CATEGORIZATION OF METEOROLOGICAL DATA BY CONTRASTIVE CLUSTERING

Michael Dammann[1]*, Ina Mattis[2], Michael Neitzke[1] & Ralf Möller[3]
[1]University of Applied Sciences Hamburg, [2]Deutscher Wetterdienst (DWD),
[3]University of Hamburg

ABSTRACT

Visualized ceilometer backscattering data, displaying meteorological phenomena like clouds, precipitation, and aerosols, is mostly analyzed manually by meteorology experts. In this work, we present an approach for the categorization of backscattering data using a contrastive clustering approach, incorporating image and spatiotemporal information into the model. We show that our approach leads to meteorologically meaningful clusters, opening the door to the automatic categorization of ceilometer data, and how our work could potentially create insights in the field of climate science.

1 INTRODUCTION

Ceilometers are stationary LIDAR systems that continuously measure meteorological backscattering data. At the moment, the analysis of this data is largely done manually by meteorology experts (Thomas et al., 2018). An example can be seen in Fig. 1, top left, where the measurements of a whole day are visualized by applying a defined colormap. The y-axis corresponds to the altitude (0 to 10 km) and the x-axis to the time of the day (0 to 24 hours). A meteorology expert can take a look at this image and make out distinct meteorological phenomena like clouds and precipitation (dark colors, high backscattering) or aerosols (lighter colors, low backscattering). In this work, we try to automate this process by applying three processing steps: (1) segmenting and *slicing* the image based on pixel similarity (classical image processing), (2) finding vector representations of these segments, and (3) clustering these representations (deep learning). We apply an unsupervised clustering method as the data is unlabeled. After clustering, the resulting clusters can be annotated by meteorology experts. This basic idea of ceilometer data analysis is based upon a previous work of ours (Dammann et al., 2022).

2 SLICING AND REPRESENTATION CONSTRUCTION

As the occurring phenomena often have very distinct colors and shapes that clearly differentiate them from the background, we apply the Felzenszwalb-Huttenlocher (FH) segmentation algorithm (Felzenszwalb & Huttenlocher, 2004) to extract these shapes. Ideally, each extracted shape corresponds to a certain phenomenon (like a cloud). In practice, some phenomena are compartmentalized, and sometimes multiple phenomena are aggregated in a single segment. Overall, however, we consider the segmentation to be suitable and leave possible optimization for future work. For each detected segment, we then determine the minimum and maximum x-values (daytimes) to create a *slice* consisting of the image data between the minimum and maximum x-values. This is done because the altitude information contained in these slices is considered vital for phenomena categorization (for example, cirrus clouds are clouds of very high altitude). This information would get lost by using the cut out segments only. This slicing step is visualized in Fig. 1 (left).

For each slice, we then construct a set of feature vectors, namely the *location* encoding, *daytime* encoding, and *month* encoding, displayed in Fig. 1 (right). We construct the multi-hot encoded daytime encoding (24-dim. vector that represents the 24 hours of the day) to incorporate the daytime

*michael.dammann@haw-hamburg.de

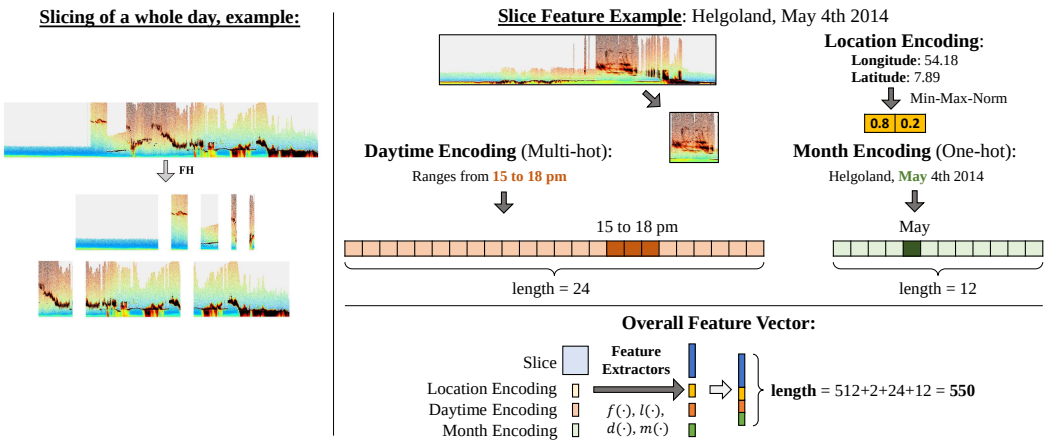


Figure 1: Slice generation and slice feature extraction.

information (*when does the slice occur?*) lost in the slicing process. We also use a month encoding (12-dim.) as seasonal patterns are important for meteorological phenomena, as well as a location encoding (2-dim.: longitude, latitude) as weather phenomena are dependent on the location. More details on segmentation and slicing are available in A.2 and A.3. In the next contrastive clustering step, see the next section, feature extractors f, l, d, and m are learned that map the image data to a fixed-size vector and further map the encodings into a new learned feature space each and the representations are then clustered in an end-to-end architecture.

3 Contrastive Clustering of Ceilometer Data

We closely follow the framework and notation of Li et al. Li et al. (2021), which our work is based on and an extension to. Given a batch of samples, we first apply an individual, stochastic data augmentation T to every sample. An original sample and its augmentation are considered a positive pair, two samples/augmentations from different original samples are defined as a negative pair.

