Selecting parameters for DBSCAN is a rather tricky task. One of the methods developed specifically for estimating the radius is AEC - Automatic Eps Calculation by Gorawski und Malczok (2006). The AEC algorithm can only estimate radius in simple datasets with a small noise ratio, while noisy datasets are much more common. That is why it will not be helpful for the feature vectors of LIDAR-dataset.

Determining density threshold, or *MinPts*, has several rules of thumb: it should not be less than the number of dimensions and should increase with the size of the dataset and noise. Sander u. a. (1998) reports that their experiments indicate that the value of $MinPts = 2 * dimensions$ performs well for datasets where each point occurs only one time.

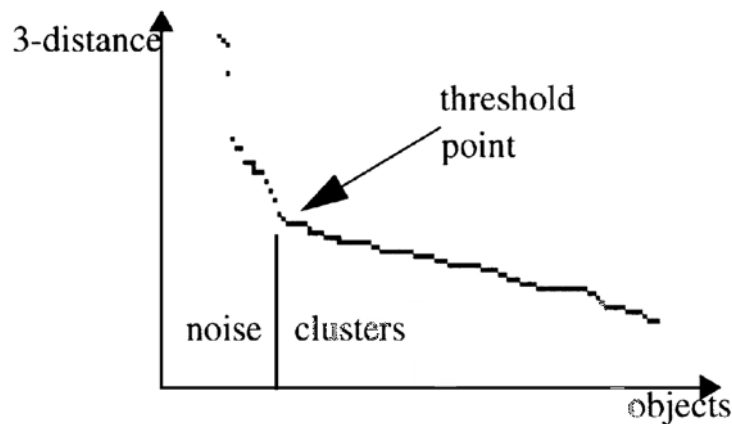


Figure 2.2.: Sorted k -distance plot by Sander et al. (1998): the sorted distances of each object to the 3d neighbour.

For eps search the author offers a "sorted k -distance plot", where $k = MinPts - 1$. After calculating the distance to the k th nearest neighbor and sorting these distances, the threshold object can be identified as an object near the first "valley" of the sorted k -distance plot. The distance of this object is then used as eps . The example is presented in figure 2.2.

2.3. Data Factor Analysis

Each slice of provided data consists of 2048 dimensions, e.g., it is considered to be high-dimensional data. Classical clustering algorithms, especially the ones that are based on the nearest neighbor approach, are reported to be inefficient in high dimensions due to the curse of dimensionality Kriegel et al. (2009). Therefore the previous work on the subject concentrated on reducing the dimensions with the help of an autoencoder. The feature vector, extracted by autoencoder, consists of 32 dimensions. This amount can be classified by several applications as high-dimensional data as well. There is no agreed-upon threshold for assigning a label of "high-dimensions" to a dataset, but regular clustering algorithms tend to have problems with more than ten variables, Han et al. (2011). This chapter investigates whether or not the data compressed by an autoencoder is suitable for dimensionality-reduction techniques, like PCA, UMAP, or t-SNE.

To assess whether or not data can be compressed that way, it is required to perform Kaiser-Meyer-Olkin criterium and the Bartlett test for sphericity.

2.3.1. Kaiser-Meyer-Olkin Criterium

Presented by Kaiser (1970), the Kaiser-Meyer-Olkin (KMO) Measure of Sampling Adequacy is a statistical measure that indicates whether the variance in data is caused by underlying structural factors, which would make the data suitable for factor analysis. KMO calculates the proportion of variance among all the variables in the data. It takes values between 0 and 1, and the results above 0.6 are considered sufficient to proceed with Factor Analysis.

2.3.2. Bartlett-Test

Bartlett's test of sphericity, proposed by Armstrong und Soelberg (1968), shows whether the observed variables of the data inter-correlate at all. In other words, this test checks whether there is some redundancy among variables that a few factors could describe. For that, the test calculates an observed correlation matrix against the identity matrix. The null hypothesis of this test is that the variables do not correlate at all and are entirely independent of one another. To assess whether or not the result of the dependence of variables is significant, we rely on the p-value. If the p-value is below a specific threshold (most common 0.05), the result is considered significant, and we can proceed with factor analysis.

2.3.3. Result Analysis

The encoded data of the LIDAR dataset (the 32-dimensional vector produced by the autoencoder) got a Kaiser-Meyer-Olkin score of 0.749 for 16000 slices and 0.746 for 19200 slices. The score in a range of 0.70s is described as middling by the author of an original paper. Dziuban und Shirkey (1974) reports that with an increasing number of samples, the score improves. Due to computational restriction, it was impossible to calculate the score for the entire dataset, but the score appears to be consistent in the 0.70s sector with various amount of samples.

The Bartlett-Test on the encoded data of 16000 slices yielded a chi result of 502797.35 with a p-value of 0.0; for the 19200 slices, a chi-value of 606846.33 and a p-value of 0.0.

Given that Bartlett-Test produced significant results and the data also satisfied the Kaiser-Meyer-Olkin criterium, it is possible to proceed with dimensionality reduction techniques such as PCA, UMAP and t-SNE, because the encoded data proved to be suitable for structure detection.

2.4. Dimensionality Reduction

In this chapter, several dimensionality reduction approaches will be considered, including their advantages and disadvantages. Their performance will be assessed based on the results of classical clustering algorithms applied to the data. All of these techniques are unsupervised, i.e., do not require labels. The techniques will be applied to already reduced data - feature vectors extracted by the autoencoder. The goal is to reduce 32 dimensions to 2. In addition to improved performance of the clustering algorithms, it will also allow to visualize clustering.

2.4.1. Principal Component Analysis

Principal component analysis (PCA), as described in Abdi und Williams (2010), is a linear transformation technique that utilizes orthogonal transformation, which preserves symmetric inner product of data. A group of correlated variables is converted to a smaller group of uncorrelated variables. The process of converting data to fewer dimensions includes standardizing data, building a co-variance matrix, extracting first k eigenvectors, which will represent new dimensions.

PCA maximizes the variance of data and reduces noise, which is then utilized by clustering techniques. Even though PCA is a widely spread technique, some papers advise against using it. Yeung und Ruzzo (2001) reports that the PCA technique used for dimensionality reduction not only failed to produce better results but also failed to capture the cluster structure. However, this problem could be specific to the type of data. PCA and clustering were performed on gene expression data. The authors also did not report on performing advised tests for factor analysis, like Kaiser-Meyer-Olkin to see whether the data was suitable for PCA.

On the other hand, Ding und He (2004) argues that the use of PCA for clustering, specifically K-Means, in this case, is justified because principal components (PCs) are the continuous solution of the cluster membership indicators.

The obvious drawback of the PCA algorithm is that it can only extract linear features. In addition to that, Jolliffe (1982) draws attention to the problem that there is no way to say in advance that low-variance components will be unimportant and that PCA will not actually lose the discriminatory features in classification.

2.4.2. T-distributed Stochastic Neighbor Embedding

T-distributed stochastic neighbor embedding (t-SNE) is a nonlinear dimensionality reduction technique. It was proposed by Van der Maaten und Hinton (2008) and is widely used for data visualization of high-dimensional clusters. The main advantage of this algorithm is its ability

to deal with linearly nonseparable data. In contrast, the drawback is that it is not deterministic and iterative, meaning that it produces different results after every run. T-SNE calculates the pairwise similarities for input dimensions and then again for reduced dimensions and attempts to minimize the divergence between these distributions. The Sum of Kullback-Leibler divergence is a measure that is being minimized with stochastic gradient descent.

T-SNE is a computationally intensive method. The recommended amount of dimension for an input object is at most 50. This method requires several hyperparameters, for example, perplexity, which is used to balance attention between local and global aspects of the data. Results may change with different hyperparameters, so it is important to run this method multiple times with different parameters and compare the results.

2.4.3. Uniform Manifold Approximation and Projection

Uniform Manifold Approximation and Projection (UMAP), introduced by McInnes u. a. (2018), is another dimension reduction and visualization technique, similar to t-SNE. In comparison to t-SNE, UMAP is a more popular technique for dimension reduction. It allows non-linear reduction as well. The mathematics behind this algorithm are rather complex, but to put it simply, UMAP constructs a representation of the high-dimensional graph and attempts to optimize a low-dimensional graph to have a similar structure as the high-dimensional one. UMAP constructs the high-dimensional graph with the help of fuzzy simplicial complexes, which are essentially weighted graphs, where edge weight is the likelihood of two nodes being connected. To determine the likelihood, UMAP uses the radius of a node. The choice for radius is essential since too big radiuses lead to the network being wholly connected, and too small radiuses result in too many isolated clusters. The radius is determined locally, depending on the nearest neighbors. After the high-dimensional graph is computed, the low-dimensional graph's structure is optimized similarly to how t-SNE does it.

UMAP and t-SNE both use gradient descent but differ in initialization. The original version of t-SNE uses random initialization, whereas UMAP utilizes graph Laplacian. Becht u. a. (2019) argues that UMAP is superior to t-SNE because it can capture the global structure better than its predecessor. Kobak und Linderman (2019) provides another point of view: multiple experiments in this paper showed that t-SNE is as capable of capturing the same global characteristics of the data, given the informative initialization. This study also demonstrated that UMAP with random initialization would perform as poorly as t-SNE.

Overall, this study provides an essential insight about both methods: an initialization technique is crucial for the results.

2. Background

Nevertheless, the UMAP method is significantly faster than t-SNE for the same amount of dimensions and scales better. UMAP can be directly applied to the original data with thousands of dimensions.

2.4.4. Overview

The main difference between the discussed methods is the ability to capture linear vs. non-linear patterns. UMAP and t-SNE both perform similarly; hence only UMAP will be tested. Data is rarely linear separable, that is why UMAP is expected to perform better than PCA.

3. Methods

This section describes the practical part of the analysis, including the procedure, data, software description, and the calculation of parameters for clustering methods.

3.1. Analysis Procedure

The methods chosen for the analysis are spectral clustering, landmark spectral clustering, spectral clustering with collaborative representation, agglomerative clustering, DBSCAN, and DBCLASD.

Some algorithms require the number of clusters to find; in that case this parameter is set to 8 to make it consistent with the previous study.

Landmark spectral clustering has two important parameters: p - the amount of landmarks and the distance measure. Tested p values are 250, 500, 1000, tested distance measures: cosine, cityblock.

Agglomerative clustering requires a choice of dissimilarity measure; the tested methods are single linkage, complete linkage, average linkage, ward linkage.

For DBSCAN parameter estimation, please refer to 3.4.

All the aforementioned methods go through the following analysis steps:

- Calculating simple statistics: the number of clusters, data distribution.
- Calculating cluster validity metrics: within-cluster similarity and between-cluster separation.
- Plotting the average representation of each identified cluster.
- Plotting the random entities assigned to each cluster.
- Plotting the 2d-representation calculated with UMAP.

This process is repeated for encoded data (feature vector), data reduced by UMAP, data reduced by PCA.

3.2. Data Description

LIDAR data for this dataset consists of 6240 slices (similar amount as used for the test in the previous work) from Leipzig 2018 (same resource). Each slice is a 256 by 8 matrix, where 256 is the number of height bins and 8 is the number of samples. The normalization parameters for backscatter were taken from the preceding work's appendix: max is $5.657 e^{-6} m^{-1} sr^{-1}$, min is $-8.944 e^{-7} m^{-1} sr^{-1}$.

The feature vector for each slice is then generated with the help of the autoencoder model - AutoEncoderArch7_1_3_1 from the preceding study, which results in the dataset of 6240 items with 32 dimensions.

The reduction of data is performed with the UMAP and PCA methods. This generates two datasets of 6240 items with two dimensions.

3.3. Software Description

DBSCAN clustering, as well as Agglomerative clustering implementation is provided by Pedregosa u. a. (2011). DBCLASD implementation is an open source project: <https://github.com/spalaciob/py-dbclasd>. Spectral clustering and its variations are a custom implementation.

3.4. Parameter Estimation for DBSCAN

As described in section 2.2.4, DBSCAN requires two parameters: density radius - eps and density threshold - $MinPts$. The plot used for eps prediction are presented in figure 3.1. The parameters for each data type, e.g. encoded or reduced, are listed in table 3.1.

