Multi-Modal Deep Clustering: Unsupervised Partitioning of Images

By

GUY SHIRAN

Under the supervision of

PROF. DAPHNA WEINSHALL

THE HEBREW
UNIVERSITY
OF JERUSALEM

Faculty of Computer Science and Engineering
THE HEBREW UNIVERSITY OF JERUSALEM

A dissertation submitted to the Hebrew University of Jerusalem as a partial fulfillment of the requirements of the degree of MASTER OF SCIENCE in the Faculty of Computer Science and Engineering.

MARCH 2021

ABSTRACT

The clustering of unlabeled raw images is a daunting task, which has recently been approached with some success by deep learning methods. Here we propose an unsupervised clustering framework, which learns a deep neural network in an end-to-end fashion, providing direct cluster assignments of images without additional processing. Multi-Modal Deep Clustering (MMDC), trains a deep network to align its image embeddings with target points sampled from a Gaussian Mixture Model distribution. The cluster assignments are then determined by mixture component association of image embeddings. Simultaneously, the same deep network is trained to solve an additional self-supervised task of predicting image rotations. This pushes the network to learn more meaningful image representations that facilitate a better clustering. Experimental results show that MMDC achieves or exceeds state-of-the-art performance on six challenging benchmarks. On natural image datasets we improve on previous results with significant margins of up to 20% absolute accuracy points, yielding an accuracy of 82% on CIFAR-10, 45% on CIFAR-100 and 69% on STL-10.

Acknowledgements

I would like to thank my academic advisor, Prof. Daphna Weinshall, who has guided me along this journey. I am grateful for your support and the academic freedom you encouraged that allowed me to pursue my research interests. I have learned a lot throughout this process. I also wish to thank my family and friends, for their help and encouragements were greatly appreciated.

Table of Contents

List of Tables

Table **Page**

List of Figures

Figure **Page**

CHAPTER 1

INTRODUCTION

Clustering involves the organization of data in an unsupervised manner, based on the distribution of datapoints and the distances between them. Since these properties are closely tied to the representation of the data, the problems of clustering and data representation are firmly connected and are therefore sometimes solved jointly. In accordance, in this work we start from a recent method for the unsupervised computation of effective data representation (or features discovery), and develop a clustering method whose results significantly improve the state of the art in the clustering of natural images.

The task of unsupervised image clustering is challenging and interesting, as the algorithm needs to discover patterns in highly entangled data, and produce separated groups without explicitly specifying the grouping features. A large body of work has been devoted to the problem of clustering [26], see Chapter 2 for a review of recent related work. In recent years, with the emergence of deep learning as the method of choice in visual object recognition and image classification, emphasis has shifted to the computation of effective representations that will support successful clustering [37]. Vice versa, unsupervised clustering loss has been used to drive the computation of image representation and the discovery of enhanced image features by making it possible to use unsupervised data in the training of deep networks, which traditionally require massive amounts of labeled data.

When learning feature representation from unsupervised data by minimizing a clustering-based loss function, one danger is cluster collapse - the representation may

collapse to the trivial solution of a single cluster. In [6], a similar problem of representation collapse is managed by mapping the network's representation to a fixed set of randomly chosen points in some target features space. Here we borrow this mapping idea, and incorporate it into a clustering algorithm.

More specifically, we first sample a fixed set of points in some target space. Since our method is designed to partition the data into k clusters, the target points are chosen from a matched density function - Gaussian Mixture Model (GMM) with k components. Our model trains a randomly initialized neural network to align its image embeddings with the sampled target points, directly inducing a partition that is based on the mixture components. This is done by simultaneously learning a one-to-one mapping between the output of the network and the target points, and updating the networks parameters to best fit images with their target points as assigned by the mapping.

In the absence of ground truth, the proposed approach is prone to instability as target points are continuously reassigned between images, creating a non-stationary online learning environment. Such instability is often linked with unsupervised learning tasks. To alleviate this problem, unsupervised tasks such as representation learning may be combined with self-supervision tasks to achieve better results [14]. Here we adopt the approach taken by [9] to deal with the notorious instability of training generative adversarial networks. Thus the model is jointly trained on the main clustering task and on a self-supervised auxiliary task as defined in RotNet [20], where all images are subjected to 4 rotation angles. In this auxiliary task the network is trained to recognize the $2D$ rotation of each rotated image.