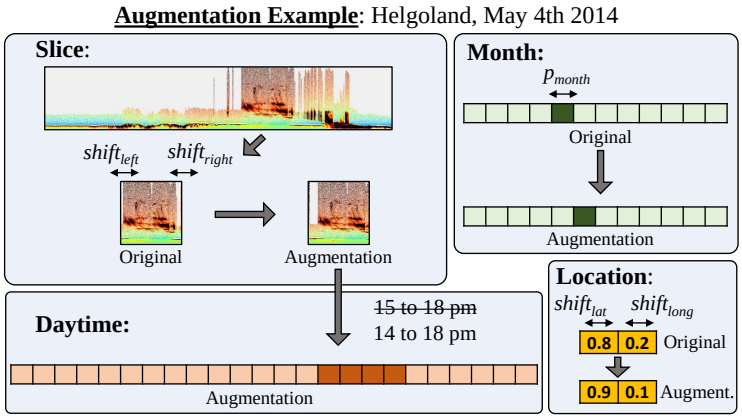


Figure 2: Augmentation of ceilometer data.

We define a set of stochastic augmentations that is applicable to our dataset, visualized in Fig. 2. To augment the slices, we decided that it is acceptable (i.e., the same phenomenon is still captured) for our kind of data to slightly shift, expand, or compress the existing window sizes. $shift_{s\_range}$ is defined, which describes the maximum percentage each side of the window (left and right) might be shifted. Then, $shift_{left}$ and $shift_{right}$ are each randomly sampled from $[-shift_{s\_range}, shift_{s\_range}]$ to augment the window size. We choose $shift_{s\_range} = 0.1$, so for example a formerly 60-minute window/slice could at maximum be shifted by ± 6 minutes on

the left and ±6 minutes on the right. The daytime encoding is adjusted correspondingly. For the month encoding, we define p_{month} as the probability of shifting the month one to the right or left. In the case of shifting, the direction is then randomly chosen. We set p_{month} to 0.25. We also apply (slight) shiftings to the location encodings, defining $shift_{c\_range}$ as the maximum, absolute shift of the latitude and longitude. We sample $shift_{lat}$ and $shift_{long}$ from $[-shift_{c\_range}, shift_{c\_range}]$ and add them to the original latitude and longitude, respectively. We set $shift_{c\_range}$ to 0.05.

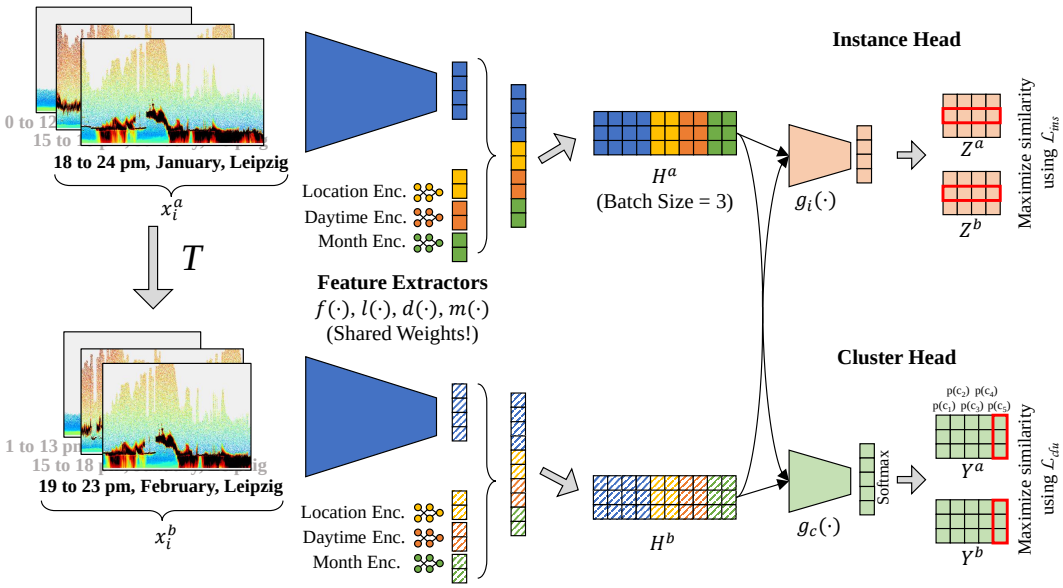


Figure 3: Contrastive clustering for ceilometer data.

The contrastive clustering approach is visualized in Fig. 3. We learn features for both the original samples and the augmentations using $f(\cdot)$, $l(\cdot)$, $d(\cdot)$ and $m(\cdot)$ as already mentioned in Sec. 2. Note that both paths use the same, shared weights. The result of this step are two feature matrices H^a and H^b for the original batch and the augmented one, respectively. The architecture is then split up into an instance head and a cluster head.

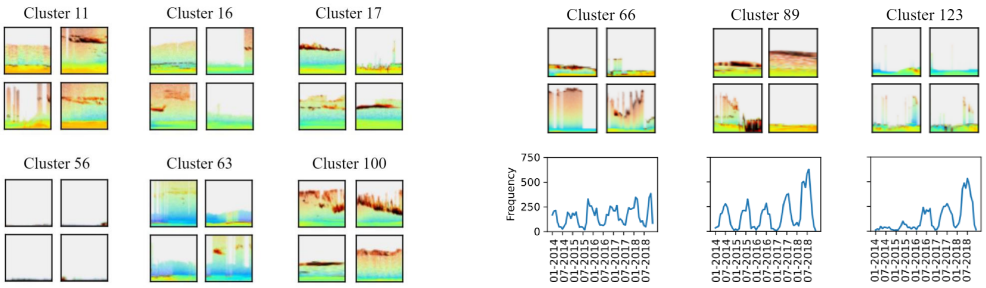
In the instance head, the features are projected to their final representations (Z^a and Z^b) by applying $g_i(\cdot)$. Here, the cosine similarity is maximized between positive pairs and minimized between negative pairs. The cluster head maps the features for each sample to a vector with a dimensionality of the desired number of clusters M, which are softmax-activated to make them interpretable as probabilities for the cluster assignments (Y^a and Y^b). When each sample corresponds to a row in Y^a and Y^b, then each column corresponds to the probabilities of all samples in the mini-batch for the respective cluster. Thus, each column can be interpreted as the mini-batch *cluster representation*. Here, the cosine similarities between cluster representations of the same clusters are maximized ("positive pairs") and between cluster representations from different clusters ("negative pairs") are minimized.