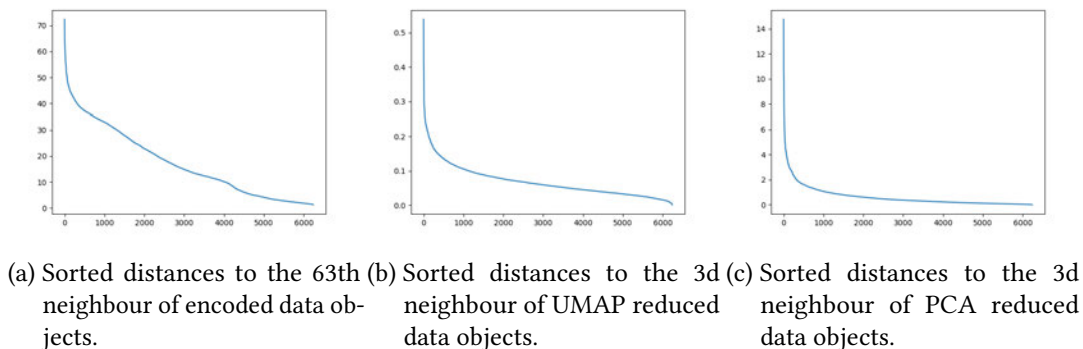


Figure 3.1.: Sorted k-distance plots for eps estimation.

Table 3.1.: DBSCAN Parameters

Data Type	MinPts	Eps
Encoded data	64	35
UMAP reduced data	4	0.15
PCA reduced data	4	2

DBSCAN with these parameters produced unsatisfactory results. The range of several parameters was tested. For encoded data, a range of *MinPts* values between 16 and 80 failed to produce better results. UMAP data has been partitioned in 50+ clusters with different *eps* parameters. Section 2.2.4 mentions that with more data, it is sensible to choose a higher value of *MinPts*, that is why for reduced data *MinPts* has been assigned to $4 * dimensions = 8$.

3.5. Interactive Cluster Visualization

This additional tool, see figure 3.2, allows to interactively explore 2D structure of the LIDAR data. Hovering over the data points visualizes them underneath and allows fast comparison of the weather slices. This tool also makes assessing the quality of the clustering algorithm easier.

It has been created with the help of Inc. (2015).

3. Methods

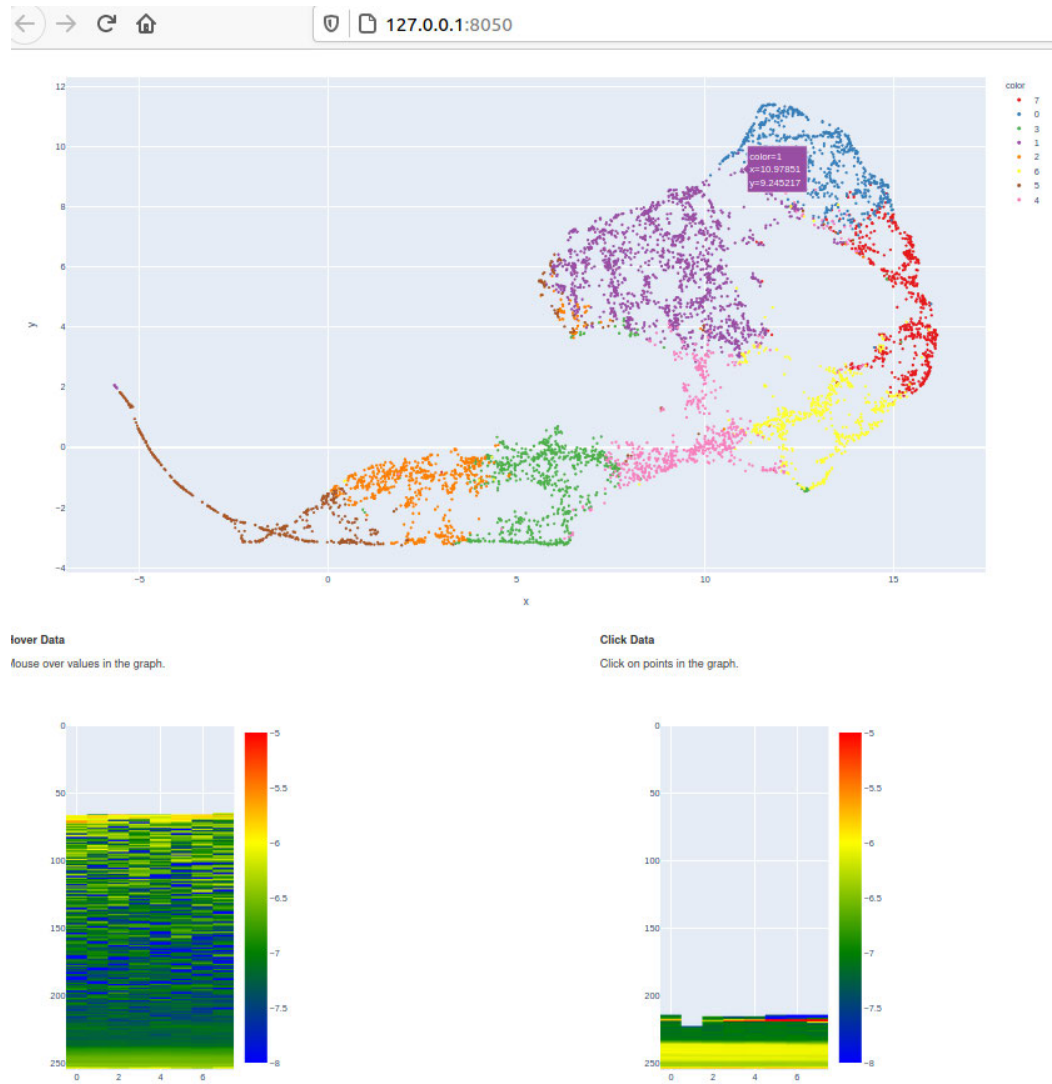


Figure 3.2.: Interactive cluster visualization tool

4. Results

4.1. Clustering Metrics

4.1.1. Clustering on Encoded Data

Table 4.1 provides an overview of different clustering methods and different hyper-parameters performed on the autoencoder feature vectors (encoded data). Agglomerative clustering with single linkage seems to have outperformed others in between-cluster separation, but only by assigning most data to one cluster. The same happens with DBSCAN and agglomerative clustering with average linkage.

The within-cluster similarity is best for the spectral collaborative clustering and the clustering that was used in the previous work.

Spectral clustering has a fair distribution over clusters and has a good within-cluster score. These results show an interesting effect: assigning data to one cluster seems to improve the between-cluster separation, but has no evident influence on within-cluster similarity. The example of agglomerative clustering with single linkage demonstrated that the similarity of the entire test dataset is at 0.72. Only a few methods perform better in this score and distribute data over multiple clusters: original clustering from previous work, spectral clustering, and spectral clustering with collaborative representation.

Several tests with different hyperparameters in landmark spectral clustering show that cityblock metric for distance outperforms cosine.

Density-based method DBSCAN identifies most of the data as one cluster and a small portion as noise.

Table 4.1.: Metrics Result: Clustering on Encoded Data

Clustering Approach	Between-cluster Separation	Within-cluster Similarity	Cluster Amount	Data Distribution
Default	0.11	0.77	8	0.36 0.16 0.12 0.09 0.09 0.07 0.06 0.06
Spectral	0.10	0.74	8	0.26 0.12 0.12 0.11 0.1 0.1 0.1 0.09
Landmark.Spectral.p1000.cosine	0.08	0.71	8	0.4 0.39 0.06 0.05 0.04 0.03 0.02 0.01
Landmark.Spectral.p250.cosine	0.07	0.67	8	0.5 0.29 0.06 0.06 0.04 0.03 0.02 0.0
Landmark.Spectral.p500.cosine	0.08	0.72	8	0.42 0.4 0.05 0.04 0.03 0.03 0.03 0.01
Landmark.Spectral.p250.cityblock	0.11	0.72	8	0.38 0.18 0.14 0.08 0.07 0.06 0.05 0.04
Landmark.Spectral.p500.cityblock	0.11	0.73	8	0.35 0.17 0.16 0.08 0.07 0.07 0.06 0.04
Landmark.Spectral.p1000.cityblock	0.11	0.72	8	0.34 0.19 0.16 0.09 0.08 0.07 0.05 0.03
Spectral.Collaborative	0.08	0.78	8	0.29 0.21 0.13 0.08 0.08 0.08 0.07 0.07
DBSCAN.30	0.24	0.74	2	0.96 0.04
DBSCAN.33	0.28	0.74	2	0.98 0.02
DBSCAN.35	0.29	0.73	2	0.99 0.01
Agglomerative.ward	0.11	0.71	8	0.58 0.11 0.09 0.06 0.06 0.04 0.03 0.03
Agglomerative.complete	0.16	0.63	8	0.8 0.06 0.04 0.03 0.03 0.02 0.01 0.0
Agglomerative.average	0.21	0.68	8	0.91 0.05 0.03 0.01 0.0 0.0 0.0 0.0
Agglomerative.single	0.45	0.73	8	1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0

4.1.2. Clustering on Reduced Data

Tables 4.2 and 4.3 show the metrics for clustering performed on the data reduced by UMAP and PCA respectively. Data for metric calculation is encoded data, not reduced data.

Reducing data to two dimensions improved the within-cluster separation for UMAP but slightly lowered the between-cluster separation index. Overall, UMAP outperformed PCA on all the scores.

Density-based approaches identified many more clusters (for example, 86 for DBCLASD on UMAP reduced data). Smaller clusters have a better similarity within, scoring 0.85, which is the highest result among all sampled methods and data types. DBSCAN algorithm with different parameters assigns approximately a third of data to one cluster on UMAP data, whereas the distribution of data after DBCLASD is more uniformly. For PCA reduced data DBSCAN interprets most of the data as one cluster for the entire range of hyperparameters.

Agglomerative clustering performs best with ward as the dissimilarity measure for PCA reduced data. On the other hand, ward, complete and average linkage have relatively similar performance on UMAP reduced data.

Spectral clustering consistently shows approximately even distribution and has above average scores in both metrics. The Landmark version of this algorithm still performs best with cityblock as a distance measure.

4. Results

Table 4.2.: Metrics Result: Clustering on UMAP Reduced Data

Clustering Approach	Between-cluster Separation	Within-cluster Similarity	Cluster Amount	Data Distribution
Default	0.11	0.77	8	0.36 0.16 0.12 0.09 0.09 0.07 0.06 0.06
Spectral	0.10	0.75	8	0.17 0.17 0.12 0.12 0.11 0.11 0.1 0.1
Landmark.Spectral.p1000.cosine	0.11	0.70	8	0.43 0.35 0.1 0.1 0.01 0.01 0.0 0.0
Landmark.Spectral.p250.cosine	0.11	0.73	8	0.52 0.34 0.1 0.01 0.01 0.01 0.01 0.0
Landmark.Spectral.p500.cosine	0.14	0.74	8	0.79 0.1 0.07 0.02 0.01 0.01 0.0 0.0
Landmark.Spectral.p250.cityblock	0.10	0.75	8	0.18 0.16 0.16 0.13 0.12 0.12 0.1 0.04
Landmark.Spectral.p500.cityblock	0.10	0.75	8	0.18 0.16 0.16 0.13 0.12 0.11 0.1 0.04
Landmark.Spectral.p1000.cityblock	0.10	0.74	8	0.18 0.16 0.16 0.13 0.12 0.12 0.09 0.04
Spectral.Collaborative	0.06	0.70	8	0.22 0.22 0.21 0.14 0.14 0.03 0.02 0.02
DBSCAN.0.2	0.08	0.80	50	0.28 0.09 0.05 0.05 0.05 0.04 0.04 0.04
DBSCAN.0.23	0.09	0.77	26	0.3 0.28 0.1 0.07 0.07 0.03 0.03 0.03
DBSCAN.0.25	0.10	0.77	18	0.55 0.28 0.03 0.03 0.02 0.02 0.02 0.02
DBCLASD	0.06	0.85	86	0.15 0.11 0.06 0.04 0.03 0.03 0.03 0.02
Agglomerative.ward	0.11	0.75	8	0.21 0.19 0.17 0.1 0.1 0.09 0.08 0.06
Agglomerative.complete	0.10	0.75	8	0.22 0.15 0.14 0.13 0.13 0.12 0.08 0.02
Agglomerative.average	0.11	0.75	8	0.22 0.19 0.14 0.14 0.13 0.08 0.08 0.02
Agglomerative.single	0.14	0.74	8	0.99 0.0 0.0 0.0 0.0 0.0 0.0 0.0