For computation engine, our method uses off the shelf ConvNets and standard SGD training with mini-batch sampling in an end-to-end fashion. It is therefore scalable to large datasets.

1.1 Contributions

Our main contributions in this study are the following:
I) Introduce a novel clustering framework, Multi-Modal Deep Clustering (MMDC), which partitions unlabeled images into semantically meaningful groups by training a ConvNet to align image embeddings with fixed targets sampled from a Gaussian Mixture Model.
II) Propose incorporating an auxiliary self-supervised representation learning task of predicting image rotations, to enhance the ConvNets features and consequentially facilitate a more accurate clustering.

III) Apply our method to a variety of datasets and demonstrate a significant improvement to previous state-of-the-art in common image clustering benchmarks, further closing the performance gap between the realms of unsupervised and supervised learning.

CHAPTER 2

BACKGROUND & RELATED WORK

In this chapter we broadly review the topic of unsupervised data clustering and the relations of clustering to data representation. We then survey recent advancements in unsupervised image representation learning and image clustering using deep neural networks.

2.1 Background

Clustering is a form of unsupervised learning, a machine learning paradigm that aims to extract patterns from unlabeled data. In a typical supervised learning setting, of either a classification or regression task, an algorithm is shown pairs of data points and labels and must learn to predict the labels of new unseen data points. In contrast, a clustering algorithm is shown a set of unlabeled data points and its goal is to determine cluster assignments for each one of the data points. This process can sometimes be thought of as unsupervised classification.

The objective of data clustering is to partition data points into groups such that points in each group are more similar to each other than to data points in the other groups. The measure of similarity between data points is heavily tied with how the data is represented and what similarity metric is used. Expectantly, these choices can greatly impact the efficacy of the clustering result. In practice, there is no one size fits all solution. Different clustering algorithms differ by what assumptions they have on the data points, how they measure similarity and in what way they approach the building of

the clusters. Prior knowledge of the characteristics of the data and clear definition of the desired outcome can guide a wiser choice of a compatible algorithm for a specific task.

The problem of data clustering has received much attention in past decades and many algorithms have been proposed to tackle it [26]. A coarse distinction to be made is between *Hierarchical* algorithms and *Partitional* algorithms. Hierarchical clustering approaches yield a dendogram of nested groups of clusters, i.e. a hierarchy of clusters where on the top the whole dataset is clustered together and at the bottom each data point is in a cluster of its own. Agglomerative clustering [22] builds the clusters in a bottom-to-up fashion and Divisive clustering [44] constructs the clusters in a top-to-bottom order. Partitional clustering algorithms [18] produce a single partition of the data by minimizing a given clustering criterion, where usually a locally optimal solution is achieved. In contrast to the hierarchical approach, typically a choice of the specific number of desired clusters has to be made and is given as input to the algorithm. As hierarchical algorithms produce many partitions of the data, they often require more computational resources and are less scalable to large datasets than partitional methods. Our work falls in the Partitional approach, as such we will continue mainly focusing on it.

2.1.1 Partitional Clustering

Partitional clustering algorithms produce a single partition of the data points. In most cases, the desired number of clusters is given as input to the algorithm. An exception to this is density based approaches, which will be discussed later. There are two complementary approaches for choosing k, the number of clusters. The first, and the more subjective one, is human evaluation. A domain expert can have prior knowledge on the expected number of natural clusters in the data and can also manually observe several different partitions of the data and select one that appears most fitting. A second approach is to use principled evaluation metrics and visualization tools such as the Elbow method [28] or Silhouette [43]. For each k, clustering is performed and some value is calculated and a choice of k is determined by plotting the values and using a visual heuristic.

Centroid-based Clustering

In centroid-based clustering, commonly known as k-means clustering, clusters are represented by a centroid vector and each data point is assigned to the cluster of its nearest centroid. A typical clustering criterion can be formulated as