As a rough estimation of an appropriate cluster number M, we first take the set of possibly occurring "base phenomena": B = {rain, precipitation, aerosol}. We also consider the set of seasons S = {summer, autumn, winter, spring} and daytimes D = {night, morning, noon, evening}. Our estimate and choice of M is then given by $M = |\mathcal{P}(B)| \cdot |S| \cdot |D| = 2^3 \cdot 4 \cdot 4 = 128$, since a slice can contain no phenomenon, single phenomena, or mixtures of multiple phenomena and each slice could theoretically occur in any season at any daytime. A large M also leads to highly specialized clusters, potentially simplifying the manual cluster annotation process.

We give a more mathematical formulation, more details regarding the hyperparameters of our approach and implementation details in A.4 and A.5.

4 RESULTS

In this section, we present the results of our approach. We show what categories of clusters are detected and how our framework could help generate insights for climate science.



(a) (b)

Figure 4: (a) Top 4 slices with highest cluster affiliations for 6 example clusters. (b) Development over years for 3 example clusters.

Fig. 4 (a) shows eight exemplary clusters and the four slices that have the highest cluster affiliation for each. Cluster 11 groups slices that contain a planetary boundary layer (PBL, region at the bottom) with relatively high backscattering values (indicated by the orange color). The slices in this cluster are often accompanied by clouds in mid to high altitudes (red horizontal structures). Clusters 16 and 17 also detect the PBL, although the backscattering in these clusters is lower, indicated by the greenish color. Examples for cluster 17 also contain more clouds than the slices in cluster 16. Cluster 56 groups typical examples of fog, which generates a high backscattering signal barely above the ceilometer altitude. Cluster 63 displays examples of clear, sunny hours, sometimes accompanied by small clouds. Cluster 100 detects large, mid to high-altitude clouds, which don't lead to precipitation.

In Fig. 4 (b) we show the occurrence of cluster affiliations of the whole time range of our dataset. Cluster 66 (low, mid, and high altitude clouds) is an example of a stable cluster, showing a steady seasonality pattern of always peaking at roughly the same number of samples in autumn. Cluster 89 (low to mid-altitude clouds, PBL) shows phenomena that slightly increase in occurrence over time. Cluster 123 (as discussed above) shows slices that capture clear whole days. It can be seen that in summer 2014 these whole days are still rare and that their frequency increases each summer. The maximum in summer 2018 corresponds to a strong anomaly of dry and warm conditions DWD (2022). This is an example of how our system could help to detect climatic patterns in ceilometer data to advance climate science. Further results are described in A.6.

5 CONCLUSION AND OUTLOOK

Our approach is able to capture meaningful clusters of meteorological phenomena. A logical next step would be to evaluate our method in a production environment. For this reason, we strongly consider integrating our evidential prototype into the infrastructure of the German Weather Service DWD in the foreseeable future.

Since the task of detecting phenomena in ceilometer data can basically be interpreted as an object detection or image segmentation task, it is planned to acquire at least a small amount of labeled data for backscattering data. This would open up the whole field of supervised learning. Then, in our case of a low number of labeled samples and sometimes very rare phenomena, methods from the field of few-shot (Wang et al., 2020) and one-shot learning (O' Mahony et al., 2019) could be applied. Another long-term goal would also be the spatiotemporal tracking of individual phenomena, distributed over several ceilometer measurements.

REFERENCES

Abien Fred Agarap. Deep learning using rectified linear units (relu). *arXiv preprint arXiv:1803.08375*, 2018.

Mihai Boldeanu, Cristian Manolache, Horia Cucu, Corneliu Burileanu, and Camelia Talianu. Aerosol layer identification and segmentation from lidar and ceilometer profiles using unsupervised deep learning. In *European Lidar Conference 2021, 16–18 Nov 2021*, November 2021.

Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. Signature verification using a "siamese" time delay neural network. In J. Cowan, G. Tesauro, and J. Alspector (eds.), *Advances in Neural Information Processing Systems*, volume 6. Morgan-Kaufmann, 1994. URL https://proceedings.neurips.cc/paper/1993/file/288cc0ff022877bd3df94bc9360b9c5d-Paper.pdf.

Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In Hal Daumé III and Aarti Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 1597–1607. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/chen20j.html.

Michael Dammann, Ina Mattis, Michael Neitzke, and Ralf Möller. Towards the automatic analysis of ceilometer backscattering profiles using unsupervised learning. In *NeurIPS 2022 Workshop on Tackling Climate Change with Machine Learning*, 2022. URL https://www.climatechange.ai/papers/neurips2022/99.

DWD. Zeitreihen und trends (time series and trends), February 2022. URL https://dwd.de/DE/leistungen/zeitreihen/zeitreihen.html.

Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Efficient graph-based image segmentation. *Int. J. Comput. Vision*, 59(2):167–181, sep 2004. ISSN 0920-5691. doi: 10.1023/B:VISI.0000022288.19776.77. URL https://doi.org/10.1023/B:VISI.0000022288.19776.77.

H. Flentje, H. Claude, T. Elste, S. Gilge, U. Köhler, C. Plass-Dülmer, W. Steinbrecht, W. Thomas, A. Werner, and W. Fricke. The eyjafjallajökull eruption in april 2010 – detection of volcanic plume using in-situ measurements, ozone sondes and lidar-ceilometer profiles. *Atmospheric Chemistry and Physics*, 10(20):10085–10092, 2010. doi: 10.5194/acp-10-10085-2010. URL https://acp.copernicus.org/articles/10/10085/2010/.

Martial Haeffelin, Thierry Bergot, Thierry Elias, Robert Tardif, Dominique Carrer, Patrick Chazette, Michèle Colomb, P. Dobrinski, Eric Dupont, Jean-Charles Dupont, L. Gomes, L. Musson-Genon, Chris Pietras, Artemio Plana-Fattori, Alain Protat, J. Rangognio, Jean-Christophe Raut, Samuel Rémy, Daniel RICHARD, and X. Zhang. Parisfog: Shedding new light on fog physical processes. *Bulletin of the American Meteorological Society*, 91:767–783, 06 2010. doi: 10.1175/2009BAMS2671.1.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016. doi: 10.1109/CVPR.2016.90.

Weihua Hu, Takeru Miyato, Seiya Tokui, Eiichi Matsumoto, and Masashi Sugiyama. Learning discrete representations via information maximizing self-augmented training. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, pp. 1558–1567. JMLR.org, 2017.

J. Huertas-Tato, F. J. Rodríguez-Benítez, C. Arbizu-Barrena, R. Aler-Mur, I. Galvan-Leon, and D. Pozo-Vázquez. Automatic cloud-type classification based on the combined use of a sky camera and a ceilometer. *Journal of Geophysical Research: Atmospheres*, 122(20):11,045–11,061, 2017. doi: https://doi.org/10.1002/2017JD027131. URL https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1002/2017JD027131.

Karoliina Hämäläinen, Anne Hirsikko, Ari Leskinen, Mika Komppula, Ewan J. O'Connor, and Sami Niemelä. Evaluating atmospheric icing forecasts with ground-based ceilometer profiles. *Meteorological Applications*, 27(6):e1964, 2020. doi: https://doi.org/10.1002/met.1964. URL https://rmets.onlinelibrary.wiley.com/doi/abs/10.1002/met.1964.

Yong-Hyuk Kim, Seung-Hyun Moon, and Yourim Yoon. Detection of precipitation and fog using machine learning on backscatter data from lidar ceilometer. *Applied Sciences*, 10(18):6452, Sep 2020. ISSN 2076-3417. doi: 10.3390/app10186452. URL http://dx.doi.org/10.3390/app10186452.

S. Kotthaus, E. O'Connor, C. Münkel, C. Charlton-Perez, M. Haeffelin, A. M. Gabey, and C. S. B. Grimmond. Recommendations for processing atmospheric attenuated backscatter profiles from vaisala cl31 ceilometers. *Atmospheric Measurement Techniques*, 9(8):3769–3791, 2016. doi: 10.5194/amt-9-3769-2016. URL https://amt.copernicus.org/articles/9/3769/2016/.

Simone Kotthaus and C. Sue B. Grimmond. Atmospheric boundary-layer characteristics from ceilometer measurements. part 1: A new method to track mixed layer height and classify clouds. *Quarterly Journal of the Royal Meteorological Society*, 144(714):1525–1538, 2018. doi: https://doi.org/10.1002/qj.3299. URL https://rmets.onlinelibrary.wiley.com/doi/abs/10.1002/qj.3299.

Yunfan Li, Peng Hu, Zitao Liu, Dezhong Peng, Joey Tianyi Zhou, and Xi Peng. Contrastive clustering. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(10):8547–8555, May 2021. URL https://ojs.aaai.org/index.php/AAAI/article/view/17037.

Stuart P. Lloyd. Least squares quantization in pcm. *IEEE Trans. Inf. Theory*, 28(2):129–136, 1982. URL http://dblp.uni-trier.de/db/journals/tit/tit28.html#Lloyd82.

Giovanni Martucci, Conor Milroy, and Colin D. O'Dowd. Detection of cloud-base height using jenoptik chm15k and vaisala cl31 ceilometers. *Journal of Atmospheric and Oceanic Technology*, 27(2):305 – 318, 2010. doi: 10.1175/2009JTECHA1326.1. URL https://journals.ametsoc.org/view/journals/atot/27/2/2009jtecha1326_1.xml.

Tero Mielonen, Veijo Aaltonen, Heikki Lihavainen, Antti-Pekka Hyvärinen, Antti Arola, Mika Komppula, and Rigel Kivi. Biomass burning aerosols observed in northern finland during the 2010 wildfires in russia. *Atmosphere*, 4(1):17–34, 2013. ISSN 2073-4433. doi: 10.3390/atmos4010017. URL https://www.mdpi.com/2073-4433/4/1/17.

N. O' Mahony, Sean Campbell, Anderson Carvalho, L. Krpalkova, Gustavo Velasco Hernandez, Suman Harapanahalli, D. Riordan, and J. Walsh. One-shot learning for custom identification tasks; a review. *Procedia Manufacturing*, 38:186–193, 2019. ISSN 2351-9789. doi: https://doi.org/10.1016/j.promfg.2020.01.025. URL https://www.sciencedirect.com/science/article/pii/S2351978920300263. 29th International Conference on Flexible Automation and Intelligent Manufacturing (FAIM 2019), June 24-28, 2019, Limerick, Ireland, Beyond Industry 4.0: Industrial Advances, Engineering Education and Intelligent Manufacturing.