Table 4.3.: Metrics Result: Clustering on PCA Reduced Data

Clustering Approach	Between-cluster Separation	Within-cluster Similarity	Cluster Amount	Data Distribution
Default	0.11	0.77	8	0.36 0.16 0.12 0.09 0.09 0.07 0.06 0.06
Spectral	0.08	0.76	8	0.19 0.15 0.14 0.13 0.13 0.1 0.09 0.07
Landmark.Spectral.p1000.cosine	0.08	0.70	8	0.36 0.27 0.13 0.12 0.07 0.02 0.01 0.01
Landmark.Spectral.p250.cosine	0.07	0.77	8	0.25 0.21 0.21 0.17 0.06 0.05 0.03 0.01
Landmark.Spectral.p500.cosine	0.07	0.76	8	0.41 0.18 0.15 0.15 0.1 0.0 0.0 0.0
Landmark.Spectral.p250.cityblock	0.09	0.73	8	0.34 0.24 0.14 0.1 0.06 0.05 0.05 0.02
Landmark.Spectral.p500.cityblock	0.09	0.73	8	0.36 0.22 0.14 0.1 0.07 0.05 0.05 0.02
Landmark.Spectral.p1000.cityblock	0.09	0.73	8	0.36 0.22 0.14 0.1 0.07 0.05 0.05 0.02
Spectral.Collaborative	0.05	0.72	8	0.2 0.18 0.16 0.13 0.09 0.08 0.08 0.08
DBSCAN.2.5	0.15	0.76	20	0.89 0.07 0.01 0.0 0.0
DBSCAN.3	0.19	0.75	7	0.93 0.05 0.0 0.0 0.0 0.0 0.0
DBSCAN.3.5	0.19	0.74	4	0.95 0.04 0.0 0.0
DBCLASD	0.06	0.73	19	0.39 0.15 0.13 0.11 0.11 0.03 0.01 0.01
Agglomerative.ward	0.08	0.74	8	0.32 0.24 0.1 0.1 0.07 0.07 0.06 0.04
Agglomerative.complete	0.11	0.64	8	0.71 0.16 0.05 0.04 0.02 0.02 0.01 0.0
Agglomerative.average	0.15	0.70	8	0.72 0.1 0.08 0.07 0.01 0.01 0.01 0.0
Agglomerative.single	0.48	0.72	8	1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0

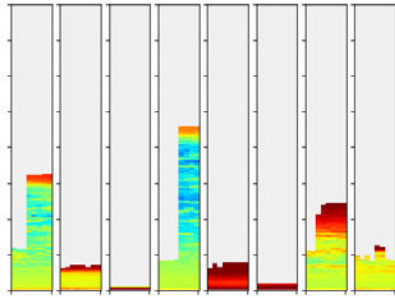
4.2. Visual Representation

4.2.1. Clustering on Encoded Data

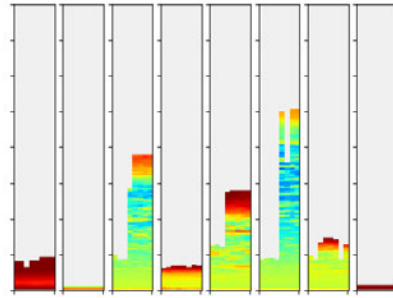
Figure 4.1 represents average data for each cluster and figure 4.2 shows data assigned to each cluster. All presented algorithms have diverse average images. DBSCAN appears to identify images with low content of measurements as noise.

Overall, none of the methods seems to have any obvious pattern of assigning weather data to clusters, as each of them includes images that visually would be better suited for another cluster.

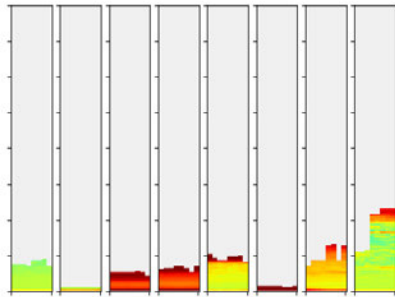
Even though all presented methods can distinguish between images with different amounts of data, clusters are not pure.



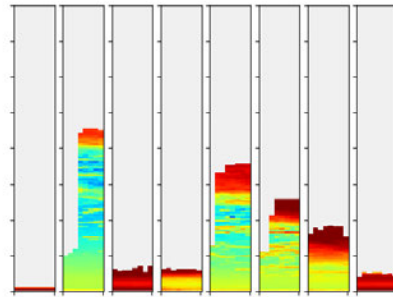
(a) Clustering method used in previous work.



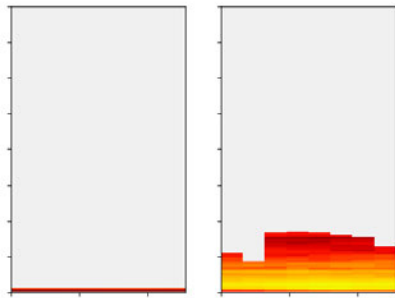
(b) Landmark spectral clustering ($p = 500$, distance metric: cityblock).



(c) Spectral clustering.



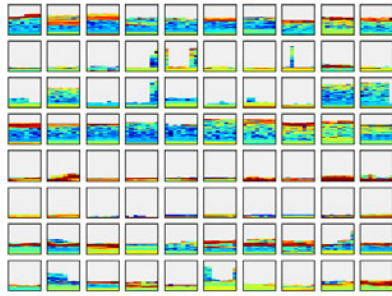
(d) Agglomerative clustering (mode: ward).



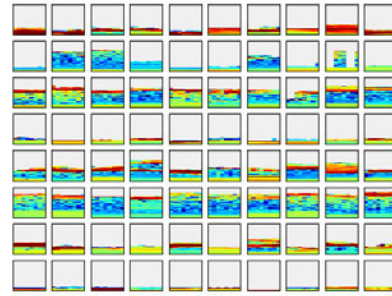
(e) DBSCAN clustering ($eps = 30$).

Figure 4.1.: Average representation of each cluster of clustering methods on encoded data.

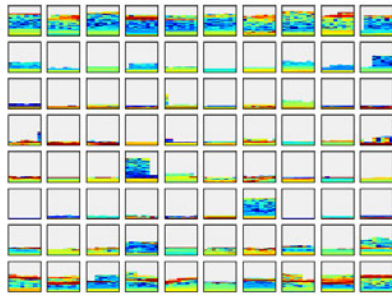
Each rectangle in an image represents one cluster.



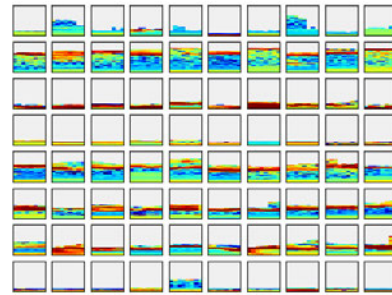
(a) Clustering method used in previous work.



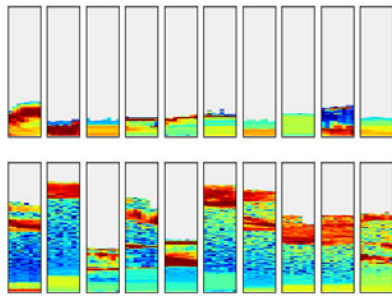
(b) Landmark spectral clustering ($p = 500$, distance metric: cosine).



(c) Spectral clustering.



(d) Agglomerative clustering (mode: ward).



(e) DBSCAN clustering ($eps = 30$).

Figure 4.2.: Data partitioning of clustering methods.

A row represents one cluster. Each rectangle in a row is a random item from all measurements assigned to this cluster. White boxes in a row mean that there were not sufficient items assigned to the particular cluster.

4.2.2. Clustering on Reduced Data

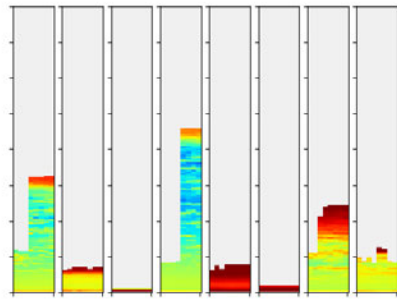
Same as before, figures 4.4 and 4.3 represent the average of each cluster for all methods, and figures 4.6 and 4.5 demonstrate members of each cluster, however all clustering methods were performed on the reduced data.

All clustering methods show variety in the average representation.

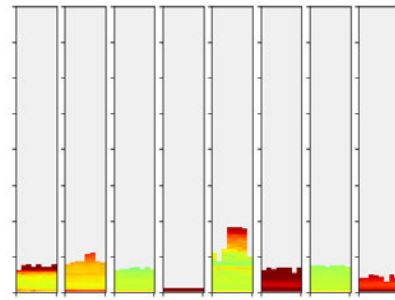
Regarding LIDAR data assigned to clusters, methods performed on UMAP reduced data show a more apparent separation between classes, compared to clustering directly on encoded data and clustering on PCA.

Density-based algorithms have more pure clusters because data is partitioned into much smaller groups. (The figure only provides an overview of the first 8 clusters.)

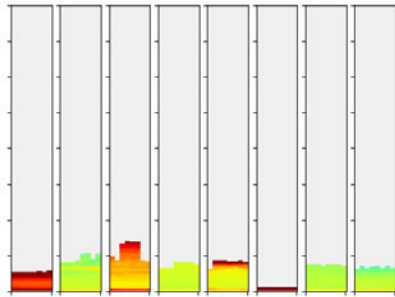
4. Results



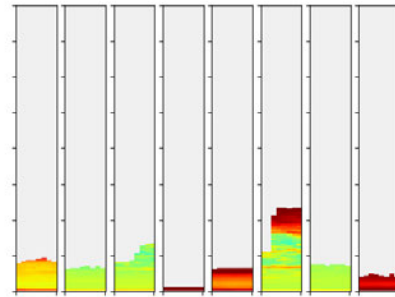
(a) Clustering method used in previous work.



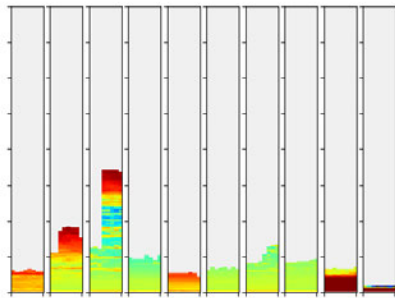
(b) Landmark spectral clustering ($p = 500$, distance metric: cityblock).



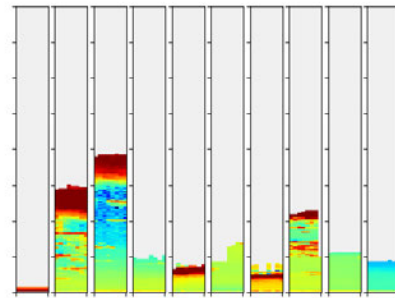
(c) Spectral clustering.



(d) Agglomerative clustering (mode: ward).



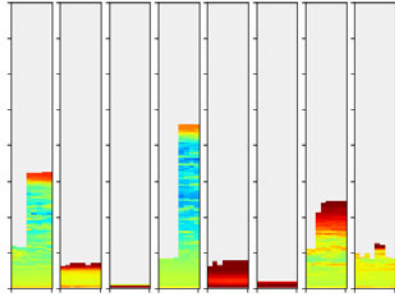
(e) DBSCAN clustering ($eps = 0.23$).



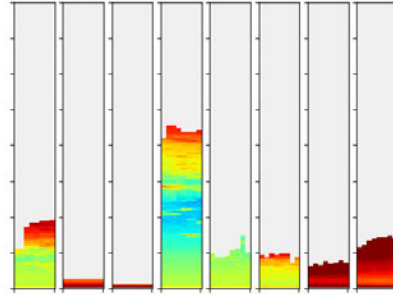
(f) DBCLASD clustering.

Figure 4.3.: Average representation of each cluster of clustering methods on UMAP reduced data.

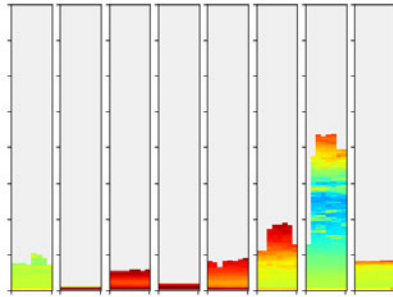
Each rectangle in an image represents one cluster.



