2021 IEEE 33rd International Conference on Tools with Artificial Intelligence (ICTAI) | 978-1-6654-0898-1/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICTAI52525.2021.00064

# Consensus Subspace Clustering

Nathan Thom
*Computer Science and Engineering*
*University of Nevada, Reno*
Reno, USA
nathanthom@nevada.unr.edu

Hung Nguyen
*Computer Science and Engineering*
*University of Nevada, Reno*
Reno, USA
hungnp@nevada.unr.edu

Emily M. Hand
*Computer Science and Engineering*
*University of Nevada, Reno*
Reno, USA
emhand@unr.edu

***Abstract*—One significant challenge in the field of supervised deep learning is the lack of large-scale labeled datasets for many problems. In this paper, we propose Consensus Spectral Clustering (CSC), which leverages the strengths of convolutional autoencoders and spectral clustering to provide pseudo labels for image data. This data can be used as weakly-labeled data for training and evaluating classifiers which require supervision. The primary weaknesses of previous works lies in their inability to isolate the object of interest in an image and cluster similar images together. We address these issues by denoising input images to remove pixels which do not contain data pertinent to the target. Additionally, we introduce a voting method for label selection to improve the clustering results. Our extensive experimentation on several benchmark datasets demonstrates that the proposed CSC method achieves competitive performance with state-of-the-art methods.**

***Index Terms*—Cluster, Image, Autoencoder, Pseudo label**

## I. Introduction

The lack of large-scale, labeled datasets for many supervised learning problems has brought about new methods that aim to cluster data and provide pseudo labels which can be used to weakly train supervised classifiers. Clustering is a widely used method for data processing, with many well-known methods including k-means [1], Gaussian Mixture Models (GMMs) [2], [3], and spectral clustering [4]. These methods unfortunately struggle with high-dimensional data [5]. Our method overcomes the limitations of these methods by reducing the dimensionality of the input data while carefully selecting the features which are most important for differentiating between samples. Our multi-step clustering process, which we call Consensus Subspace Clustering (CSC) is detailed below:

1) An autoencoder is used to prune unnecessary features, capture spatial relationships of input images and flatten the data, resulting in the feature vector $v$.
2) Non-negative matrix factorization (NMF) is then applied to $v$ to filter out features that are not important for differentiating input samples producing $v_f$.
3) We build multiple low-dimensional representations from $v_f$ using a variational autoencoder (VAE) [6].
4) Spectral clustering is applied to our low-dimensional representations and consensus clustering [7] is used to stablize clustering results.

We introduce a novel method for unsupervised pseudo labeling, which we call Consensus Subspace Clustering (CSC). Experimentation shows that CSC produces results which are competitive with state-of-the-art methods.

This work was supported in part by NSF #1909707.

The novel contributions of this work are as follows:

- CSC utilizes a convolutional autoencoder and NMF to capture spatial relationships of input data and filter out features of the representation which are not valuable for separating the data into clusters.
- CSC learns multiple representations of the data to enable multiple rounds of clustering and voting. The purpose of this step is to reinforce the robustness of selected cluster labels.

## II. Related Work

The first class of methods that we describe jointly learn to compress images into dense representations and cluster the dense representations into classes. Fard et al. [8], [9] propose methods that tune an autoencoder to generate k-means friendly representations. In [10], Xie et al. pass samples through an encoder to generate representations, cluster with k-means and correct the cluster assignments with a clustering loss based on a KL divergence between soft assignments and their target distribution. Borrowing from Xie et al., [11] and [12] use the same learning framework with an undercomplete autoencoder to preserve the local structure of input data. Wang et al. [13] pass the input image through an orthogonal autoencoder prior to applying spectral clustering. Affeldt et al. [14] use multiple autoencoder architectures to generate multiple representations from the input data. The representations are then clustered with spectral clustering. The authors of [15] propose an architecture in which a neural network reduces the dimension of input images. The learned representations are clustered and the corresponding pseudo labels are used as supervision for training the network.