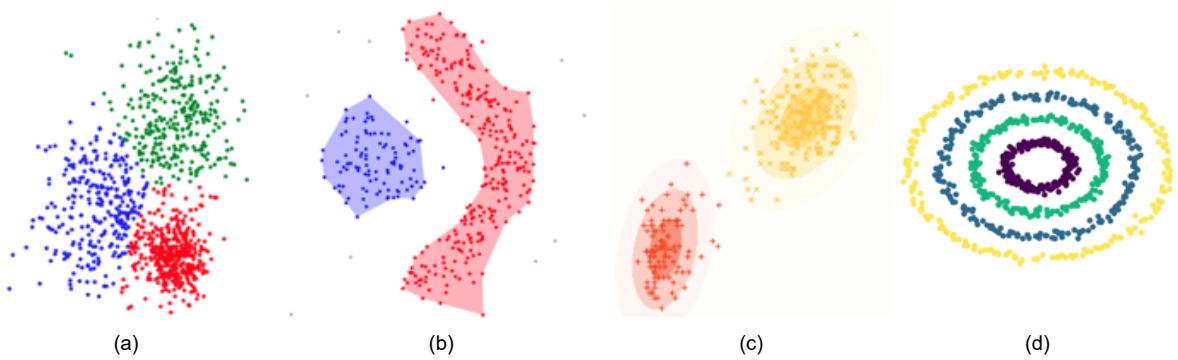

Figure 2.1: Illustration of different clustering approaches. (a) Centroid based methods, such as k-means, associate each cluster with a centroid vector. (b) Density based methods, such as DBSCAN, define dense areas as clusters and sparse areas as the buffer between clusters. (c) Distribution based methods, such as Gaussian Mixture Model, fit the data points to a parameterized distribution. (d) Spectral clustering methods make use of the spectrum (eigenvalues) of the similarity matrix of the data to perform dimensionality reduction before clustering in fewer dimensions.

$$\min_{c_1,\dots,c_K} \sum_{j=1}^{K} \sum_{i=1}^{n_j} d(x_i, c_j) \tag{2.1}$$

$\{x_i\}_{i=1}^{n}$ are the data points we wish to clusters, $\{c_j\}_{j=1}^{K}$ are the cluster centroids we aim to find, n_j is the number of data points in cluster $j \in [K]$ and $d(\cdot,\cdot)$ is a distance function.

Solving this optimization problem directly is known to be NP-Hard. An approximate solution can be found using Llyod's algorithm [34], commonly referred to as k-means. Initially, data points are assigned random clusters. In each step, new cluster centroids are calculated as the average of all vectors currently assigned to the cluster and then each data point is reassigned to the cluster of its nearest centroid. This is repeated iteratively until convergance. Although this process leads to a local optimum of the k-means objective, in practice it quite often produces pleasing results. Variations of the original algorithm include optimizations such as running the algorithm several times and choosing the partition with the best k-means score or initializing the cluster assignments using better heuristics [3, 40].

Distribution-based Clustering

A clustering approach that is closely tied to statistics is distributed-based clustering. Data points most likely belonging to the same distribution are clustered together. The

most prominent model is the Gaussian Mixture Model, where each cluster is assumed to be distributed normally and the number of clusters is equal to the number of gaussians in the mixture. After the parameters of the mixture model is determined, each data point is assigned to the gaussian it most likely belongs to. Naturally, soft cluster assignments are obtained as well. Determining the mixture models parameters is commonly done using the Expectation-Maximization (EM) algorithm [11]. First, the distributions parameters θ are initialized to some random values. Then a two-step procedure is iteratively repeated until convergence. In the first step, each data point is assigned the gaussian it has most likely come from. In the second step, the gaussians parameters θ are updated to maximize the likelihood of the data points.

Density-based Clustering

In the paradigm of density-based approaches, clusters are defined as areas with a high density of data points, separated by areas with low density of data points which are considered noise. Density-based spatial clustering of applications with noise (DBSCAN) [17, 48] is perhaps the most popular algorithm in this class. In contrast to other partitional clustering algorithms, the number of desired clusters k is not required as input. Substituting, are the impactful parameters ϵ and $minPts$. A data point is considered to be in a cluster if there are at least $minPts$ other points within a radius of ϵ from it. DBSCAN is one of the most commonly used clustering algorithms thanks to its rather efficient $O(n^2)$ complexity and its ability to discover clusters of various shapes and sizes. Ordering points to identify the clustering structure (OPTICS) [2] adapts the density-based approach to a hierarchical framework where a hierarchy of clusters is produced. In DeLi-Clu [1], further improvments are made and the need for the ϵ parameter is eliminated.

Spectral Clustering

The general scheme of Spectral Clustering algorithms is to perform dimensionality reduction on the data points prior to clustering using standard methods (such as k-means) by making use of the spectrum of the similarity matrix of the data. Given a similarity matrix of the data, first a Laplacian matrix L is calculated. Then the first k eigenvectors, corresponding to the k smallest eigenvalues, of L are calculated and are used to form a new matrix where each row corresponds to the features of a specific data point. Finally some other clustering algorithm is applied on the matrix rows. Different

spectral clustering algorithms [38, 45] mainly differ by how they define and interpret the Laplacian L.