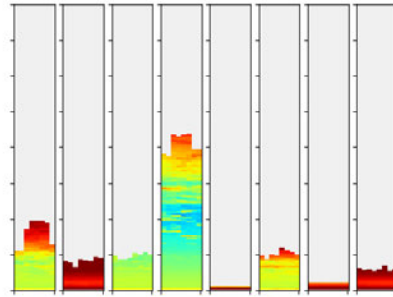
(a) Clustering method used in previous work.



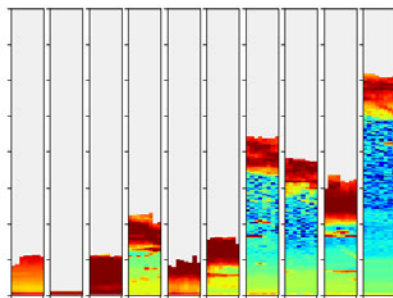
(b) Landmark spectral clustering ($p = 500$, distance metric: cityblock).



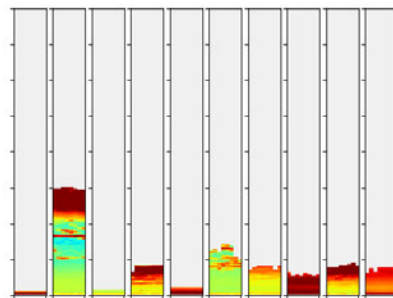
(c) Spectral clustering.



(d) Agglomerative clustering (mode: ward).

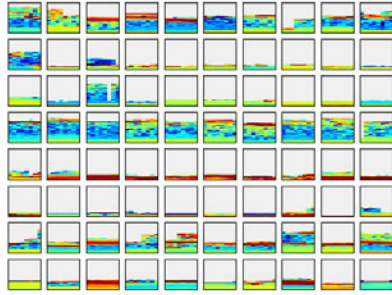


(e) DBSCAN clustering ($eps = 2.5$).

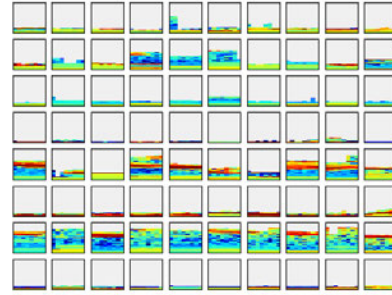


(f) DBCLASD clustering.

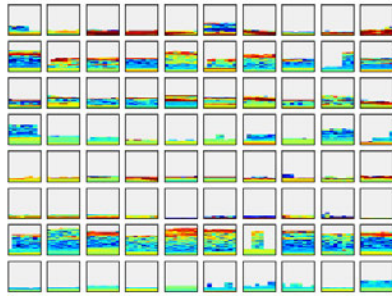
Figure 4.4.: Average representation of each cluster of clustering methods on PCA reduced data.
Each rectangle in an image represents one cluster.



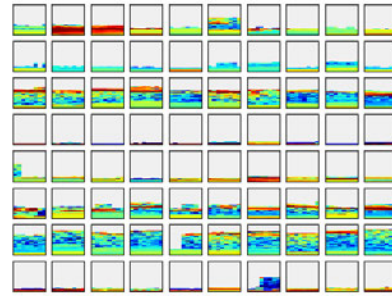
(a) Clustering method used in previous work.



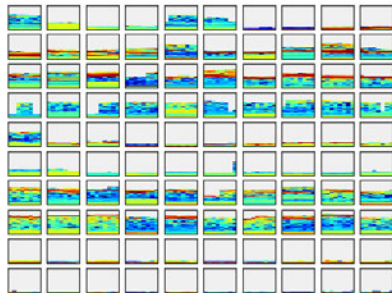
(b) Landmark spectral clustering ($p = 500$, distance metric: cityblock).



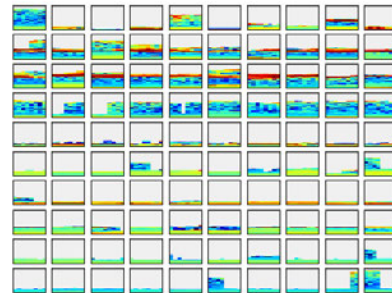
(c) Spectral clustering.



(d) Agglomerative clustering (mode: ward).



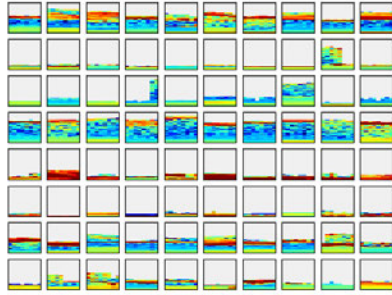
(e) DBSCAN clustering ($eps = 0.23$).



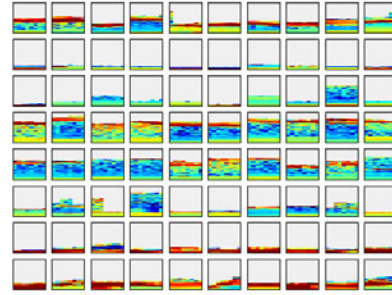
(f) DBCLASD clustering.

Figure 4.5.: Data partitioning of clustering methods on UMAP reduced data.

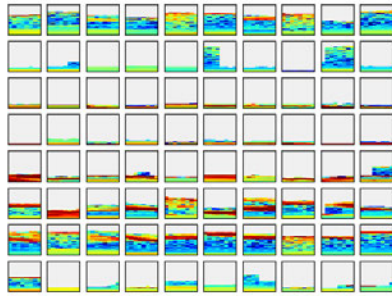
A row represents one cluster. Each rectangle in a row is a random item from all measurements assigned to this cluster. White boxes in a row mean that there were not sufficient items assigned to the particular cluster.



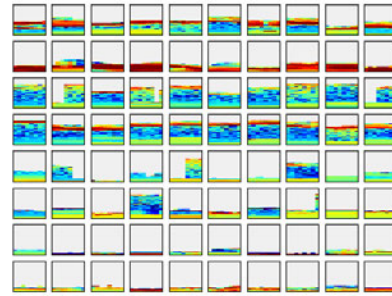
(a) Clustering method used in previous work.



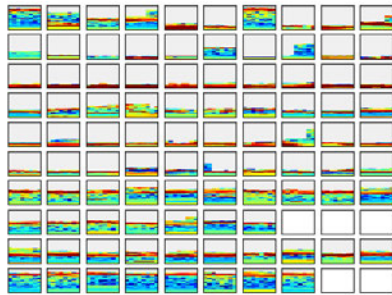
(b) Landmark spectral clustering ($p = 500$, distance metric: cityblock).



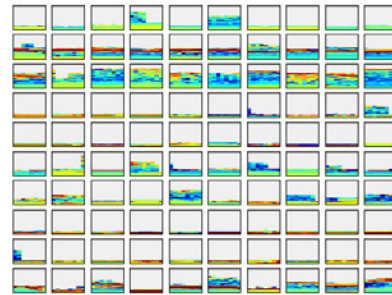
(c) Spectral clustering.



(d) Agglomerative clustering (mode: ward).



(e) DBSCAN clustering ($eps = 2.5$).



(f) DBCLASD clustering.

Figure 4.6.: Data partitioning of clustering methods on PCA reduced data.

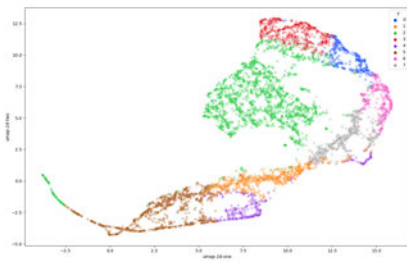
A row represents one cluster. Each rectangle in a row is a random item from all measurements assigned to this cluster. White boxes in a row mean that there were not sufficient items assigned to the particular cluster.

4.3. 2D Visualization of Clusters

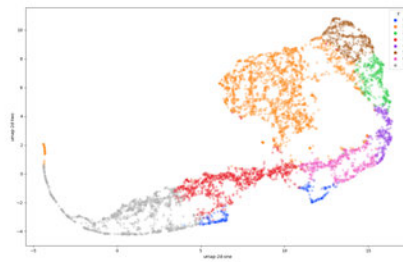
4.3.1. Clustering on Encoded Data

Figure 4.7 visualizes data structure in two dimensions with the help of UMAP.

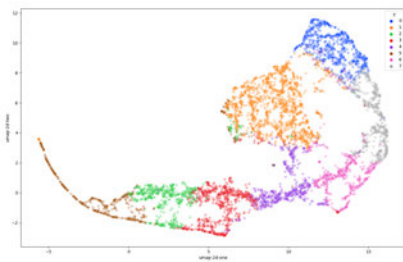
DBSCAN fails to identify separate clusters. All the other clustering algorithms demonstrate clusters with borders close to intuitive.



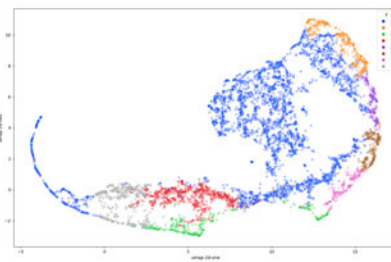
(a) Clustering method used in previous work.



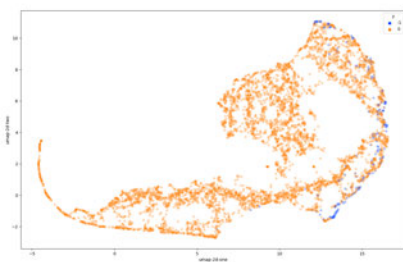
(b) Landmark spectral clustering ($p = 500$, distance metric: cityblock).



(c) Spectral clustering.



(d) Agglomerative clustering (mode: ward).



(e) DBSCAN clustering ($eps = 30$).

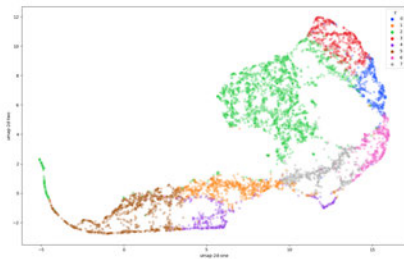
Figure 4.7.: UMAP visualization of clustering methods on encoded data.

Clustering and UMAP were both performed on the encoded data.

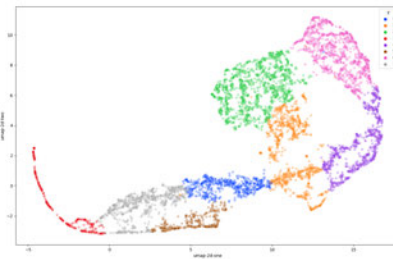
4.3.2. Clustering on Reduced Data

Both agglomerative and spectral clustering on UMAP reduced data and clustering used in the previous work provide comprehensible partitioning. DBSCAN and DBCLASD identify more clusters, and even though the partitioning is not intuitive, 4.5 showed that such small clusters appear purer.

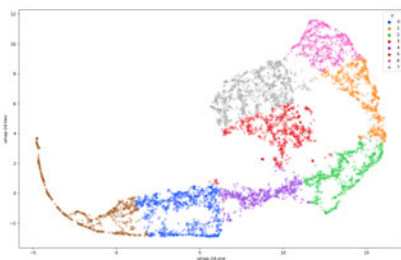
Clustering on PCA reduced data yield similar results, though some of the clusters' borders are fuzzier than clusters assigned on UMAP data. DBSCAN on PCA failed to capture the structure as well, as with UMAP.



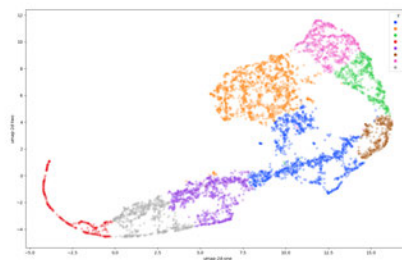
(a) Clustering method used in previous work.



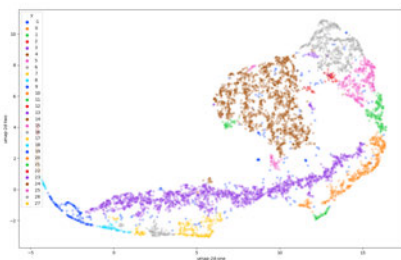
(b) Landmark spectral clustering ($p = 500$, distance metric: cityblock).