The second group of works that we highlight are miscellaneous techniques for improving cluster performance. Li et al. [16] use a boosting method to train on easier samples, then gradually expose the model to more challenging data. [17] utilizes an ensemble of classifiers to generate cluster assignments and compute a similarity graph. Finally the similarity graph is pruned to extract high confidence cluster assignments. [18] uses a modified VAE in which the latent space is sampled from a mixture of Gaussian distributions. Clustering is achieved by calculating how far the mixture distribution is from the

978-1-6654-0898-1/21/$31.00 ©2021 IEEE
DOI 10.1109/ICTAI52525.2021.00064

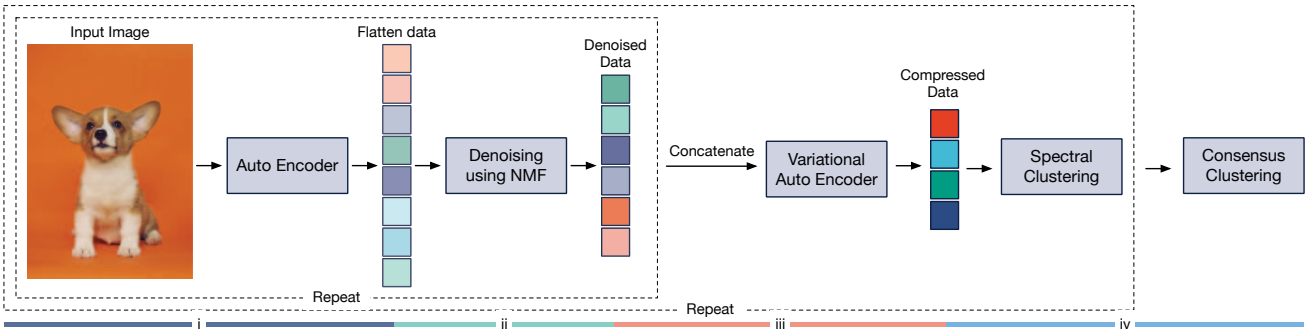


Fig. 1. Overview of the proposed Consensus Subspace Clustering (CSC) pipeline. The method consists of four main modules: i) a flattening module using an autoencoder to extract features from input images, ii) a denoising module using NMF to remove unimportant features, iii) a compression module using VAE to generate a low-dimensional representation of denoised features and iv) a clustering module using spectral clustering to cluster images from their compressed representations.

normal distribution. Lastly, Li et al. [19] implement multi-view autoencoders for multi-view data with shared weights. Their network structure has a deep embedding clustering layer which recalculates cluster centers each iteration.

The proposed CSC differs from prior work in that the autoencoder is primarily used for flattening the data and capturing spatial relationships. In addition, related works do not filter out features of the learned representation which are not valuable for separating input data into clusters. Lastly, CSC utilizes consensus clustering to stabilize clustering results.

## III. Methods

The Consensus Subspace Clustering (CSC) pipeline consists of four core modules as shown in Fig.1. The first module aims to extract features from input images using an autoencoder. The second module removes noise and unimportant features from input images. This is accomplished by detecting meta-features with NMF and inspecting errors in the reconstructed data. CSC only keeps features that have significant contributions to the reconstruction error because these are likely the features which separate each class from the others. The first and second module are repeated to generate multiple denoised versions of the input images. The third module is a VAE that projects the denoised features from the second module into multiple lower-dimensional representations. The fourth module applies spectral clustering to the low-dimensional representations. All four modules are repeated to generate multiple cluster assignments for each image. Finally, an ensemble approach is used across multiple clustering runs to determine the final cluster assignment for each image. This is accomplished by using the cluster assignments obtained from each representation. We detail each of the four modules in the following subsections.

### A. Feature extraction

We scale the range of pixel values in each image from 0 to 1 using min-max scaling. After normalization, a 1-layer convolutional autoencoder is used to extract features from the input images (Fig.2). The encoder consists of one convolutional layer and the decoder consists of one deconvolutional layer. For each image, we obtain 500 features from the bottleneck layer of the autoencoder. Optimizing this model so that it can generate a good, compact representation requires that we identify the significant features of input images. The following section details our denoising module.