2.2 Related Work

2.2.1 Unsupervised Representation Learning

Naïvely attempting to cluster images with traditional approaches does not produce a pleasing partitions of the images, as they work on the raw representations of the images in pixel space, whereas semantically similar images are not necessarily similar, by standard metrics such as the euclidean distance, in the high-dimensional pixel space in which the images reside. Learning effective representations of data is crucial for the success of machine learning tasks such as classification and clustering [4]. In recent years learning useful image representations in an unsupervised manner has been dominated by deep-learning-based approaches.

Autoencoders (AEs) [5] encode images with a deep network and are trained by reconstructing the image using a decoder network. Different variations typically supplement the reconstruction loss with regularization terms that encourage the latent space of the autoencoder to have specific properties that are believed to facilitate better representations. Sparse autoencoders apply $L1$ regularization on the output of the encoder to encourage the optimization process to produce sparse representations. Denoising autoencoders [47] add gaussian noise to the encoders input in the training phase. Many more variations exist [36, 54].

Generative models such as Generative Adversarial Networks (GAN) [21] and variational autoencoders (VAE) [29] learn representations as a byproduct of learning to generate images. One may simply use the discriminator of a GAN framework as a feature extractor [42]. In a more principled manner, BiGANs [15, 16] learn an inverse mapping $f : X \to Z$ that encodes images into the generators latent space.

Tightly connected to our work, Noise-As-Targets (NAT) [6] and DeepCluster [7] adopt a training strategy of iteratively reassigning psuedo-labels to date points while training the network to fit them. In [6] each data point is assigned a unique target sampled uniformly from the unit sphere, whilst [7] produces psuedo-labels by applying k-means clustering on the activations of the penultimate layer of the network and using the cluster assignments as psuedo-labels for the data points. We expand on these methods in chapter 3.

2.2.2 Self-Supervised Learning

A family of unsupervised learning algorithms that gained popularity in recent years are self-supervised methods. In the self-supervised learning framework, labels are automatically generated from the data and are used to train a parameterized model. In the context of unsupervised representation learning, this is typically done by training a deep network to solve a pretext task, where labels can be produced directly from the data. The task is designed in such a way that solving it implicitly requires the network to learn semantically meaningful features of the data. In [39] a network is tasked with solving a jigsaw puzzle. Given an unlabeled image, non-overlapping tiles are extracted and shuffled and the network is required to predict the correct order of the tiles. [41] proposes the pretext task of generating image regions conditioned on their surroundings. A region within an image is masked out and an autoencoder is trained to fill in the missing pixels. In [13] image patches representations are learnt by predicting the relative position of patches in an image. More recently, [20] proposed rotating an image and tasking a network with predicting in what orientation was the image rotated (RotNet). In self-supervised GANs [9], predicting image rotations is used as an auxiliary task to stabilize and improve training, by enhancing the discriminator's representation capabilities. In our method we adopt this approach as well, as elaborated later on.

2.2.3 Deep Clustering

A straightforward approach for image clustering is to first perform unsupervised representation learning with some existing method and then apply clustering on the representations. Deep Embedded Clustering (DEC) [50] first trains an autoencoder and then discards of the decoder. In the second stage, cluster centroids in the encoders embedding space are iteratively calculated and refined until convergence.

The dominant and most successful approach to clustering of images in recent years has been to incorporate the tasks of representation learning and clustering into a single framework. In Joint Unsupervised Learning (JULE) [52], the authors adopt an agglomerative clustering approach by iteratively merging clusters of deep representations and updating the networks parameters to better predict the cluster assignments of the images. Associative Deep Clustering (ADC) [23] jointly learns network parameters and embedding centroids with an association loss in order to estimate cluster membership. Deep Adaptive Clustering (DAC) [8] and DCCM [49] recast the clustering problem into a binary pairwise-classification framework, where cosine distances between image features

of image pairs are used as a similarity measure to decide if they belong to the same cluster.