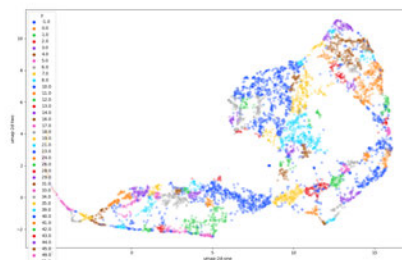
(c) Spectral clustering.



(d) Agglomerative clustering (mode: ward).

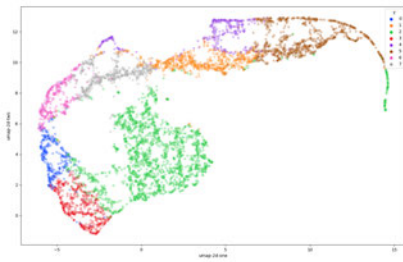


(e) DBSCAN clustering ($eps = 0.23$).

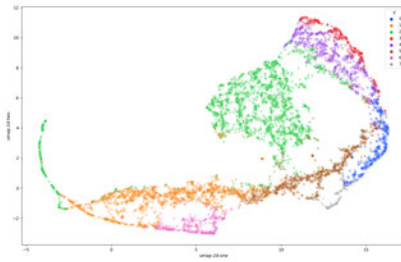


(f) DBCLASD clustering.

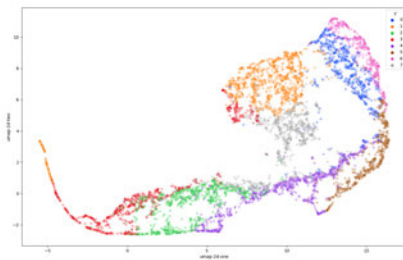
Figure 4.8.: UMAP visualization of clustering methods on the UMAP reduced data.



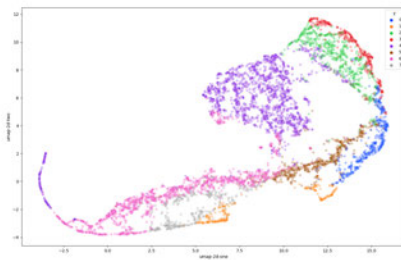
(a) Clustering method used in previous work.



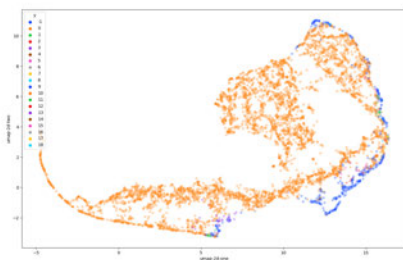
(b) Landmark spectral clustering ($p = 500$, distance metric: cityblock).



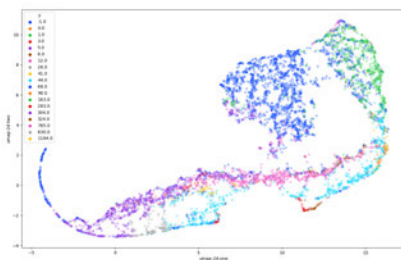
(c) Spectral clustering.



(d) Agglomerative clustering (mode: ward).



(e) DBSCAN clustering ($eps = 2.5$).



(f) DBCLASD clustering.

Figure 4.9.: UMAP visualization of clustering methods on the PCA reduced data.

5. Discussion

The goal of this thesis is to examine how various clustering algorithms compare on LIDAR data. Overall, when applied directly on the feature vector, no algorithms perform significantly better than the original K-Means. That is most likely due to the curse of dimensionality. Even though not all algorithms cluster directly on the distance between objects, similar distance functions are utilized in some steps in all algorithms. According to Aggarwal u. a. (2001), the notion of distance becomes meaningless in high dimensions because the farthest and nearest points approach the ratio of 1, essentially making the distance uniform. This effect is especially pronounced for Euclidean distance measure and less for Manhattan (or cityblock). That is why landmark spectral clustering with cityblock in visual assessment performed better than other methods. High-dimensional data also suffers from a high noise-to-signal ratio.

DBSCAN failed to identify clusters even with different hyperparameters, which could mean that data has a very variable density and is not suited for density-based algorithms.

The methods tested in this part are usually assessed on data with a low number of variables and are not explicitly expected to work well with a high number of features. There are more complicated methods, like CLIQUE, that were designed for high-dimensional data and could potentially work better. However, the high-dimensional methods still rely on the same technologies, like density and distance measure, but for example, search for clusters in subsets of variables. This type of methods is discussed in the next part of the study.

The other explanation for the unsuccessful results could be the data itself. If the autoencoder does not capture the essential structural elements of the data, then no sophisticated method will be able to partition the data well. Most of the data have a high proportion of null-values in the higher part of the image that are translated to 0 before autoencoder; this could potentially disrupt the representation that autoencoder learns by shifting the focus on learning these values and paying less attention to the actual area of interest in the lower part of the image.

When the data is reduced to two dimensions, all algorithms work well on identifying borders of the cluster, which does not necessarily lead to good partitioning of the original data. Double reducing of the data - by autoencoder and UMAP - inevitably leads to the loss of data. Nevertheless, results on UMAP reduced data look better than for the algorithms applied on

encoded vectors. The density-based algorithms look especially promising because they identify smaller clusters with higher within-cluster similarity. It could mean that the reasonable amount of cluster could potentially be higher than eight.

PCA, as expected, achieved worse results than UMAP, mainly because it can only capture the linear dependencies. However, it is still comparable to the clustering on the original data. This indicates that there is not much difference between representing structural dependencies in 32 and 2 dimensions.

Another exciting discovery involves the validity metrics. The between-cluster separation metric does not seem to reflect well the quality of clusters. The score can be artificially boosted by assigning most of the data to one cluster. That is why it could not be used as an optimization metric for algorithms. On the other hand, within-cluster similarity appears to be a valid measure of the quality even though it should be considered carefully, together with the cardinality - number of clusters - and data distribution over clusters.

This work presented the results of clustering exclusively on LIDAR data and has not utilized any other dataset for algorithm quality validation. This has been a deliberate decision to avoid confusion about methods' performance. The clustering algorithms are very susceptible to the data structure, and the results of clustering on other datasets would be misleading.

Overall, the clusters require an assessment by a weather expert. Even though some clusters can look mixed to a general audience, they might identify the same weather events.

Part II.

**High-Dimensional Clustering
Methods**

6. Background

As indicated previously, the feature vectors of LIDAR data consist of high dimensions. All the methods mentioned above are not specifically tailored for handling this type of data. There exists a separate class of algorithms developed for partitioning data with a large number of variables.

The most utilized group of such clustering algorithms is called subspace clustering. The main idea behind it is that irrelevant features influence the appearance of clusters in the full-dimensional space, as explained in Kriegel u. a. (2009). These features could be masking the actual partitioning by introducing noise. The goal of subspace clustering is to identify relevant features. For each cluster, a different subset of features may be relevant. The subspace method differs from PCA and other dimensionality reduction methods since it keeps the subset of existing dimensions instead of creating new ones. The similarity between objects is assessed by conventional measures such as distance and density. Clusters may overlap since an object can be clustered with a different subset of features in multiple groups.

Subspace search is further divided into two categories, bottom-up and top-down approach. Han u. a. (2011) explains that the bottom-up approach first searches for clusters in lower dimensions and only proceeds by including more dimensions if the clusters are identified. CLIQUE by Agrawal u. a. (1998) is an example of such an algorithm. CLIQUE is a grid-based and density-based algorithm that produces deterministic results and does not assume any canonical distribution for input data. CLIQUE has several modification algorithms, like ENCLUS or MAFLA, and all of them rely on grids to calculate clusters. However, using grids leads to a significant disadvantage of all these methods. Grid-based algorithms massively depend on the positioning of the grids. Some neighboring objects can get separated by grid boundary, which means that some clusters can be missed if they are shaped inadequately. Another algorithm is SUBCLU, presented by Kailing u. a. (2004), which is a density-based cluster variant of DBSCAN with the same user-defined parameters. It can detect clusters of arbitrary size and shape. This paper argued that SUBCLU outperforms CLIQUE and is a more time-efficient algorithm. CLIQUE results are not calculated sometimes, even up to several days for more

than 25 dimensions; therefore it will not be considered further. SUBCLU algorithm will be analyzed for LIDAR data.

Top-down approaches start with full space and search smaller subspaces recursively. Kriegel u. a. (2008) explains that most of these approaches rely on "locality assumption", meaning that the subspace is derived from the local neighborhood. The arising problem is that neighborhood might contain much noise. This is a direct effect of the curse of dimensionality. PROCLUS Aggarwal u. a. (1999) is one of such algorithms. It is a k-medoid-like method that starts by creating k clusters and then proceeds to refine the subspace. For each medoid, the local neighborhood is considered to identify a subspace by minimizing the standard deviation of the objects' distances to medoid in each dimension. By each iteration, medoids are replaced by new ones in case if it improves cluster quality. PROCLUS is also sometimes referenced as a projected clustering method, but literature is not consistent about terms "subspace" and "projected" clustering. PROCLUS requires a number of clusters k and an average dimensionality l . Average dimensionality is a very sensitive parameter. An unfortunately chosen value can result in algorithm missing clusters.

High-dimensional clustering algorithms also include classes like correlation-based clustering algorithms (or biclustering) that cluster objects and attributes simultaneously. The resulted clusters have a small number of objects and a small number of attributes, as explained by Han u. a. (2011), which is not the desired outcome for LIDAR data; ergo, biclustering algorithms are not a good fit for this analysis.

Projection-based clustering has been explored in the previous chapter. Its main idea consists of projecting the data into two dimensions and clustering with some algorithm.

Another exciting direction is concept clustering. It is explained well in Kaufman (2012). This type of clustering utilizes not only the distance of objects but also some type of background knowledge, which makes clustering more meaningful for the user. Background knowledge could be defined attribute ranges, hierarchies over attribute values, implicative rules of constraints of one variable on others, rules for constructing new attributes, rules for ranking variables according to their relevance, as described in Stepp (1987). Since this knowledge is not available for the LIDAR dataset, methods from conceptual clustering will not be considered.

7. Methods

Data overview and analysis procedure is described in 3. The following section describes the hyperparameter selection and the used software.

7.1. Parameter Settings

PROCLUS cluster amount was set to 8, whereas the average dimensionality is tested in range between 5 and 15.

SUBCLU algorithm requires same parameters as DBSCAN that is why same parameters have been utilized, as for the DBSCAN: $MinPts = 33$, eps set to range of values [27, 30, 33, 36].

Since distance might change with fewer dimensions, SUBCLU was also tested with lower $MinPts$ values, for example 10, but that had no detectable effect on the quality of results.

7.2. Software

The algorithms' implementation is provided by an R package "subspace".

Source: <https://rdr.io/cran/subspace/>.

Partition creation is performed in an R script and the result analysis is generated with the help of Python.

8. Results

8.1. Clustering Metrics

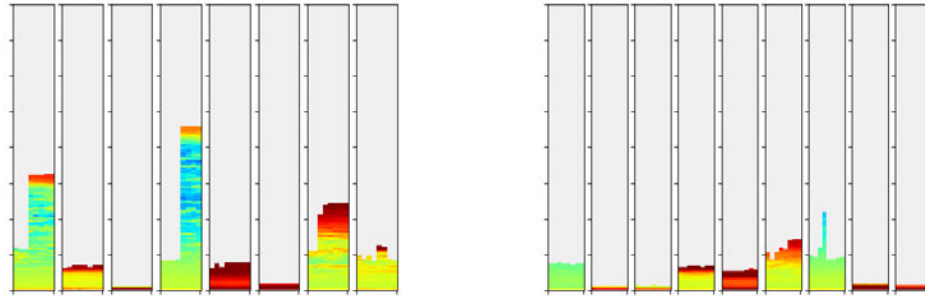
The table 8.1 presents metric results. The result for both methods and different hyperparameter settings are consistent in range 0.07-0.08 for between-cluster separation (these numbers are lower than for classical clustering algorithms applied directly on encoded data). The within-cluster similarity vary between 0.76 and 0.81, which is comparable to classical algorithms. Both algorithm identify a signifact part of data as noise (up to 58 percent by Subclu with epsilon parameter set to 36).

