Application of Deep Clustering Algorithms

Collin Leiber
Institute for Informatics,
MCML,
LMU Munich
Munich, Germany
leiber@dbs.ifi.lmu.de

Lukas Miklautz
Faculty of Computer Science,
UniVie Doctoral School Computer Science,
University of Vienna
Vienna, Austria
lukas.miklautz@univie.ac.at

Claudia Plant
Faculty of Computer Science,
ds:UniVie,
University of Vienna
Vienna, Austria
claudia.plant@univie.ac.at

Christian Böhm
Faculty of Computer Science,
University of Vienna
Vienna, Austria
christian.boehm@univie.ac.at

ABSTRACT

Deep clustering algorithms have gained popularity for clustering complex, large-scale data sets, but getting started is difficult because of numerous decisions regarding architecture, optimizer, and other hyperparameters. Theoretical foundations must be known to obtain meaningful results. At the same time, ease of use is necessary to get used by a broader audience. Therefore, we require a unified framework that allows for easy execution in diverse settings. While this applies to established clustering methods like k-Means and DBSCAN, deep clustering algorithms lack a standard structure, resulting in significant programming overhead. This complicates empirical evaluations, which are essential in both scientific and practical applications. We present a solution to this problem by providing a theoretical background on deep clustering as well as practical implementation techniques and a unified structure with predefined neural networks. For the latter, we use the Python package ClustPy. The aim is to share best practices and facilitate community participation in deep clustering research.

CCS CONCEPTS

• **Information systems → Clustering**; • **Software and its engineering → Software libraries and repositories**; • **Computing methodologies → Cluster analysis**; *Dimensionality reduction and manifold learning*; *Neural networks.*

KEYWORDS

deep clustering, data mining, unsupervised learning, representation learning

ACM Reference Format:
Collin Leiber, Lukas Miklautz, Claudia Plant, and Christian Böhm. 2023. Application of Deep Clustering Algorithms. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management (CIKM '23), October 21–25, 2023, Birmingham, United Kingdom.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3583780.3615290

This work is licensed under a Creative Commons Attribution International 4.0 License.

CIKM '23, October 21–25, 2023, Birmingham, United Kingdom
© 2023 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-0124-5/23/10.
https://doi.org/10.1145/3583780.3615290

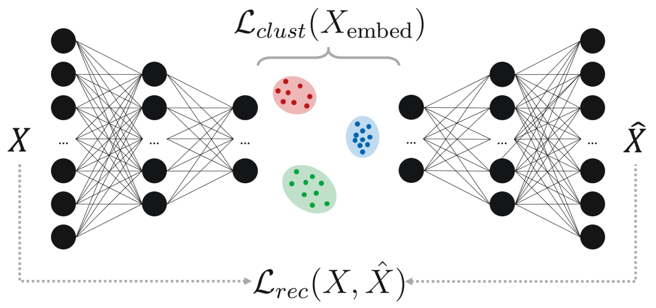
$\mathcal{L}_{clust}(X_{\text{embed}})$
X
$\hat{X}$
$\mathcal{L}_{rec}(X, \hat{X})$

Figure 1: Schematic representation of deep clustering methods. The clustering loss $\mathcal{L}_{clust}$ uses the lower-dimensional representation of an input data set X to optimize the final clustering result. The optional autoencoder reconstruction loss $\mathcal{L}_{rec}$ is used as a data-dependent regularizer for X_{embed}.

1 FORMAT

Half-day (3 hours plus breaks).

2 TUTORIAL OUTLINE

This tutorial serves as a good introduction to the topic of deep clustering. This involves theoretical concepts and their application using our open-source Python package ClustPy[1]. The following topics will be covered:

2.1 Fundamentals of Clustering

Clustering creates groups of objects, where similar objects should be contained in the same group and dissimilar objects in different groups. In contrast to classic machine learning, no predefined labels are used for this. Therefore, the algorithm cannot use any known information to learn the boundaries between the groups, but must define the boundaries on its own. Depending on the objective function, different measures of similarity can be used.

[1]https://github.com/collinleiber/ClustPy

We will briefly discuss the basics of clustering using the well-known k-Means [19] algorithm. k-Means starts by initially selecting k cluster centers and then assigns each object to its closest center. Afterwards, all cluster centers are updated by setting them to the mean of all assigned objects. These steps repeat until the assignments remain constant.