More recently, Invariant Information Clustering (IIC) [27] adopts an approach that achieves clustering based on maximizing the mutual information between two sets: deep embeddings of images, and instances of the images that underwent random image transformations while keeping the image semantic meaning intact. The natural properties of the mutual information formulation assures that not all images are assigned to the same cluster, as when maximizing the mutual information, the entropy of the cluster assignments is maximized as well. IIC leverages auxiliary over-clustering to increase expressivity in the learned feature representation, improving the representation capabilities of its network. This tactic bears resemblance to our incorporation of rotation prediction as an auxiliary task.

CHAPTER 3

OUR METHOD

In this chapter we present Multi-Modal Deep Clustering (MMDC), a novel image clustering algorithm. Our goal is to partition a set of images into k clusters, which reflect internal structure in the data. Fig. 3.2 shows an overview of the proposed approach. The algorithm alternates between solving the main unsupervised clustering task, and an auxiliary self-supervised task that helps the training process. The ingredients of the method are described next. The full method is summarized in Algorithm 1.

3.1 Unsupervised Learning

The starting point for this work is an unsupervised learning framework for learning image representation from unlabeled data. The method, Noise as Targets [6], learns useful representations of images by training a deep network to align its images' embeddings with a fixed set of target points. The target points are uniformly scattered on the d-dimensional unit sphere.

More specifically, let $X = \{x_i\}_{i=1}^n$ denote a set of images, and $f_\theta : X \to Z$ the parameterized deep network we wish to train. The output of f_θ is normalized to have ℓ_2 norm of 1, entailing that Z is the d-dimensional unit sphere. NAT starts by uniformly sampling n targets on this unit sphere. Let $\{t_i\}_{i=1}^n$ denote the set of target points, which remain fixed throughout the training. Each image x_i is assigned a unique target y_i through a

permutation $P : [n] \rightarrow [n]$. The optimization objective is formulated as

$$\min_{\theta,P} \frac{1}{n} \sum_i \ell(f_\theta(x_i), y_i) \qquad y_i = t_{P(i)} \tag{3.1}$$

where ℓ is the Euclidean distance.

This optimization problem is solved in a stochastic manner, by iteratively solving it over randomly sampled mini-batches. Given a mini-batch of images X_b, the current representation vectors $f_\theta(X_b)$ are first computed. Subsequently, Equation (3.1) is optimized for P over the points in mini-batch X_b using the Hungarian method [32], which reassigns the currently assigned targets of the mini-batch to minimize the Euclidean distance (ℓ_2) between images and their assigned target points. Finally, the gradients of f_θ on X_b with respect to θ are computed, and an SGD step is executed.

Intuitively, NAT permutes the assignment of image representation vectors to target points delivered by f_θ, so that nearby embedding vectors are mapped to nearby target vectors, and then updates θ accordingly. This process leads to the grouping of semantically similar images in target space, and to effective representations that perform well in downstream computer vision classification and detection tasks.

3.2 Multi-Modal Distribution of Target Points

The uniform distribution of target points on the unit sphere, as described above, is not well suited for unsupervised clustering, since it is likely to blur the dividing lines between clusters rather than sharpen them. Instead, multi-modal distribution seems like a natural choice for the objective of clustering, as it directly produces separated groups in target space.

In this work, we propose to use the mixture of Gaussians distribution, projected to the unit sphere, for the sampling of target points. Formally, this implies:

$$\begin{aligned} p(u) &= \sum_{k=1}^{K} \alpha_k \cdot p_k(u) && u \in \mathbb{R}^d \\ p(t_i) &= \int_{\frac{u}{\|u\|_2}=t_i} p(u)du && t_i \in Z \end{aligned} \tag{3.2}$$

where K denotes the number of Gaussians in the mixture, d the dimension of the embedding space, $\alpha_{k=1..K}$ a categorical random variable, and $p_k(u)$ the multivariate normal distribution $N(\mu_k, \Sigma_k)$, parameterized by mean vector μ_k and covariance matrix Σ_k. In the absence of prior knowledge we assume that the mixture components are