Table 8.1.: Metrics Result: High-Dimensional Clustering on Encoded Data

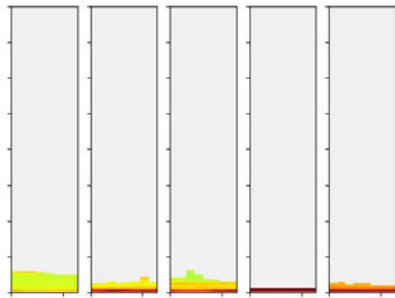
Clustering Approach	Between-cluster Separation	Within-cluster Similarity	Cluster Amount	Data Distribution	Noise Part
Proclus.Dim.5	0.07	0.77	8+noise	0.48 0.1 0.1 0.09 0.09 0.07 0.03 0.03 0.0	0.48
Proclus.Dim.6	0.07	0.78	8+noise	0.4 0.23 0.07 0.07 0.07 0.06 0.04 0.04 0.02	0.40
Proclus.Dim.7	0.07	0.79	8+noise	0.24 0.2 0.18 0.16 0.06 0.06 0.05 0.04 0.02	0.16
Proclus.Dim.8	0.07	0.78	8+noise	0.35 0.27 0.12 0.11 0.07 0.06 0.01 0.01 0.01	0.35
Proclus.Dim.9	0.08	0.78	8+noise	0.51 0.23 0.07 0.06 0.04 0.04 0.03 0.02 0.0	0.51
Proclus.Dim.10	0.08	0.79	8+noise	0.25 0.22 0.18 0.09 0.08 0.07 0.05 0.04 0.02	0.25
Proclus.Dim.11	0.07	0.79	8+noise	0.3 0.3 0.16 0.06 0.05 0.04 0.04 0.03 0.02	0.30
Proclus.Dim.12	0.08	0.78	8+noise	0.38 0.15 0.12 0.1 0.1 0.05 0.05 0.04 0.02	0.38
Proclus.Dim.13	0.07	0.81	8+noise	0.25 0.22 0.11 0.09 0.09 0.07 0.07 0.06 0.03	0.25
Proclus.Dim.14	0.08	0.79	8+noise	0.42 0.2 0.12 0.06 0.05 0.05 0.05 0.03 0.02	0.42
Proclus.Dim.15	0.08	0.79	8+noise	0.41 0.19 0.13 0.08 0.05 0.04 0.03 0.03 0.03	0.19
Subclu.Eps.27	0.07	0.76	4+noise	0.38 0.34 0.12 0.11 0.06	0.38
Subclu.Eps.30	0.08	0.76	4+noise	0.42 0.36 0.12 0.06 0.04	0.36
Subclu.Eps.33	0.07	0.77	4+noise	0.39 0.23 0.16 0.12 0.1	0.39
Subclu.Eps.36	0.08	0.76	4+noise	0.58 0.2 0.1 0.06 0.06	0.58

8.2. Visual Representation

Same as in the previous part, figures 8.1 and 8.2 represent average data and examples of images assigned to each cluster. To keep a clear overview, only one parameter setting has been displayed for each cluster method. For visual assessment of other results please refer to supplementary data. The chosen parameters present best visual results. PROCLUS has several groups, that seem to exhibit consistent pattern, whereas other groups look rather messy. SUBCLU clustering, on the other hand, does not have a clear pattern of assigning data.



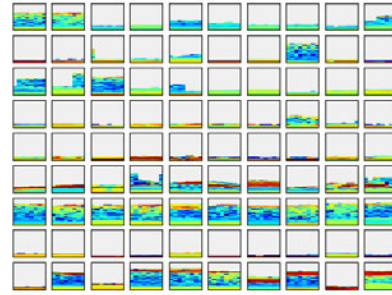
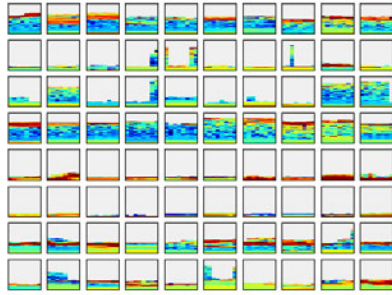
(a) Clustering method used in previous work. (b) PROCLUS clustering (average dimension amount = 14).



(c) SUBCLU clustering (epsilon = 36).

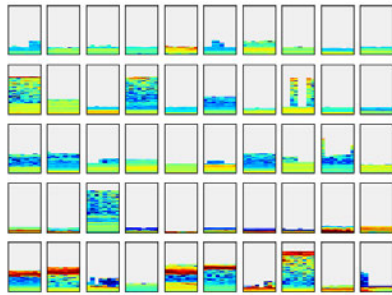
Figure 8.1.: Average representation of high-dimensional clustering methods and the clustering method from previous work on encoded data.

Each rectangle in an image represents one cluster. For PROCLUS and SUBCLU the last rectangle represents data marked as noise.



(a) Clustering method used in previous work.

(b) PROCLUS clustering (average dimension amount = 14).



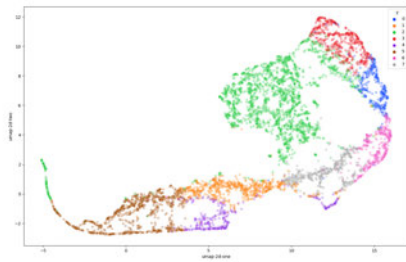
(c) SUBCLU clustering (epsilon = 36).

Figure 8.2.: Data partitioning of high-dimensional clustering methods and the clustering method from previous work on encoded data.

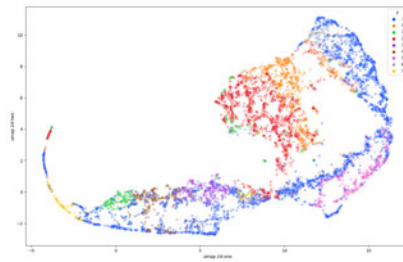
A row represents one cluster. Each rectangle in a row is a random item from all measurements assigned to this cluster. White boxes in a row mean that there were not sufficient items assigned to the particular cluster. For PROCLUS and SUBCLU the last row represents noise.

8.3. 2D Visualization

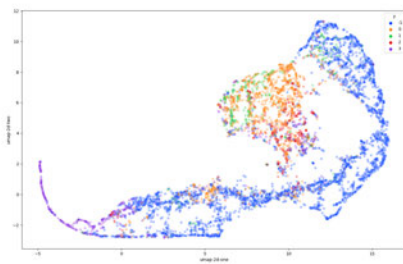
Both high-dimensional algorithms do not have clear cut borders for each group. As mentioned previously, most of the data is treated as noise.



(a) Clustering method used in previous work.



(b) PROCLUS clustering (average dimension amount = 14).



(c) SUBCLU clustering (epsilon = 36).

Figure 8.3.: UMAP visualization of of high-dimensional clustering methods and the clustering method from previous work on encoded data.

9. Discussion

This part of the thesis analysed the algorithms, that were specifically designed to treat high-dimensional data. Both algorithms belong to the class of subspace search algorithms, meaning that constructed clusters took into account only some features available from 32-dimensional feature vector. The results clearly demonstrate that high-dimensional clustering does not outperform the classical algorithms and can even be considered inferior to them.

It could mean, that most of the features carry important information and can not be disregarded for partitioning. However, factor analysis showed that feature vectors contain variables that are in correlation and therefore could be reduced to lower dimensions. It is possible, that meaningful combinations of features were simply not discovered by the algorithms.

PROCLUS is known to perform poorly if different subspace clusters consist of different dimensionality. If LIDAR data has partitions with such characteristics, it could account for unsatisfactory performance.

SUBCLU is a high-dimensional version of DBSCAN. The results could be explained by a poor choice of parameter, since hyperparameter selection for DBSCAN is known to be complicated and not straightforward. On the other hand, several parameter options were tested and failed to produce qualitative partitioning. Previous discussion mentioned the idea that variable density of the data could account for inadequate results. This argumentation holds for the high-dimensional clustering as well.

Part III.

Clustering Original Data

10. Motivation

Previous results indicated that multiple dimensions produced by autoencoder might be redundant. Factor analysis showed correlation between variables and it could possibly mean that the particular model of autoencoder used in this study does not contribute to quality of clustering results. To be able to better analyse the merit of this hypothesis, this additional part investigates whether data reduced to 2 dimensions only by UMAP would yield better or comparable results.

11. Results

Table 11.1 shows the metrics results on clustering data reduced directly to 2 dimensions by UMAP, skipping the dimensionality reduction by autoencoder. An interesting artifact can be observed, where different index shows much higher results than in the previous clustering approaches (note the difference between default clustering observations and others).

Overall, different clustering methods seem to have little impact on metrics, but differ in data distribution between identified clusters.

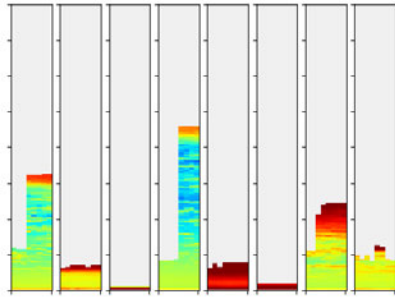
11.1. Clustering Metrics

Table 11.1.: Metrics Result: Clustering on UMAP Reduced Original Data

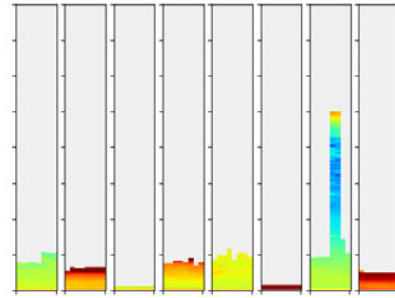
Clustering Approach	Between-cluster Separation	Within-cluster Similarity	Cluster Amount	Data Distribution
Default	0.11	0.77	8	0.36 0.16 0.12 0.09 0.09 0.07 0.06 0.06
Spectral	0.95	0.03	8	0.18 0.14 0.13 0.13 0.12 0.11 0.11 0.08
Landmark.Spectral.p1000.cosine	0.95	0.03	8	0.42 0.18 0.13 0.12 0.09 0.04 0.01 0.0
Landmark.Spectral.p250.cosine	0.95	0.03	8	0.48 0.21 0.15 0.1 0.04 0.01 0.01 0.0
Landmark.Spectral.p500.cosine	0.95	0.03	8	0.37 0.18 0.16 0.09 0.09 0.06 0.03 0.01
Landmark.Spectral.p250.cityblock	0.95	0.03	8	0.24 0.15 0.13 0.12 0.11 0.09 0.08 0.08
Landmark.Spectral.p500.cityblock	0.95	0.03	8	0.24 0.17 0.13 0.11 0.11 0.08 0.08 0.08
Landmark.Spectral.p1000.cityblock	0.95	0.03	8	0.25 0.19 0.12 0.12 0.11 0.09 0.08 0.05
Spectral.Collaborative	0.95	0.02	8	0.31 0.15 0.13 0.13 0.13 0.06 0.06 0.02
DBCLASD	0.94	0.02	109	0.14 0.06 0.05 0.03 0.03 0.03 0.02 0.02 0.02
Agglomerative.ward	0.95	0.03	8	0.21 0.17 0.13 0.13 0.12 0.12 0.06 0.06
Agglomerative.complete	0.95	0.03	8	0.22 0.17 0.13 0.13 0.12 0.11 0.06 0.04
Agglomerative.average	0.95	0.03	8	0.26 0.2 0.18 0.13 0.08 0.06 0.05 0.04
Agglomerative.single	0.94	0.03	8	1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0

11.2. Visual Representation

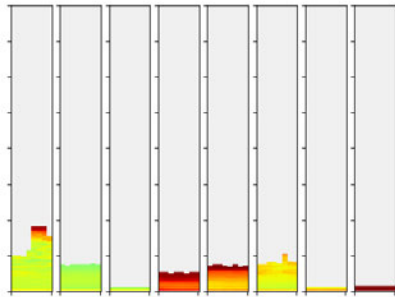
Figures 11.1 and 11.2 do not show significant improvement over the clustering done on data reduced by autoencoder. On the other hand, the new results also do not appear to be worse. The reduction techniques, whether autoencoder or UMAP, seem to produce comparable cluster distribution.