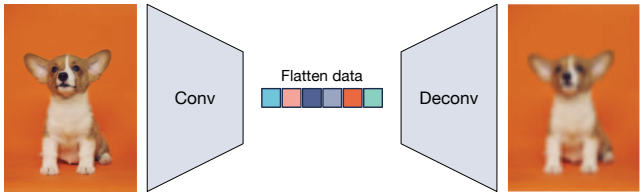


Fig. 2. Feature extraction using our autoencoder. A 1-layer autoencoder is used to extract features from input images. The representation generated by the autoencoder has 500 dimensions.

### B. Denoising Module

We expect that only a subset of features extracted in the first step are useful for clustering images into different groups. Therefore, we focus on filtering out features that are not likely to play a major role in clustering. Figure 3 shows the workflow of our feature filtering approach using 1-factor NMF.

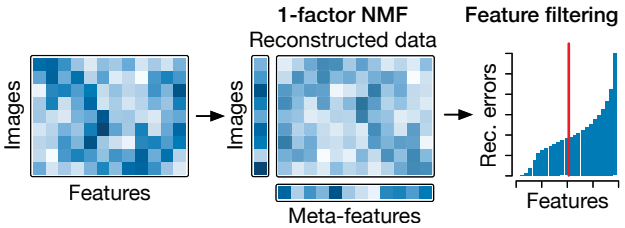


Fig. 3. Denoising extracted features from input images using NMF. The original data matrix is decomposed into two vectors representing images and their features in 1-dimensional latent space. The error of the reconstructed data using these two vectors is used to rank each feature. Only 50% of features that have the largest error are kept for the next steps.

Briefly, Matrix Factorization is a technique that decomposes a matrix into the product of two lower dimensional matrices, $W$ and $H$ with reconstruction error $E$: $V = W \times H + E$ where $E$ is a matrix of size $m \times n$, the error between the original data and the reconstructed data from $W$ and $H$.

In this model, we set the number of factors $k = 1$. This makes it very difficult to fit the model for the features which are significantly different between clusters. In other words, the features most valuable for the clustering task. Therefore, when we attempt to reconstruct the original matrix $V$, we can select the features most important for clustering by selecting those which have the highest reconstruction error [20], [21]. We do this selection by sorting the features by their absolute error and removing 50% of the features with lowest error. Since both the feature extraction and denoising modules are non-deterministic and can be sensitive to random factors, we repeat these two modules ten times to obtain different denoised versions of the data. The concatenation of these denoised features are passed to the next step.

### C. Variational Autoencoder

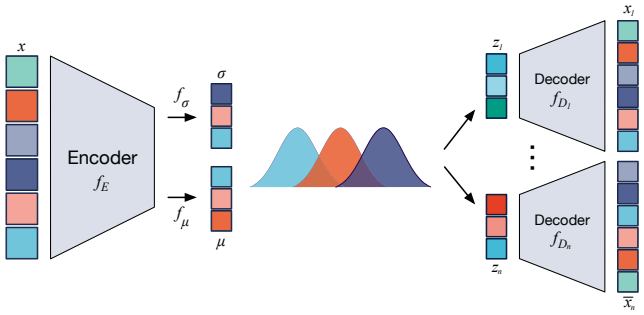


Fig. 4. Compressing images using a VAE. Denoised images are compressed into multiple representations using a VAE. Multiple representations are obtained from one image. This is accomplished by adding different noise into the latent space and the use of multiple decoders to reconstruct the image. The representations of each image are used for clustering.