Jennifer Sleeman, Milton Halem, Zhifeng Yang, Vanessa Caicedo, Belay Demoz, and Ruben Delgado. A deep machine learning approach for lidar based boundary layer height detection. In *IGARSS 2020 - 2020 IEEE International Geoscience and Remote Sensing Symposium*, pp. 3676–3679, 2020. doi: 10.1109/IGARSS39084.2020.9324191.

Werner Thomas, Harald Flentje, Ina Mattis, and Gerhard Müller. How ceilometers detect saharan dust events, May 2018. URL https://www.dwd.de/DWD/forschung/projekte/ceilomap/files/Saharan_dust_example_en.pdf.

Yaqing Wang, Quanming Yao, James T. Kwok, and Lionel M. Ni. Generalizing from a few examples: A survey on few-shot learning. 53(3), jun 2020. ISSN 0360-0300. doi: 10.1145/3386252. URL https://doi.org/10.1145/3386252.

M. Wiegner, F. Madonna, I. Binietoglou, R. Forkel, J. Gasteiger, A. Geiß, G. Pappalardo, K. Schäfer, and W. Thomas. What is the benefit of ceilometers for aerosol remote sensing? an answer from earlinet. *Atmospheric Measurement Techniques*, 7(7):1979–1997, 2014. doi: 10.5194/amt-7-1979-2014. URL https://amt.copernicus.org/articles/7/1979/2014/.

Jin Ye, Lei Liu, Qi Wang, Shuai Hu, and Shulei Li. A novel machine learning algorithm for planetary boundary layer height estimation using aeri measurement data. *IEEE Geoscience and Remote Sensing Letters*, 19:1–5, 2022. doi: 10.1109/LGRS.2021.3073048.

Yunpeng Zheng, Xuelong Li, and Xiaoqiang Lu. Unsupervised learning of human action categories in still images with deep representations. 15(4), dec 2019. ISSN 1551-6857. doi: 10.1145/3362161. URL https://doi.org/10.1145/3362161.

A Appendix

A.1 Related Work

In this section, we give a brief overview of important works in the two areas that are linked to our work: ceilometer data analysis (section A.1.1) and contrastive clustering (section A.1.2).

A.1.1 Ceilometer Data Analysis

There exists a range of analytical (non-ML) approaches for the analysis of ceilometer data. Most of them focus on only one "task" (Kotthaus et al., 2016; Wiegner et al., 2014), one example being cloud layer identification (Martucci et al., 2010). Apart from clouds, works on (among others) precipitation (Hämäläinen et al., 2020), fog (Haeffelin et al., 2010), and boundary layer dynamics (Kotthaus & Grimmond, 2018) have been published. Furthermore, studies of biomass burning (Mielonen et al., 2013), volcanic ash events (Flentje et al., 2010), and Saharan dust (Thomas et al., 2018) were carried out.

In recent years, ML methods have found their way into the analysis of ceilometer data. A "single-task" trend can be observed here as well. Kim et al. (2020) apply supervised learning to detect precipitation and fog in backscattering profiles. Huertas-Tato et al. (2017) combine ceilometer and sky camera data to train a random forest classifier on cloud-type detection. Sleeman et al. (2020) and Ye et al. (2022) use ML to determine planetary boundary layer heights. The work of Boldeanu et al. (2021) is more in the spirit of our approach. Their method is broader in the sense that it aims to categorize a range of aerosol phenomena and also applies unsupervised ML. However, we further incorporate spatiotemporal information into our approach, apply a deep learning clustering method (CC) and aim at categorizing every possibly occurring phenomenon using unsupervised learning.

A.1.2 Contrastive Clustering

Dealing with unlabeled data, clustering methods are common tools to extract structure in an unsupervised fashion. However, in an era of ever-growing data, "classical" approaches like k-means (Lloyd, 1982) tend to perform sub-optimally on large, high-dimensional datasets, mostly because they lack the capability of learning meaningful representations. In recent years, multiple approaches have arisen that apply deep learning (as a powerful representation learning tool) to the clustering problem.

One straightforward approach is DeepClustering (Caron et al., 2018), which clusters the features generated by a neural net in an iterative, two-step fashion and then uses the cluster assignment for supervised learning. Another example of a likewise two-step approach would be JULE (Zheng et al., 2019).

In recent years, contrastive learning has emerged as a powerful tool for representation learning, tracing back to the original idea of "siamese" neural nets (Bromley et al., 1994). The basic idea behind contrastive learning is the construction of *positive pairs* that should have a representation close to each other and *negative pairs* whose representations should be apart. One common way (also applied in this work) to construct a positive pair is to apply stochastic augmentations to a given data sample (Chen et al., 2020). The intuition behind this is as follows: The high-level vector representation of - for example - a dog should ideally always be the same, regardless of the dog facing to the right or left, being mirrored or slightly stretched. An example of a negative pair would be an image of a dog and an image of a cat. Since the number of possible negative pairs vastly outnumbers the amount of possible positive pairs, the ideal and efficient construction of negative pairs is a big topic of debate. One option, which is used in this work, is to simply generate a positive pair and to use all other samples of a batch for the construction of negative pairs (Chen et al., 2020).

Consequently, due to their performance in representation learning, it is natural to combine contrastive learning and clustering. In this work, we closely follow Contrastive Clustering, proposed by Li et al. (2021), which applies contrastive learning both on the feature and the clustering level.