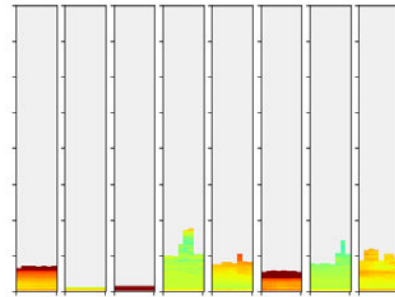
(a) Clustering method used in previous work.



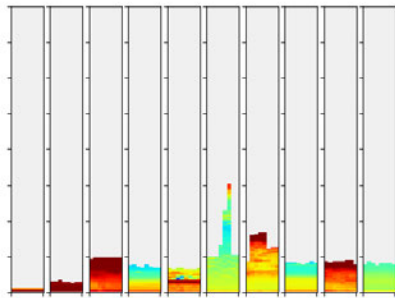
(b) Landmark spectral clustering ($p = 500$, distance metric: cityblock).



(c) Spectral clustering.



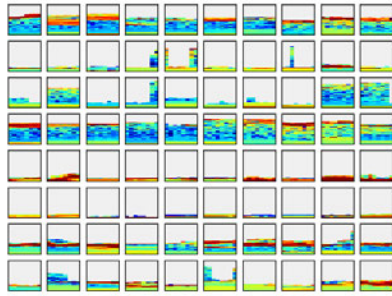
(d) Agglomerative clustering (mode: ward).



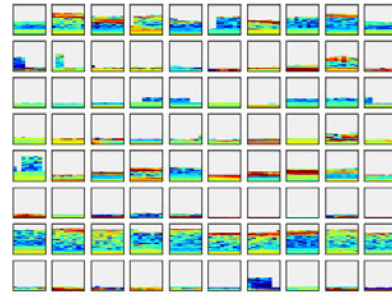
(e) DBCLASD clustering.

Figure 11.1.: Average representation of each cluster of clustering methods on original data reduced to 2D by UMAP.

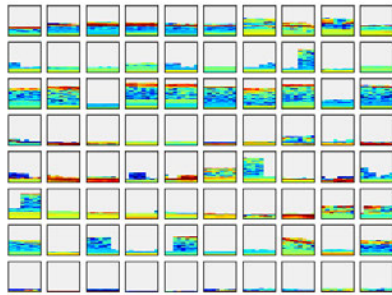
Each rectangle in an image represents one cluster.



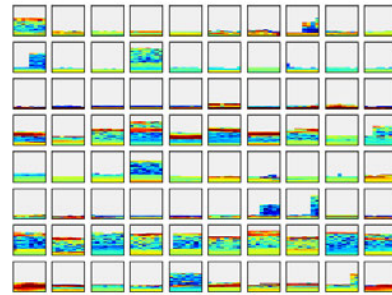
(a) Clustering method used in previous work.



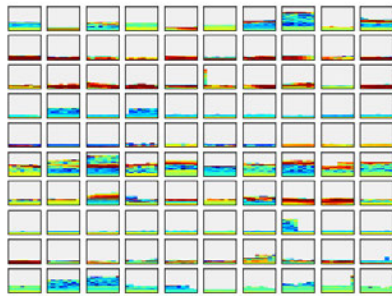
(b) Landmark spectral clustering ($p = 500$, distance metric: cosine).



(c) Spectral clustering.



(d) Agglomerative clustering (mode: ward).

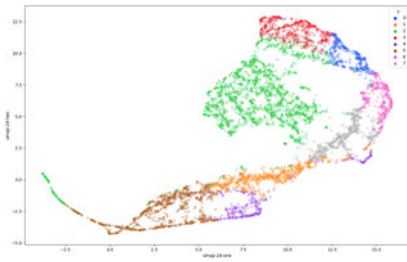


(e) DBCLASD clustering.

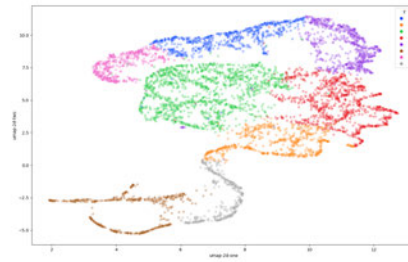
Figure 11.2.: Data partitioning of clustering methods on original data reduced to 2D by UMAP. A row represents one cluster. Each rectangle in a row is a random item from all measurements assigned to this cluster. White boxes in a row mean that there were not sufficient items assigned to the particular cluster.

11.3. 2D Visualization of Clusters

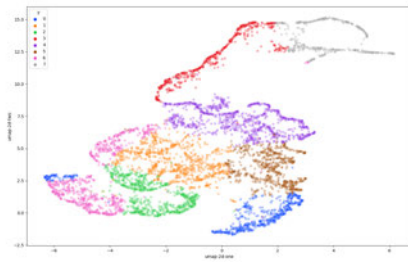
Data reduced directly by UMAP produced different shape of data distribution with some of the data groups being more clearly separated from others. Though it can be argued, that the data in the middle of the plot has less clear borders between possible clusters.



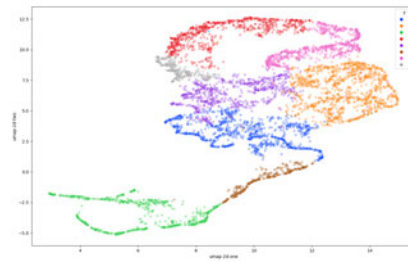
(a) Clustering method used in previous work.



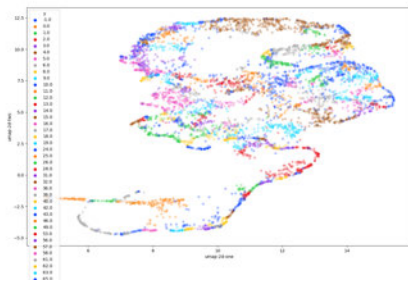
(b) Landmark spectral clustering ($p = 500$, distance metric: cityblock).



(c) Spectral clustering.



(d) Agglomerative clustering (mode: ward).



(e) DBCLASD clustering.

Figure 11.3.: UMAP visualization of clustering methods on original data reduced to 2D by UMAP.

12. Conclusion

The purpose of this thesis is to compare the performance of several clustering algorithms on the LIDAR data, that has been previously reduced to a feature vector by the preceding work. The feature vector is an encoding produced by an autoencoder and consists of 32 dimensions. The original work only explored the partitioning done with K-Means, whereas this study explored different classes of clustering algorithms, like hierarchical, spectral, density clustering. The second part investigates further methods, that are developed specifically for high-dimensional data and belong to subspace clustering techniques. Several Python scripts were created to enable automated analysis process. These scripts can be used by subsequent works on the topic for a fast clustering evaluation.

Overall, this work shows that there is no method that outperformed others and should be further considered and optimized. Only some of the methods, like density based clustering on UMAP reduced data, had a visually better separation than the clustering used in the previous work. On the other hand, these algorithms produced 20 to 80 groups, which is not necessarily a meaningful partition.

All of the inspected methods were visualized in a 2D space with the help of UMAP. To make exploration of such visualization easier, an additional tool was created that allows fast comparison of different slices by hovering over them. The quick manual assessment of the 2D map revealed that sometimes visually very similar objects are found further apart and dissimilar objects are plotted very close to each other. However, on the first look data groups have similarities within. It would be important to explore this map further to learn how valid the 2D representation of the data is, and whether it should be considered a suitable visualization of objects.

I believe, that the most reasonable direction for future studies on the topic would be exploring different hyperparameters and types of autoencoder models and finding an appropriate way of dealing with null values in the top part of each slice, that might be focusing attention on the autoencoder on the wrong fragment of data. To appreciate the importance of different hyperparameters of autoencoder and their influence on the clustering result, please consider following observation: when the autoencoder was trained on data normalized with different

max and min backscatter values (with the actual maximum and minimum and not adjusted values like used in the preceding and this work), the clustering results appeared random and most of the algorithms classified data to one cluster. This was probably due to the very bad reconstruction after decoder, where output did not resemble input, which could mean that feature vectors were not representative of the data. On the other hand, autoencoder model and settings, as described in the methodology, had a much better reconstruction of the image, but still had potential to improve.

Part 3 also provided reasons to believe that the problem lies in the current architecture of an autoencoder. Clustering that was performed on the original data reduced directly by UMAP to 2 dimensions had similar results, based on the visual assessment. That might indicate that the current autoencoder model does not contribute significantly to the extracting of meaningful features.

One of the possible modification, that could be done to autoencoder architecture, is to include self-attention layers, introduced by Bahdanau u. a. (2014). Attention mechanism allows to better handle long-range dependencies. A self-attention layer complements convolutional layers and helps to create a better representation of an object in a feature vector. A new paper Chen u. a. (2021) describes the approach to clustering with the help of autoencoder that combines convolutional and attention layers, and the feature vectors are then clustered with the subspace clustering algorithm. The paper showed promising results and should therefore be considered as a starting point for research in that direction.

Bibliography

- [Abboud u. a. 2019] ABBOUD, Amir ; COHEN-ADDAD, Vincent ; HOUDROUGÉ, Hussein: Sub-quadratic high-dimensional hierarchical clustering. In: *NeurIPS'19-33rd Conference on Neural Information Processing Systems*, 2019
- [Abdi und Williams 2010] ABDI, Hervé ; WILLIAMS, Lynne J.: Principal component analysis. In: *Wiley interdisciplinary reviews: computational statistics* 2 (2010), Nr. 4, S. 433–459
- [Aggarwal u. a. 2001] AGGARWAL, Charu C. ; HINNEBURG, Alexander ; KEIM, Daniel A.: On the surprising behavior of distance metrics in high dimensional space. In: *International conference on database theory* Springer (Veranst.), 2001, S. 420–434
- [Aggarwal u. a. 1999] AGGARWAL, Charu C. ; WOLF, Joel L. ; YU, Philip S. ; PROCOPIUC, Cecilia ; PARK, Jong S.: Fast algorithms for projected clustering. In: *ACM SIGMoD Record* 28 (1999), Nr. 2, S. 61–72
- [Agrawal u. a. 1998] AGRAWAL, Rakesh ; GEHRKE, Johannes ; GUNOPULOS, Dimitrios ; RAGHAVAN, Prabhakar: Automatic subspace clustering of high dimensional data for data mining applications. In: *Proceedings of the 1998 ACM SIGMOD international conference on Management of data*, 1998, S. 94–105
- [Armstrong und Soelberg 1968] ARMSTRONG, J S. ; SOELBERG, Peer: On the interpretation of factor analysis. In: *Psychological Bulletin* 70 (1968), Nr. 5, S. 361
- [Bahdanau u. a. 2014] BAHDANAU, Dzmitry ; CHO, Kyunghyun ; BENGIO, Yoshua: Neural machine translation by jointly learning to align and translate. In: *arXiv preprint arXiv:1409.0473* (2014)
- [Becht u. a. 2019] BECHT, Etienne ; MCINNES, Leland ; HEALY, John ; DUTERTRE, Charles-Antoine ; KWOK, Immanuel W. ; NG, Lai G. ; GINHOUX, Florent ; NEWELL, Evan W.: Dimensionality reduction for visualizing single-cell data using UMAP. In: *Nature biotechnology* 37 (2019), Nr. 1, S. 38–44