The previous step has removed insignificant features from the original extracted features, but the dimensions of the remaining features are still too large (2,500 features) to perform clustering efficiently. Hence, a VAE is applied to compress the significant features into a lower dimension (Fig.4). The VAE architecture is similar to that of a standard autoencoder. However, rather than attempting to encode each input sample into fixed floating point features, the VAE encodes features as various Normal distributions. The result of this technique is a smooth and continuous latent space. The input of the decoder is sampled from the latent space by adding a small random noise into the latent space. Autoencoders are, however, prone to overfitting [22]. Therefore, instead of using one decoder as in a standard VAE, we use multiple decoders in our implementation to ensure that the encoder learns the generalized presentation of the input. At the end of this modules, we obtain multiple (three) compact representations for each image by repeatedly sampling from the latent space. These compact representations are used for spectral clustering in the next step.

### D. Basic Subspace clustering

After training the VAE, the final informative and dense representations of the input images are obtained. Next, spectral clustering is performed on each subspace to form pseudo labels for the input data (i.e. each cluster represents a class). We utilize spectral clustering rather than k-means because spectral clustering is expected to capture better non-linear relationships among input images.

In our pipeline, we use the K-Means adaptation of spectral clustering, proposed by Ng et al. [23], to generate pseudo labels for input images. The clustering procedure first computes the similarity matrix for all samples to use as the input graph. It then computes the symmetric and normalized Laplacian matrix ($L^{sym}$). Then, the $K$ largest eigenvectors for $L^{sym}$, are computed and normalized to unit length. The eigenvectors are then used to make up the columns of a matrix. Finally, the algorithm uses K-means clustering to segment the subspace into $K$ clusters.

To select the optimal number of clusters, we run the algorithm with a different number of clusters and select the clusters that give us the best ratio $r$ of between-sum-of-squares and total-sum-of-squares by cluster. Since the input data can be large, for each number of clusters, we subsample the input multiple times and perform clustering to obtain multiple $r$. We take the average of all $r$ for each $k$ and select the optimal number of clusters $K$ such that $r$ is maximized.

### E. Consensus Clustering

We repeat the clustering pipeline multiple times (ten in our experiments) to obtain multiple cluster assignments for each image. To generate the final cluster assignment for each image, we adopt an ensemble clustering strategy called weighted-based meta-clustering (wMetaC) [24]. wMetaC was originally developed to combine clustering results from random projections for single-cell data (gene expression data of individual cells) to separate cells into different cell types. Here, we use this strategy in an ensemble clustering approach to combine multiple cluster assignments for each image. In this work, our method uses voting from each cluster assignment to determine the final clusters. First, an image-image similarity matrix is computed such that each value in the matrix represents the likelihood that two images are clustered together. Next, each image is assigned a weight by summing all of the pairs that an image appears in. These similarity matrices are used to form a cluster-to-cluster similarity matrix. Lastly, final clusters are selected by performing hierarchical clustering on the cluster-cluster similarity matrix.

## IV. Experiments and Results

To evaluate our proposed method, we compare CSC with several existing clustering methods on two different handwritten digit datasets and one general object classification dataset. Baseline methods included in our comparison are k-means,

Deep Cluster [25], and Deep k-means [9]. The datasets used for experimentation are MNIST [26], USPS [27], and CIFAR-10 [28]. Widely used performance metrics are computed to compare CSC to baseline techniques and state-of-the-art methods.

### A. Datasets

The datasets that we select for evaluation are MNIST, USPS and CIFAR-10. Each of these collections are relatively small and contain low-resolution images (32x32 pixels or less). The MNIST dataset contains a total of 70,000 images of size 28x28 (60,000 images for training and 10,000 images for testing). MNIST is relatively balanced with each of the 10 classes representing close to 10 percent of the total population. The group with most representation makes up 11.25 percent and the group with least representation makes up 9 percent. USPS contains a total of 11,000 images with of size 16x16. Both datasets have 10 classes, which correspond with the integers ranging from 0 to 9. Each image depicts a hand-written digit. USPS is mostly balanced with the largest group representing 17 percent and the smallest group representing 8 percent. The CIFAR-10 dataset contains total of 60,000 images of size 32x32x3 (50,000 images for training and 10,000 images for testing). This dataset is balanced, with 6000 images per class. CIFAR-10 provides a much more challenging task due to significantly larger feature space and diverse class labels: airplane, automobile, bird, cat, deer, dog, frog, horse, ship and truck.