2.2 Introduction to Deep Clustering

Many classical clustering methods have problems processing high-dimensional data. Therefore, an initial dimensionality reduction is often performed (e.g., PCA [9] or ICA [14]). In deep learning, an autoencoder is a common non-linear dimensionality reduction method. The idea is that after reducing the dimensionality of the data set, an arbitrary clustering method such as k-Means [19] or DBSCAN [7] is executed. A problem is that dimensionality reduction and clustering run independently of each other. This means that an intermediate clustering result has no influence on the representation of the data. There are several subspace clustering methods that address this problem (e.g., ORCLUS [1], LDA-k-Means [6], FOSSCLU [10], SubKmeans [20], DipNSub [4]) but they do not have the non-linear representation capabilities that neural networks offer.

Deep clustering on the other hand tries to combine clustering with modern representation learning. Here, the optimization function of an autoencoder is supplemented by a second clustering-based optimization. The combined optimization ensures that the embedding of the autoencoder is improved simultaneously with the clustering result. The tutorial will illustrate how a corresponding architecture looks schematically, an exemplary illustration is given in Fig. 1. Here, the clustering process can be combined with various neural networks like regular feed-forward autoencoders [3], stacked autoencoders [25], convolutional autoencoders [16] or variational autoencoders [15]. However, the combination of deep learning and clustering faces several challenges, which will be discussed in the course of the tutorial. An example is the initialization of deep clustering algorithms [29]. For most procedures an already pretrained autoencoder is used to obtain initial cluster labels and the initial clustering result is crucial as it strongly influences the final clustering result. This leads to several design decisions, e.g., the type of neural network architecture, the amount of pretraining and the type of initial clustering algorithm. Another difficulty is the handling of unbalanced data sets [29]. Small clusters can be heavily underrepresented, especially when working in a mini-batch fashion. During the course of the tutorial we will provide some suggestions on how to deal with these challenges.

2.3 Fundamental Deep Clustering Algorithms

One of the first deep clustering algorithms was DEC [26]. It uses a Kullback-Leibler-based optimization function to identify good clusters. The idea was taken up by IDEC [11]. Unlike DEC, IDEC uses the reconstruction loss also during the clustering process. DCN [27] uses a k-Means-like optimization and shows that hard cluster labels can also be used instead of soft ones. Here, no joint optimization takes place, but representation learning and clustering are performed alternately. This circumstance is addressed by DKM [8], which combines the k-Means approach with a simultaneous optimization of the deep learning structure. All mentioned references will be described in the course of the tutorial since they form a basis for many other procedures (see [29]). Further, so far all algorithms utilize a stacked autoencoder. However, when analyzing images, the performance can often be improved by using convolutional neural networks instead. This is also discussed in the course of the tutorial.

ClustPy

Figure 2: The coding examples are based on ClustPy, which is accessible at: https://github.com/collinleiber/ClustPy.

2.4 Application Process

In order to compare specific deep clustering algorithms we utilize our open-source Python package ClustPy (see Fig. 2). For this purpose, after each rather theoretical section of the tutorial, an appropriate coding example is presented. Here, the complete pipeline for executing the corresponding algorithm is shown. This includes loading an appropriate data set, executing the clustering process, and evaluating and visualizing the result. For example, the code to run DEC on the MNIST data set [17] and evaluating the result using the unsupervised clustering accuracy [28] is as follows:

```python
from clustpy.deep import DEC
from clustpy.data import load_mnist
from clustpy.metrics import cluster_accuracy

data, labels = load_mnist()
dec = DEC(n_clusters=10)
dec.fit(data)
result = cluster_accuracy(labels, dec.labels)
print(result)
```

In addition, we show how essential components of the algorithms, such as the type of autoencoder or optimizer, can be easily exchanged. By using parameters that are as identical as possible, the results of different algorithms can be compared resulting in a fair benchmark for the algorithms presented.

In the tutorial, we will also show some prior work for comparing deep clustering algorithms [2] and discuss them along with our approach. This example will serve as a case study on the importance of a unified setup.

2.5 Discussion of Results

After discussing the previously mentioned procedures, we compare their final results on several datasets. Here, we keep structural differences (e.g., the type of autoencoder) to a minimum which enables a fair comparison. Variations in the results are highlighted and explained by pointing out individual strengths and weaknesses. We will see that deep clustering succeeds in processing large high-dimensional data sets like image data.