A.2 Data and Slicing

In the first step, we apply the Felzenszwalb-Huttenlocher (FH) Felzenszwalb & Huttenlocher (2004) segmentation algorithm to our data samples to generate individually sized slices of possibly occurring phenomena (see Fig. 1, left). Under the assumption that single phenomena (like clouds) can be located based on their color and contouring, using an image segmentation algorithm like FH that clusters parts of the image based on pixel intensity and distance is a natural choice.

We use the backscattering data from 87 stations in Germany, ranging from the years 2014 to 2018. A quick look is created for every single day applying the colormap used by the DWD. We had to decide whether to use the actual backscattering values or the resulting RGB values by applying said colormap. Having compared both options on FH, we chose the RGB values since it led to better segmentation.

The quick looks get cut off at an altitude of 10 km and the overall size of the resulting images is scaled down by a factor of 4 to reduce computational complexity. We use the `scikit-image` implementation of FH using the parameters `scale`=1500, `sigma`=2 and `min_size`=200. Every quick look generated from our dataset which is segmented results in a total of about 1.2 million slices. An example is given in Fig. 1 (left). It is apparent that reasonable slices are generated, each showing different kinds of phenomena like in this case *clear sky*, *clouds* or *precipitation*. While in general, FH generates precise pixel outlines of the detected segments (for example the exact outline of a cloud), we chose to consider not only these sole segments. Instead, the whole altitude of a segment is taken into account, leading to the visible "sliding window-like segmentation" resulting in the differently sized slices. Doing this, we hypothesize to capture meaningful context for each segment. For example, dealing with a cloud above some aerosol layer is different from a sole cloud and extracting a cloud without its neighborhood leads to loss of context and information. It can also be seen that some overlap between the slices exists. This is accepted since some phenomena can indeed overlap (for example two stacked clouds) and many phenomena are fuzzy and transition smoothly between each other.

We analyzed the resulting slice dataset. The histogram showing the distribution of slice durations is shown in Fig. 5. It can be seen that most slices (i.e., by FH detected phenomena) are comparatively short and in the 0- to 3-hour range. There exists a spike in the 23- to 24-hour range due to FH in many cases detecting the outline of a whole day as a single segment. These samples are left in the slice dataset since often it might be sensible to use the whole day as a single phenomenon. For example, when the whole day is constantly cloudy it's perfectly fine to detect all 24 hours as a single coherent segment. Table 1 displays descriptive statistical properties of the slice durations. The median being lower than the mean again reflects there being outliers to the right ("whole day segments"). 25 % of the slices are at most ca. 55 minutes long (0.92 hours), again showing that most slices are rather short.

Table 1: Descriptive statistics on slice durations

#Slices	1,213,238
Mean Duration	6.48
Std. deviation	7.64
Median	3.35
25 % percentile	0.92
75 % percentile	7.34

A.3 Feature Extraction

We use the dataset obtained by the slicing in section A.2 for CC. We incorporate multiple features into our approach. Since meteorological phenomena are, in general, dependent on when and where they take place, we not only use the image data from the slices but further use, as mentioned before, information about the location of the ceilometer, the time of the day the slice took place in and the month as input for the model.

The detailed preprocessing of the data is visualized in Fig. 1 (right). The raw image data from the extracted slice represents the first set of initial features. For each slice, we also encode the location

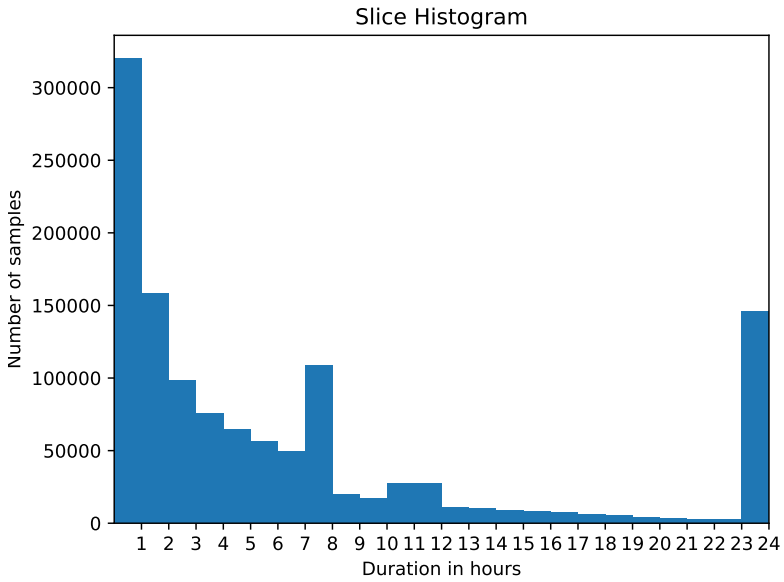


Figure 5: Histogram of slice durations.

of the associated ceilometer. We use the min-max-normalization of longitude and latitude of the location, resulting in a 2-dimensional vector. A multi-hot encoding is used for the daytime; a 24-dimensional vector representing the 24 hours of a day is applied here. The hours the slice took place in are set to 1, all others to 0. A 12-dimensional one-hot encoding is used to represent the month the slice took place in.