### B. Methods for Comparison

Effective evaluation of CSC is achieved via comparison to state-of-the-art methods in the field. In addition, we select k-means as a baseline model. Images are flattened before being passed to k-means. K-means is run with 10 cluster centers for a maximum of 1000 iterations or until convergence. We run k-means 20 times on each dataset and select the run with best results for comparison. The selected state-of-the-art methods are Deep Cluster [25] and Deep k-means [9]. Results shown in Table I are those reported in each publication.

### C. Metrics

We use Accuracy (ACC) and Normalized Mutual Information (NMI) as metrics to evaluate performance of each method. Accuracy and NMI metrics are used to be consistent with the evaluations in the original papers of corresponding methods included in the comparison. The metrics are calculated as follow:

$$ACC = max_m \frac{\sum_{n=1}^{N} \mathbf{1}(l_i = m(c_i))}{N}$$

where $\mathbf{1}(.)$ is an indicator function, $l_i$ is the true label, $c_i$ is the label assigned by the clustering method and $m(.)$ denotes all possible one-to-one mappings between clusters.

$$NMI = \frac{I(l, \mathbf{c})}{(H(\mathbf{l}) + H(\mathbf{c}))/2}$$

where $\mathbf{l}$ denotes the ground truth labels, $\mathbf{c}$ is the cluster assignments, $I(.)$ is the mutual information metric, and $H(.)$ is the entropy.

### D. Results

Table I shows the Accuracy and NMI for CSC and comparison methods on the MNIST, USPS and CIFAR-10 datasets. On the MNIST task, CSC far exceeds performance of the baseline and outperforms the other methods in accuracy. Deep Cluster reports slightly better NMI for MNIST and Deep K-means outperforms CSC in both metrics on the USPS dataset. In the case of Deep Cluster, the margin of difference is very slight and shows that CSC is competitive with state-of-the-art on this task. Regarding Deep k-means, we believe that the architecture is better suited for the smaller feature space found in USPS. Each image in this dataset contains a total of only 256 features. To reinforce this claim, we point to the method's decreased performance on the larger MNIST and CIFAR-10 datasets. We note that the authors of Deep Cluster and Deep K-means did not evaluate their methods on the CIFAR-10 dataset.

TABLE I
PERFORMANCE OF K-MEANS, DEEP CLUSTER, DEEP K-MEANS, AND CSC ON MNIST, USPS AND CIFAR-10 DATASETS.

| Method | MNIST | | USPS | | CIFAR-10 | |
|---|---|---|---|---|---|---|
| | ACC | NMI | ACC | NMI | ACC | NMI |
| K-means | 0.58 | 0.49 | 0.48 | 0.42 | 0.14 | **0.12** |
| Deep Cluster | **0.86** | **0.83** | 0.67 | 0.69 | — | — |
| Deep K-means | 0.84 | 0.80 | 0.76 | 0.78 | — | — |
| CSC No Flatten | 0.85 | 0.79 | 0.83 | 0.78 | 0.12 | 0.08 |
| CSC No Filter | 0.83 | 0.76 | **0.84** | **0.79** | 0.14 | 0.10 |
| CSC No Voting | 0.82 | 0.77 | 0.82 | 0.76 | 0.14 | 0.10 |
| CSC | **0.86** | 0.81 | 0.83 | **0.79** | **0.15** | 0.11 |

Complete analysis of CSC requires an understanding of how each component in the pipeline effects the end performance of the model. Referencing the latter half of Table I, removing the flattening module reports the least change out of all modules. However, flattening appears to become more important as the complexity of the dataset increases. Next, the filtering module is particularly important for MNIST, but less important for USPS. This is likely because the samples in USPS are mostly separated before being processed by the VAE, see Figures 5 and 6. Last, voting or consensus clustering is very important for stability of clustering results. In our trials without voting, results can be extremely variable.