2.6 Outlook: Specialized Deep Clustering Algorithms

At the end of the tutorial, we give a brief outlook on modern approaches. These are often more specialized methods that try to handle more complex problems.

An example is DeepCluster [5]. It was one of the first methods to use a convolutional autoencoder to analyze image data. Due to the good results, it is often used and commonly cited. The algorithm VaDE [13] demonstrates that deep clustering can also be applied in combination with a variational autoencoder. Here, the autoencoder optimizes a Gaussian Mixture Model, which is used to cluster the data. Another example is SpectralNet [23]. This algorithm combines deep clustering with the popular SpectralClustering [24] approach, which enables scaling to large data sets. In the before-mentioned algorithms, the correct number of clusters was always assumed to be known a priori. However, this is often not the case in practice. The deep clustering algorithm DipDECK [18] is able to automatically determine the number of clusters in the embedding of an autoencoder, using a statistical test for unimodality [12].

Another category of algorithms are non-redundant clustering methods. These assume that there is not just one but several ways to cluster a data set. For example, one can group objects by their color as well as by their shape. The ENRC [21] algorithm delivers such results in a deep learning setting. Finally, we would like to mention DECCS [22]. This method combines consensus and deep clustering. With consensus clustering, clustering results that come from a wide variety of methods are combined into a common clustering result. Due to the different properties of the original results, the algorithm is very flexible, which often results in high-quality outcomes.

All of these methods are only described briefly in order to show the diversity of deep clustering research. Additionally, we show some examples and highlight the advantages compared to baseline approaches. With this chapter, we would like to encourage the audience to follow up on the methods they are interested in.

3 TARGET AUDIENCE

The target audience of the tutorial are individuals from academia and industry who already have some background in unsupervised learning. The tutorial is divided into two parts. First, we talk about the theoretical background of deep clustering and discuss the main differences between well-known deep clustering algorithms. This part is more interesting for people who are already familiar with clustering but have limited experience with deep clustering techniques. In the second part, we then show how deep clustering algorithms can be run in an easy way using the Python package ClustPy. This part is especially interesting for people who did not work with deep clustering so far, as it makes a good entry point for future analysis.

4 PREREQUISITES

As a prerequisite, a basic understanding of clustering methods such as k-Means is assumed. Furthermore, the audience should have a basic knowledge of statistical methods and distributions. To understand the deep clustering algorithms, an understanding of deep learning concepts (e.g., mini-batch gradient descent) and in particular, the mechanics of an autoencoder are required. For the coding-based part of the tutorial, basic knowledge of standard Python packages such as NumPy[2] and scikit-learn[3] should be already available. Basic knowledge of deep learning frameworks like PyTorch[4] is helpful, but not strictly required.

5 BENEFITS AND IMPACT FOR CIKM COMMUNITY

Due to the representational capability of Deep Learning, a large number of deep clustering methods have been published in recent years. Our tutorial provides an introduction to fundamental deep clustering methods and provides an outlook to more specialized algorithms. These have shown very good performance with respect to large, high-dimensional data (especially image data), but are often difficult to apply in practice. After the tutorial participants will know about several deep clustering methods and will know how they can be applied without a substantial effort.

Moreover, deep clustering methods are hard to compare with each other due to the many structural design decisions and other parameterization options. A fair evaluation is therefore difficult, which in turn makes it complicated to correctly classify the strengths and weaknesses of different methods. In this tutorial, we draw attention to the issues and suggest best practices. We hope this will lead to better and fairer comparability in the future.

6 LIST OF FORUMS

The tutorial hasn't been presented at any other venues yet.

7 TUTORS

The following tutors are involved in the preparation of the tutorial and will present the tutorial in person.

Collin Leiber

Collin Leiber is a PhD student at Ludwig-Maximilians-Universität München, Germany, and a member of the Munich Center for Machine Learning (MCML). He received his MSc degree in Computer Science with a specialization in data analytics from Ludwig-Maximilians-Universität München in 2019. His current research interests are mainly focused on data mining. In particular, he is working on the question of how to determine an appropriate number of clusters in complex environments, such as deep clustering and alternative clustering. For this purpose, he researches statistics-based methods, density-based methods, and those based on information theory. Furthermore, he is the main developer of the open-source ClustPy package.