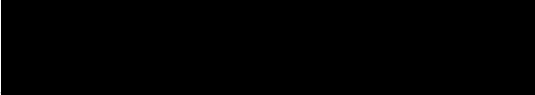
- [Chen und Cai 2011] CHEN, Xinlei ; CAI, Deng: Large scale spectral clustering with landmark-based representation. In: *Proceedings of the AAAI Conference on Artificial Intelligence* Bd. 25, 2011
- [Chen u. a. 2021] CHEN, Zhengfan ; DING, Shifei ; HOU, Haiwei: A novel self-attention deep subspace clustering. In: *International Journal of Machine Learning and Cybernetics* (2021), S. 1–11
- [Davies und Bouldin 1979] DAVIES, David L. ; BOULDIN, Donald W.: A cluster separation measure. In: *IEEE transactions on pattern analysis and machine intelligence* (1979), Nr. 2, S. 224–227
- [Ding und He 2004] DING, Chris ; HE, Xiaofeng: K-means clustering via principal component analysis. In: *Proceedings of the twenty-first international conference on Machine learning*, 2004, S. 29
- [Donath und Hoffman 1973] DONATH, WE ; HOFFMAN, AJ: Lower Bounds for the Partitioning of Graphs. In: *IBM Journal of Research and Development* 17 (1973), Nr. 5, S. 420–425
- [Dubey und Choubey 2017] DUBEY, Ankita ; CHOUBEY, APDA: A systematic review on k-means clustering techniques. In: *Int J Sci Res Eng Technol (IJSRET, ISSN 2278–0882)* 6 (2017), Nr. 6
- [Dunn 1974] DUNN, Joseph C.: Well-separated clusters and optimal fuzzy partitions. In: *Journal of cybernetics* 4 (1974), Nr. 1, S. 95–104
- [Dziuban und Shirkey 1974] DZIUBAN, Charles D. ; SHIRKEY, Edwin C.: When is a correlation matrix appropriate for factor analysis? Some decision rules. In: *Psychological bulletin* 81 (1974), Nr. 6, S. 358
- [Ester u. a. 1996] ESTER, Martin ; KRIEGEL, Hans-Peter ; SANDER, Jörg ; XU, Xiaowei u. a.: A density-based algorithm for discovering clusters in large spatial databases with noise. In: *Kdd* Bd. 96, 1996, S. 226–231
- [Franck u. a. 2004] FRANCK, Pierre ; CAMERON, E ; GOOD, G ; RASPLUS, J-Y ; OLDROYD, BP: Nest architecture and genetic differentiation in a species complex of Australian stingless bees. In: *Molecular Ecology* 13 (2004), Nr. 8, S. 2317–2331
- [G. Lufft Mess- und Regeltechnik GmbH] G. Lufft Mess- und Regeltechnik GmbH (Veranst.): *Manual Ceilometer CHM 15k „NIMBUS“*

- [Gandraß 2019] GANDRASS, Niels: *Merkmalsextraktion durch Methoden des unüberwachten maschinellen Lernens zur Klassifikation von Aerosol-Rückstreuprofilen aus LIDAR-Ceilometern*. 2019. – URL <https://kataloge.uni-hamburg.de/DB=2/XMLPRS=N/PPN?PPN=1666382442>
- [Gorawski und Malczok 2006] GORAWSKI, Marcin ; MALCZOK, Rafal: AEC algorithm: a heuristic approach to calculating density-based clustering Eps parameter. In: *International Conference on Advances in Information Systems* Springer (Veranst.), 2006, S. 90–99
- [Halkidi u. a. 2015] HALKIDI, MMCC ; VAZIRGIANNIS, M ; HENNIG, C: Method-independent indices for cluster validation and estimating the number of clusters. In: *Handbook of cluster analysis* 26 (2015), S. 595–618
- [Han u. a. 2011] HAN, Jiawei ; KAMBER, Micheline ; PEI, Jian: Data mining concepts and techniques third edition. In: *The Morgan Kaufmann Series in Data Management Systems* 5 (2011), Nr. 4, S. 545
- [Hennig 2017] HENNIG, Christian: Cluster validation by measurement of clustering characteristics relevant to the user. In: *arXiv preprint arXiv:1703.09282* (2017)
- [Hennig und Meila 2015] HENNIG, Christian ; MEILA, Marina: Cluster analysis: an overview. In: *Handbook of cluster analysis* 1 (2015)
- [Hennig u. a. 2015] HENNIG, Christian ; MEILA, Marina ; MURTAGH, Fionn ; ROCCI, Roberto: *Handbook of cluster analysis*. CRC Press, 2015
- [Hinneburg und Gabriel 2007] HINNEBURG, Alexander ; GABRIEL, Hans-Henning: Denclue 2.0: Fast clustering based on kernel density estimation. In: *International symposium on intelligent data analysis* Springer (Veranst.), 2007, S. 70–80
- [Hinneburg u. a. 1998] HINNEBURG, Alexander ; KEIM, Daniel A. u. a.: An efficient approach to clustering in large multimedia databases with noise. In: *KDD* Bd. 98, 1998, S. 58–65
- [Illingworth u. a. 2019] ILLINGWORTH, Anthony J. ; CIMINI, D ; HAEFELE, A ; HAEFFELIN, M ; HERVO, M ; KOTTHAUS, S ; LÖHNERT, U ; MARTINET, P ; MATTIS, I ; O’CONNOR, EJ u. a.: How can existing ground-based profiling instruments improve European weather forecasts? In: *Bulletin of the American Meteorological Society* 100 (2019), Nr. 4, S. 605–619
- [Inc. 2015] INC., Plotly T.: *Collaborative data science*. 2015. – URL <https://plot.ly>

- [Jolliffe 1982] JOLLIFFE, Ian T.: A note on the use of principal components in regression. In: *Journal of the Royal Statistical Society: Series C (Applied Statistics)* 31 (1982), Nr. 3, S. 300–303
- [Kailing u. a. 2004] KAILING, Karin ; KRIEGEL, Hans-Peter ; KRÖGER, Peer: Density-connected subspace clustering for high-dimensional data. In: *Proceedings of the 2004 SIAM international conference on data mining* SIAM (Veranst.), 2004, S. 246–256
- [Kaiser 1970] KAISER, Henry F.: A second generation little jiffy. In: *Psychometrika* 35 (1970), Nr. 4, S. 401–415
- [Kaufman 2012] KAUFMAN, Kenneth A.: *Conceptual Clustering*. S. 738–740. In: SEEL, Norbert M. (Hrsg.): *Encyclopedia of the Sciences of Learning*. Boston, MA : Springer US, 2012. – URL https://doi.org/10.1007/978-1-4419-1428-6_1219. – ISBN 978-1-4419-1428-6
- [Kaufman 1990] KAUFMAN, Leonard: Rousseeuw. PJ Finding Groups in Data: an Introduction to Cluster Analysis. In: *Applied Probability and Statistics, New York, Wiley Series in Probability and Mathematical Statistics* (1990)
- [Kobak und Linderman 2019] KOBAK, Dmitry ; LINDERMAN, George C.: UMAP does not preserve global structure any better than t-SNE when using the same initialization. In: *bioRxiv* (2019)
- [Kramer 1991] KRAMER, Mark A.: Nonlinear principal component analysis using autoassociative neural networks. In: *AICHE journal* 37 (1991), Nr. 2, S. 233–243
- [Kriegel u. a. 2008] KRIEGEL, Hans-Peter ; KRÖGER, Peer ; ZIMEK, Arthur: Detecting clusters in moderate-to-high dimensional data: subspace clustering, pattern-based clustering, and correlation clustering. In: *Proceedings of the VLDB Endowment* 1 (2008), Nr. 2, S. 1528–1529
- [Kriegel u. a. 2009] KRIEGEL, Hans-Peter ; KRÖGER, Peer ; ZIMEK, Arthur: Clustering high-dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering. In: *Acm transactions on knowledge discovery from data (tkdd)* 3 (2009), Nr. 1, S. 1–58
- [Van der Maaten und Hinton 2008] MAATEN, Laurens Van der ; HINTON, Geoffrey: Visualizing data using t-SNE. In: *Journal of machine learning research* 9 (2008), Nr. 11
- [McInnes u. a. 2018] MCINNES, Leland ; HEALY, John ; MELVILLE, James: Umap: Uniform manifold approximation and projection for dimension reduction. In: *arXiv preprint arXiv:1802.03426* (2018)

- [Nagpal und Mann 2011] NAGPAL, Pooja B. ; MANN, Priyanka A.: Comparative study of density based clustering algorithms. In: *International Journal of Computer Applications* 27 (2011), Nr. 11, S. 421–435
- [Ng u. a. 2001] NG, Andrew ; JORDAN, Michael ; WEISS, Yair: On spectral clustering: Analysis and an algorithm. In: *Advances in neural information processing systems* 14 (2001), S. 849–856
- [Pedregosa u. a. 2011] PEDREGOSA, F. ; VAROQUAUX, G. ; GRAMFORT, A. ; MICHEL, V. ; THIRION, B. ; GRISEL, O. ; BLONDEL, M. ; PRETTENHOFER, P. ; WEISS, R. ; DUBOURG, V. ; VANDERPLAS, J. ; PASSOS, A. ; COURNAPEAU, D. ; BRUCHER, M. ; PERROT, M. ; DUCHESNAY, E.: Scikit-learn: Machine Learning in Python. In: *Journal of Machine Learning Research* 12 (2011), S. 2825–2830
- [Roux 2018] ROUX, Maurice: A comparative study of divisive and agglomerative hierarchical clustering algorithms. In: *Journal of Classification* 35 (2018), Nr. 2, S. 345–366
- [Sander u. a. 1998] SANDER, Jörg ; ESTER, Martin ; KRIEGEL, Hans-Peter ; XU, Xiaowei: Density-based clustering in spatial databases: The algorithm gdbscan and its applications. In: *Data mining and knowledge discovery* 2 (1998), Nr. 2, S. 169–194
- [Stepp 1987] STEPP, Robert E.: Concepts in Conceptual Clustering. In: *IJCAI* Citeseer (Veranst.), 1987, S. 211–213
- [Von Luxburg 2007] VON LUXBURG, Ulrike: A tutorial on spectral clustering. In: *Statistics and computing* 17 (2007), Nr. 4, S. 395–416
- [Wang u. a. 2015] WANG, Shulin ; GU, Jinchao ; CHEN, Fang: Clustering high-dimensional data via spectral clustering using collaborative representation coefficients. In: *International Conference on Intelligent Computing* Springer (Veranst.), 2015, S. 248–258
- [Xu u. a. 1998] XU, Xiaowei ; ESTER, Martin ; KRIEGEL, H-P ; SANDER, Jörg: A distribution-based clustering algorithm for mining in large spatial databases. In: *Proceedings 14th International Conference on Data Engineering* IEEE (Veranst.), 1998, S. 324–331
- [Yeung und Ruzzo 2001] YEUNG, Ka Y. ; RUZZO, Walter L.: Principal component analysis for clustering gene expression data. In: *Bioinformatics* 17 (2001), Nr. 9, S. 763–774

Hiermit versichere ich, dass ich die vorliegende Arbeit ohne fremde Hilfe selbständig verfasst und nur die angegebenen Hilfsmittel benutzt habe.

Hamburg, 22. July 2021  Anastasiya Vladimirova



Erklärung zur selbstständigen Bearbeitung einer Abschlussarbeit

Gemäß der Allgemeinen Prüfungs- und Studienordnung ist zusammen mit der Abschlussarbeit eine schriftliche Erklärung abzugeben, in der der Studierende bestätigt, dass die Abschlussarbeit „– bei einer Gruppenarbeit die entsprechend gekennzeichneten Teile der Arbeit [(§ 18 Abs. 1 APSO-TI-BM bzw. § 21 Abs. 1 APSO-INGI)] – ohne fremde Hilfe selbständig verfasst und nur die angegebenen Quellen und Hilfsmittel benutzt wurden. Wörtlich oder dem Sinn nach aus anderen Werken entnommene Stellen sind unter Angabe der Quellen kenntlich zu machen.“

Quelle: § 16 Abs. 5 APSO-TI-BM bzw. § 15 Abs. 6 APSO-INGI

Dieses Blatt, mit der folgenden Erklärung, ist nach Fertigstellung der Abschlussarbeit durch den Studierenden auszufüllen und jeweils mit Originalunterschrift als letztes Blatt in das Prüfungsexemplar der Abschlussarbeit einzubinden.

Eine unrichtig abgegebene Erklärung kann -auch nachträglich- zur Ungültigkeit des Studienabschlusses führen.

Erklärung zur selbstständigen Bearbeitung der Arbeit

Hiermit versichere ich,

Name: Vladimirova

Vorname: Anastasiya

dass ich die vorliegende Bachelorarbeit bzw. bei einer Gruppenarbeit die entsprechend gekennzeichneten Teile der Arbeit – mit dem Thema:

Analysis and Comparison of Various Clustering and
Dimensionality Reduction Algorithms on LIDAR-Ceilometer
Aerosol-Backscatter Data

ohne fremde Hilfe selbständig verfasst und nur die angegebenen Quellen und Hilfsmittel benutzt habe. Wörtlich oder dem Sinn nach aus anderen Werken entnommene Stellen sind unter Angabe der Quellen kenntlich gemacht.

- die folgende Aussage ist bei Gruppenarbeiten auszufüllen und entfällt bei Einzelarbeiten -

Die Kennzeichnung der von mir erstellten und verantworteten Teile der -bitte auswählen- ist erfolgt durch:

Hamburg

Ort

22.07.2021

Datum

Unterschrift im Original