## V. CONCLUSION

In this work, we have introduced a novel method for providing pseudo labels on arbitrary image data, which we call Consensus Subspace Clustering (CSC). To the best of our knowledge we are the first to present a deep clustering method which removes inconsequential features from input data and learns multiple representations of the data to reinforce the robustness of selected cluster labels. Our experimentation shows that our work is competitive with, and in some cases,

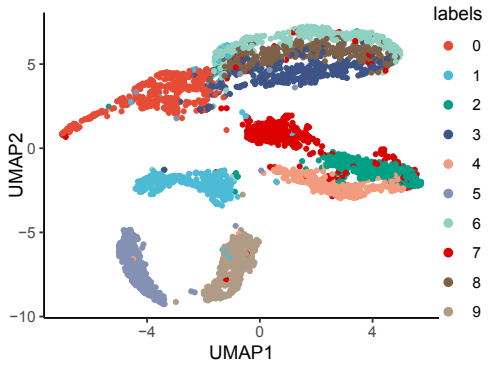


Fig. 5. A UMAP [29] visualization of the raw USPS dataset. Each colored dot represents an input sample.

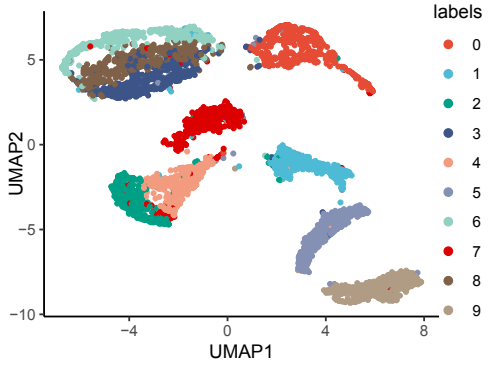


Fig. 6. A UMAP [29] visualization of the USPS dataset after it is processed with CSC. Each colored dot represents a latent representation of an input sample from the dataset. Note that the points within each cluster tighten together and the clusters are separated by a greater margin than those that appear in Fig. 5.

exceeds the state-of-the-art for deep clustering of image data. Future work in this area could introduce a confidence measure to the sample in each cluster. Additionally the method could be expanded to process data beyond images.

## References


[1] J. B. MacQueen, "Some methods for classification and analysis of multivariate observations," 1967.

[2] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 39, no. 1, pp. 1–22, 1977. [Online]. Available: https://rss.onlinelibrary.wiley.com/doi/abs/10.1111/j.2517-6161.1977.tb01600.x

[3] G. J. McLachlan, S. X. Lee, and S. I. Rathnayake, "Finite mixture models," *Annual Review of Statistics and Its Application*, vol. 6, no. 1, pp. 355–378, 2019. [Online]. Available: https://doi.org/10.1146/annurev-statistics-031017-100325

[4] A. Y. Ng, M. I. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," in *Proceedings of the 14th International Conference on Neural Information Processing Systems: Natural and Synthetic*, ser. NIPS'01. Cambridge, MA, USA: MIT Press, 2001, p. 849–856.

[5] M. Freimer and R. Bellman, "Adaptive control processes: A guided tour." *Journal of the American Statistical Association*, vol. 60, p. 383, 1965.

[6] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[7] S. Monti, P. Tamayo, J. Mesirov, and T. Golub, "Consensus clustering: A resampling-based method for class discovery and visualization of gene expression microarray data," *Mach. Learn.*, vol. 52, no. 1–2, p. 91–118, Jul. 2003. [Online]. Available: https://doi.org/10.1023/A:1023949509487

[8] B. Yang, X. Fu, N. D. Sidiropoulos, and M. Hong, "Towards k-means-friendly spaces: Simultaneous deep learning and clustering," in *international conference on machine learning*. PMLR, 2017, pp. 3861–3870.

[9] M. M. Fard, T. Thonet, and E. Gaussier, "Deep k-means: Jointly clustering with k-means and learning representations," *Pattern Recognition Letters*, vol. 138, pp. 185–192, 2020.