- ORCID: https://orcid.org/0000-0001-5368-5697
- DBLP: https://dblp.uni-trier.de/pid/299/4795.html
- Google scholar https://scholar.google.com/Leiber

Lukas Miklautz

Lukas Miklautz is a PhD student at the Data Mining and Machine Learning research group at the Faculty of Computer Science University of Vienna, Austria, and a member of the UniVie Doctoral School

[2]https://numpy.org/
[3]https://scikit-learn.org/
[4]https://pytorch.org/

Computer Science. His research focuses on combining representation learning and clustering (Deep Clustering). Deep clustering methods use unsupervised or self-supervised learning algorithms with clustering objectives to improve clustering performance. Deep clustering is applicable to many clustering paradigms like non-redundant, consensus, or subspace clustering. For this purpose, he researches both clustering and deep learning methods. Furthermore, he is a contributor of the open-source ClustPy package.

- ORCID: https://orcid.org/0000-0002-2585-5895
- DBLP: https://dblp.uni-trier.de/pid/262/6652.html
- Google scholar https://scholar.google.com/Miklautz

Claudia Plant

Claudia Plant is a full professor and leader of the research group Data Mining and Machine Learning at the Faculty of Computer Science University of Vienna, Austria. Her research focuses on new methods for exploratory data mining, mostly on clustering and representation learning. Many approaches relate unsupervised learning to data compression, i.e. the better the found patterns compress the data the more information we have learned. Other methods rely on finding statistically independent patterns or multiple non-redundant solutions, on ensemble learning or on nature-inspired concepts such as synchronization. Together with her group, she contributed a lot of clustering methods based on deep learning, spectral methods and matrix factorization. She also develops application-oriented approaches to data mining in the context of interdisciplinary projects with experts from neuroscience, bio-medicine, particle physics, social sciences and archaeology.

- ORCID: https://orcid.org/0000-0001-5274-8123
- DBLP: https://dblp.uni-trier.de/pid/65/3446.html
- Google scholar https://scholar.google.com/Plant

Christian Böhm

Christian Böhm is a professor of Computer Science at the University of Vienna, Austria. He received his PhD degree in 1998 from Ludwig-Maximilians-Universität München, Germany. His research interests cover clustering and representation learning with a particular focus on high-performance aspects such as index structures, parallelization, vectorization, and efficient memory accesses through cache. He has more than 150 publications including KDD, ICDM, and SIGMOD.

- ORCID: https://orcid.org/0000-0002-2237-9969
- DBLP: https://dblp.uni-trier.de/pid/b/ChBohm.html
- Google scholar https://scholar.google.com/Boehm

REFERENCES

[1] Charu C Aggarwal and Philip S Yu. 2000. Finding generalized projected clusters in high dimensional spaces. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*. 70–81.

[2] Elie Aljalbout, Vladimir Golkov, Yawar Siddiqui, Maximilian Strobel, and Daniel Cremers. 2018. Clustering with deep learning: Taxonomy and new methods. *arXiv preprint arXiv:1801.07648* (2018).

[3] Dana H Ballard. 1987. Modular learning in neural networks.. In *AAAI*, Vol. 647. 279–284.

[4] Lena GM Bauer, Collin Leiber, Christian Böhm, and Claudia Plant. 2023. Extension of the Dip-test Repertoire-Efficient and Differentiable p-value Calculation for Clustering. In *Proceedings of the 2023 SIAM International Conference on Data Mining (SDM)*. SIAM, 109–117.

[5] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. 2018. Deep Clustering for Unsupervised Learning of Visual Features. In *ECCV (14) (Lecture Notes in Computer Science, Vol. 11218)*. Springer, 139–156.

[6] Chris Ding and Tao Li. 2007. Adaptive dimension reduction using discriminant analysis and k-means clustering. In *Proceedings of the 24th international conference on Machine learning*. 521–528.

[7] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise.. In *kdd*, Vol. 96. 226–231.

[8] Maziar Moradi Fard, Thibaut Thonet, and Eric Gaussier. 2020. Deep k-means: Jointly clustering with k-means and learning representations. *Pattern Recognition Letters* 138 (2020), 185–192.