We then deploy four feature extractors to generate the final feature vector for CC. $f(\cdot)$ is used to extract a feature vector from the slice image data. Following Li et al. (2021), the ResNet-34 (He et al., 2016) architecture is applied in this work as $f(\cdot)$ to extract a 512-dimensional feature vector for each slice. Due to ResNet-34 requiring a 224x224 input, we resize each slice to these dimensions. We further extract learnable features for the location, daytime and month encodings, using $l(\cdot)$, $d(\cdot)$ and $m(\cdot)$. Each of these three is formulated as a small, non-linear multi-layer perceptron, each consisting of three layers (first two ReLU-activated (Agarap, 2018), last one with linear activation). The number of neurons for each layer are 2 for $l(\cdot)$, 24 for $d(\cdot)$ and 12 for $m(\cdot)$, resulting in the final features having the same dimension as the input encodings. Thus, the final feature vector has a dimension of 512+2+24+12=550.

A.4 Contrastive Clustering Framework

We closely follow the framework and notation of Li et al. (2021), which our work is based on and an extension to. The approach is visualized in Fig. 3. Given a batch of samples, we first apply an individual, stochastic data augmentation T to every sample. An original sample and its augmentation are considered a positive pair. The exact construction of these augmentations is explained in the main text (Sec. 3). Then, we extract features for both the original samples and the augmentations using $f(\cdot)$, $l(\cdot)$, $d(\cdot)$ and $m(\cdot)$ as described in section A.3. Note that both paths use the same, shared weights. The result of this step are two feature matrices H^a and H^b for the original batch and the augmented one, respectively. The architecture is then split up into an instance head and a cluster head.

First, the instance head projects each row h_i^a and h_i^b of both feature matrices to z_i^a and z_i^b in a new feature space using $g_I(\cdot)$, which is an MLP consisting of two layers with an output feature dimension of 16. It then maximizes the similarity between positive pairs in this feature space using contrastive loss. All other remaining possible pairs between the original and the other samples in a batch are

considered negative pairs ($2N$-2 in number for a mini-batch size of N). The cosine distance s is used to calculate pair-wise similarity between two vectors v_1 and v_2:

$$s(v_1, v_2) = \frac{(v_1)(v_2)}{\|v_1\|\|v_2\|}. \tag{1}$$

For a single sample x_i^a, the loss is given by

$$\ell_i^a = -\log \frac{\exp(s(z_i^a, z_i^b)/\tau_I)}{\sum_{j=1}^{N}[\exp(s(z_i^a, z_j^a)/\tau_I) + \exp(s(z_i^a, z_j^b)/\tau_I)]}, \tag{2}$$

where τ_I is the temperature ("softness") parameter for the instance head and N is the mini-batch size. The instance head contrastive loss $\mathcal{L}_{ins}$ is then computed using all augmented samples (Li et al., 2021):

$$\mathcal{L}_{ins} = \frac{1}{2N}\sum_{i=1}^{N}(\ell_i^a + \ell_i^b). \tag{3}$$

The cluster head uses g_C (again a two-layer MLP) to project the features into vectors with the dimensionality of the cluster number M. Applying a softmax to these vectors, they can be interpreted as the probabilities of the given sample being assigned to each of the M clusters. The results of applying g_C for both the original samples and the augmentations are the matrices Y^a and Y^b. While the rows, as explained, are considered probabilities for each cluster for a single sample, each column can be considered a *cluster representation*. Analog to the instance head, the cluster head tries to maximize the similarity between each cluster representation for the original and augmented samples. Defining $\hat{y}_i^a$ as the representation of cluster i using the original sample (i-th row of Y^a), a positive pair is given by $(\hat{y}_i^a, \hat{y}_i^b)$ (where $\hat{y}_i^b$ indicates the augmented cluster representation). Again, all other possible pairs between $\hat{y}_i^a$ and all other cluster representations are considered negative pairs ($2M$-2 in number). The cluster head contrastive loss is then given by

$$\hat{\ell}_i^a = -\log \frac{\exp(s(\hat{y}_i^a, \hat{y}_i^b)/\tau_C)}{\sum_{j=1}^{M}[\exp(s(\hat{y}_i^a, \hat{y}_j^a)/\tau_C) + \exp(s(\hat{y}_i^a, \hat{y}_j^b)/\tau_C)]}, \tag{4}$$

with τ_C being the temperature parameter for the cluster head. Overall, the cluster head contrastive loss is then defined as

$$\mathcal{L}_{clu} = \frac{1}{2M}\sum_{i=1}^{N}(\hat{\ell}_i^a + \hat{\ell}_i^b) - H(Y), \tag{5}$$

with the cluster assignment probabilities entropy term $H(Y) = -\sum_{i=1}^{M}[P(\hat{y}_i^a)logP(\hat{y}_i^a) + P(\hat{y}_i^b)logP(\hat{y}_i^b)]$, where $P(\hat{y}_i^k) = \frac{1}{N}\sum_{t=1}^{N} Y_{ti}^k$ with $k \in \{a, b\}$. According to Hu et al. (Hu et al., 2017), this term prevents the collapse of the model to the trivial solution of all samples being mapped to the same cluster.