[10] J. Xie, R. B. Girshick, and A. Farhadi, "Unsupervised deep embedding for clustering analysis," *CoRR*, vol. abs/1511.06335, 2015. [Online]. Available: http://arxiv.org/abs/1511.06335

[11] X. Guo, L. Gao, X. Liu, and J. Yin, "Improved deep embedded clustering with local structure preservation," in *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, 2017, pp. 1753–1759. [Online]. Available: https://doi.org/10.24963/ijcai.2017/243

[12] X. Guo, X. Liu, E. Zhu, and J. Yin, "Deep clustering with convolutional autoencoders," in *Neural Information Processing*, D. Liu, S. Xie, Y. Li, D. Zhao, and E.-S. M. El-Alfy, Eds. Cham: Springer International Publishing, 2017, pp. 373–382.

[13] W. Wang, D. Yang, F. Chen, Y. Pang, S. Huang, and Y. Ge, "Clustering with orthogonal autoencoder," *IEEE Access*, vol. 7, pp. 62 421–62 432, 2019.

[14] S. Affeldt, L. Labiod, and M. Nadif, "Spectral clustering via ensemble deep autoencoder learning (sc-edae)," *Pattern Recognition*, vol. 108, p. 107522, 2020.

[15] M. Caron, P. Bojanowski, A. Joulin, and M. Douze, "Deep clustering for unsupervised learning of visual features," *CoRR*, vol. abs/1807.05520, 2018. [Online]. Available: http://arxiv.org/abs/1807.05520

[16] F. Li, H. Qiao, and B. Zhang, "Discriminatively boosted image clustering with fully convolutional auto-encoders," *Pattern Recognition*, vol. 83, pp. 161–173, 2018.

[17] D. Gupta, R. Ramjee, N. Kwatra, and M. Sivathanu, "Unsupervised clustering using pseudo-semi-supervised learning," in *International Conference on Learning Representations*, 2020. [Online]. Available: https://openreview.net/forum?id=rJlnxkSYPS

[18] N. Dilokthanakul, P. A. M. Mediano, M. Garnelo, M. C. H. Lee, H. Salimbeni, K. Arulkumaran, and M. Shanahan, "Deep unsupervised clustering with gaussian mixture variational autoencoders," *CoRR*, vol. abs/1611.02648, 2016. [Online]. Available: http://arxiv.org/abs/1611.02648

[19] Z. Li, Q. Wang, Z. Tao, Q. Gao, and Z. Yang, "Deep adversarial multi-view clustering network," 08 2019, pp. 2952–2958.

[20] S. Wang, W. Pedrycz, Q. Zhu, and W. Zhu, "Subspace learning for unsupervised feature selection via matrix factorization," *Pattern Recognition*, vol. 48, no. 1, pp. 10–19, 2015.

[21] F. Saberi-Movahed, M. Eftekhari, and M. Mohtashami, "Supervised feature selection by constituting a basis for the original space of features and matrix factorization," *International Journal of Machine Learning and Cybernetics*, vol. 11, no. 7, pp. 1405–1421, 2020.

[22] H. Steck, "Autoencoders that don't overfit towards the identity," in *NeurIPS*, 2020.

[23] A. Ng, M. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," *Adv. Neural Inf. Process. Syst*, vol. 14, 04 2002.

[24] S. Wan, J. Kim, and K. Won, "Sharp: hyper-fast and accurate processing of single-cell rna-seq data via ensemble random projection," *Genome Research*, vol. 30, p. gr.254557.119, 01 2020.

[25] K. Tian, S. Zhou, and J. Guan, "Deepcluster: A general clustering framework based on deep learning," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2017, pp. 809–825.

[26] Y. LeCun, "The mnist database of handwritten digits," *http://yann. lecun. com/exdb/mnist/*, 1998.

[27] J. J. Hull, "A database for handwritten text recognition research," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, no. 5, pp. 550–554, 1994.

[28] A. Krizhevsky, "Learning multiple layers of features from tiny images," 2009.

[29] L. McInnes, J. Healy, and J. Melville, "Umap: Uniform manifold approximation and projection for dimension reduction," 2020.