[9] Karl Pearson F.R.S. 1901. LIII. On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* 2, 11 (1901), 559–572.

[10] Sebastian Goebl, Xiao He, Claudia Plant, and Christian Böhm. 2014. Finding the optimal subspace for clustering. In *2014 IEEE International Conference on Data Mining*. IEEE, 130–139.

[11] Xifeng Guo, Long Gao, Xinwang Liu, and Jianping Yin. 2017. Improved Deep Embedded Clustering with Local Structure Preservation. In *IJCAI*. ijcai.org, 1753–1759.

[12] John A Hartigan and Pamela M Hartigan. 1985. The dip test of unimodality. *The annals of Statistics* (1985), 70–84.

[13] Zhuxi Jiang, Yin Zheng, Huachun Tan, Bangsheng Tang, and Hanning Zhou. 2017. Variational Deep Embedding: An Unsupervised and Generative Approach to Clustering. In *IJCAI*. ijcai.org, 1965–1972.

[14] Christian Jutten and Jeanny Hérault. 1991. Blind separation of sources, part I: An adaptive algorithm based on neuromimetic architecture. *Signal Process.* 24, 1 (1991), 1–10.

[15] Diederik P. Kingma and Max Welling. 2014. Auto-Encoding Variational Bayes. In *ICLR*.

[16] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. 1989. Backpropagation applied to handwritten zip code recognition. *Neural computation* 1, 4 (1989), 541–551.

[17] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. *Proc. IEEE* 86, 11 (1998), 2278–2324.

[18] Collin Leiber, Lena G. M. Bauer, Benjamin Schelling, Christian Böhm, and Claudia Plant. 2021. Dip-based Deep Embedded Clustering with k-Estimation. In *ACM SIGKDD*. ACM, 903–913.

[19] Stuart Lloyd. 1982. Least squares quantization in PCM. *IEEE transactions on information theory* 28, 2 (1982), 129–137.

[20] Dominik Mautz, Wei Ye, Claudia Plant, and Christian Böhm. 2017. Towards an optimal subspace for k-means. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 365–373.

[21] Lukas Miklautz, Dominik Mautz, Muzaffer Can Altinigneli, Christian Böhm, and Claudia Plant. 2020. Deep embedded non-redundant clustering. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 34. 5174–5181.

[22] Lukas Miklautz, Martin Teuffenbach, Pascal Weber, Rona Perjuci, Walid Durani, Christian Böhm, and Claudia Plant. 2022. Deep Clustering With Consensus Representations. In *2022 IEEE International Conference on Data Mining (ICDM)*. 1119–1124. https://doi.org/10.1109/ICDM54844.2022.00141

[23] Uri Shaham, Kelly Stanton, Henry Li, Boaz Nadler, Ronen Basri, and Yuval Kluger. 2018. Spectralnet: Spectral clustering using deep neural networks. *arXiv preprint arXiv:1801.01587* (2018).

[24] Jianbo Shi and Jitendra Malik. 2000. Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence* 22, 8 (2000), 888–905.

[25] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, Pierre-Antoine Manzagol, and Léon Bottou. 2010. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of machine learning research* 11, 12 (2010).

[26] Junyuan Xie, Ross B. Girshick, and Ali Farhadi. 2016. Unsupervised Deep Embedding for Clustering Analysis. In *ICML (JMLR Workshop and Conference Proceedings, Vol. 48)*. JMLR.org, 478–487.

[27] Bo Yang, Xiao Fu, Nicholas D. Sidiropoulos, and Mingyi Hong. 2017. Towards K-means-friendly Spaces: Simultaneous Deep Learning and Clustering. In *ICML (Proceedings of Machine Learning Research, Vol. 70)*. PMLR, 3861–3870.

[28] Yi Yang, Dong Xu, Feiping Nie, Shuicheng Yan, and Yueting Zhuang. 2010. Image clustering using local discriminant models and global integration. *IEEE Transactions on Image Processing* 19, 10 (2010), 2761–2773.

[29] Sheng Zhou, Hongjia Xu, Zhuonan Zheng, Jiawei Chen, Jiajun Bu, Jia Wu, Xin Wang, Wenwu Zhu, Martin Ester, et al. 2022. A comprehensive survey on deep clustering: Taxonomy, challenges, and future directions. *arXiv preprint arXiv:2206.07579* (2022).