The model displayed in Fig. 3 is trained end-to-end using the objective function

$$\mathcal{L} = \mathcal{L}_{ins} + \mathcal{L}_{clu}. \tag{6}$$

A.5 Implementation and Training

$m(\cdot)$, $d(\cdot)$, $l(\cdot)$, $g_I(\cdot)$ and $g_C(\cdot)$ are each MLPs. Using the Notation of (Input-Output1-Activation1, Output1-Output2-Activation2,...), they are each defined as:

- $m(\cdot)$: (12-12-ReLU, 12-12-ReLU, 12-12-Linear)

- $d(\cdot)$: (24-24-ReLU, 24-24-ReLU, 24-24-Linear)
- $l(\cdot)$: (2-2-ReLU, 2-2-ReLU, 2-2-Linear)
- $g_I(\cdot)$: (550-512-ReLU, 512-16-Linear)
- $g_C(\cdot)$: (550-512-ReLU, 512-128-Softmax)

Model settings:

- Batch size: 128
- Optimizer: Adam (learning rate = 0.0003, weight decay = 0.0)
- $\tau_I = 0.5$
- $\tau_C = 1.0$
- $shift_{s\_range} = 0.1$
- $shift_{c\_range} = 0.05$
- $p_{month} = 0.25$
- Epochs trained: 160

Our work is based upon the code made publicly available by Li et al. (2021). PyTorch is used for the implementation of the neural network. The model was trained for 160 epochs, which took a week on an NVIDIA GeForce RTX 3090 GPU.

A.6 Further Results

A.6.1 Daytime Length

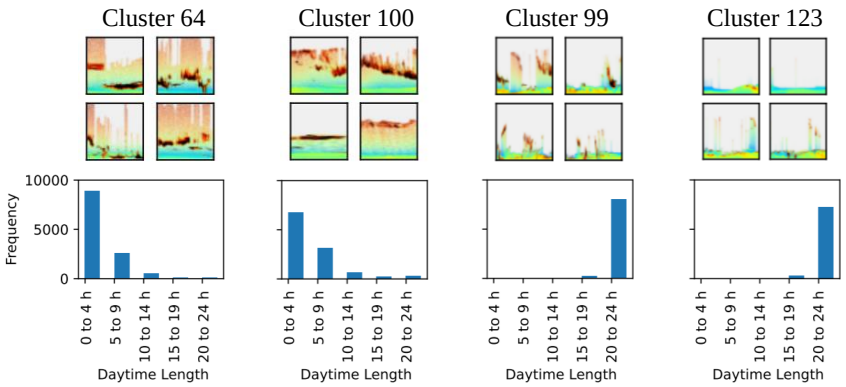


Figure 6: Examples for clusters that capture short and long slice durations.

Fig. 6 shows that different clusters capture phenomena that differ significantly in length. Cluster 64 and 100 capture phenomena that are rather short in length (mostly 0 to 4 hours). For cluster 64, these are low-altitude clouds, sometimes leading to precipitation; cluster 100, as discussed above, shows large clouds of higher altitudes than cluster 64. Cluster 99 and 123 are examples of clusters that capture mainly whole days. Cluster 99 shows typically mixed days, sometimes cloudy, possibly accompanied by short rain, then followed by a clear sky. Cluster 123 displays examples of overall clear days, with only a few clouds showing.

A.6.2 Seasonality

Fig. 7 displays cluster examples that group slices which accumulate in certain seasons of the year. Cluster 63, showing clear hours, is an example of a phenomenon that is not dependent on the season of the year. Cluster 56, fog, is very prevalent in the winter months. The phenomena in cluster 26, a PBL accompanied by low to mid-altitude clouds, take place mainly in the summer. Cluster 69, showing both clear sky and clouds, displays a peak in autumn. Cluster 40 (low backscattering PBL, clear sky, sometimes clouds) is quite evened out, but with a (small) maximum in spring.

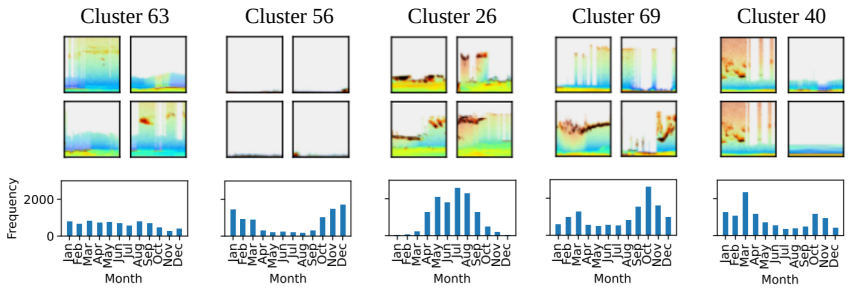


Figure 7: Examples for clusters that capture seasonality.

A.6.3 Location Dependency

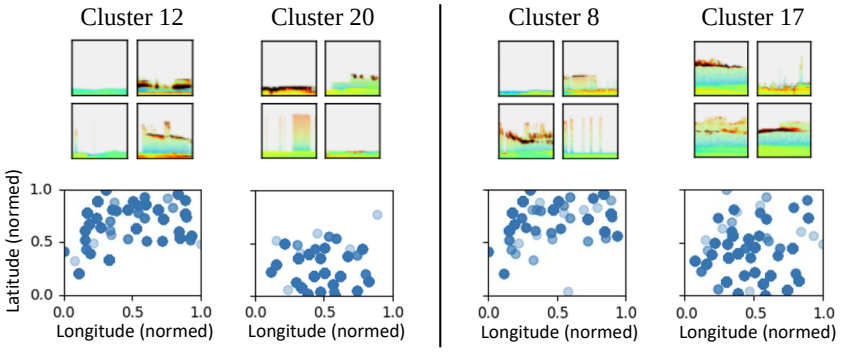


Figure 8: Clustering of ceilometer location.

Fig. 8 shows examples for clusters that group different latitudes. Cluster 12 and 20 both detect relatively similar phenomena (PBL, sometimes low-altitude clouds). However, cluster 12 is restricted to phenomena detected in the north, while cluster 20 groups slices in the south. A similar observation can be made for clusters 8 and 17. This could be a hint that the location information added to the multimodal CC framework proposed in this work indeed gets incorporated by the model to find good clusters.