Algorithm 1

Input:
$\{x_i\}_{i=1}^n$ - images
f_θ - ConvNet with two heads
k - number of clusters
$epochs$ - number of epochs to train
$iters$ - number of iterations in an epoch
σ - variance of normal distribution
d - dimension of embedding space
λ_c, λ_r - learning rates
g - random image transformation
r - number of instances of g in a batch
Init:
$P \leftarrow$ initialize with random assignments
$\theta \leftarrow$ initialize with random weights
$T \leftarrow$ initialize empty list
for $i = 1...n$ **do**
  sample $c \sim Categ(\frac{1}{K}, ..., \frac{1}{K})$
  sample $u \sim N(\mu_c, \sigma \cdot I_{d \times d})$
  $T[i] \leftarrow t_i = \frac{u}{\|u\|}$
end for
for $e = 1...epochs$ **do**
  for $i = 1...iters$ **do**
    sample batch X_b and assigned targets T_b
    compute $f_\theta(X_b)$
    update P by minimizing Equation (3.1) w.r.t P
    compute $\nabla_\theta L_c(\theta)$ of Equation (3.1) for $g(X_b)$
    update $\theta \leftarrow \theta - \lambda_c \nabla_\theta L_c(\theta)$
  end for
  for $i = 1...iters$ **do**
    sample batch X_b
    rotate X_b $\forall r \in \{0^\circ, 90^\circ, 180^\circ, 270^\circ\}$
    compute $\nabla_\theta L_r(\theta)$ // L_r is cross-entropy loss
    update $\theta \leftarrow \theta - \lambda_r \nabla_\theta L_r(\theta)$
  end for
end for

equally likely, namely $\alpha_k = \frac{1}{K} \forall k \in [K]$. Finally, since the target points are constrained to lie on the unit sphere, we project the sample in $\mathbb{R}^d$ to the unit sphere by $t_i = \frac{u}{\|u\|_2}$.

We define the cluster assignment c_i of image x_i as follows

$$c_i = \arg\min_k \|f_\theta(x_i) - \mu_k\|_2 \tag{3.3}$$

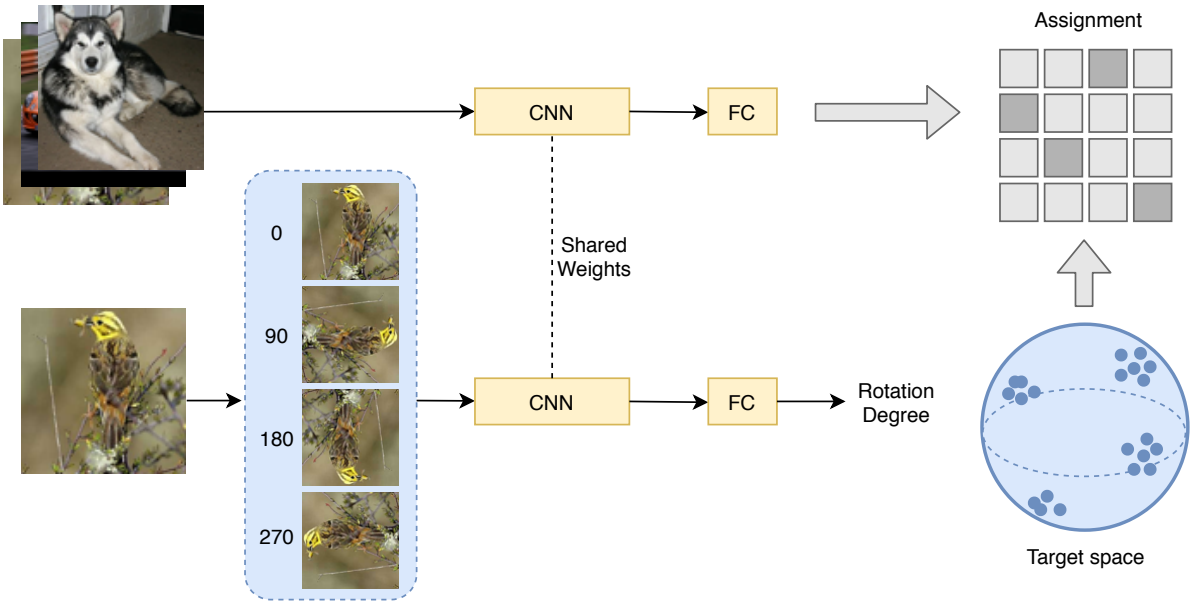

Figure 3.1: Our approach takes a set of images and solves two tasks in alternating epochs. In the primary task, a CNN is trained to produce output which matches some predefined set of target points sampled from a Gaussian mixture model, and optimally aligned with the training set. In the secondary task, given a rotated image, the same CNN is trained to predict the rotation angle of the image.

Note that if the final network f_θ fits that target points exactly, namely $f_\theta(x_i) = y_i$, and if Σ_k are the same $\forall k$, then with high probability c_i is the index of the mixture component from which target point y_i has been sampled.

3.3 Image Transformations

Data augmentation is a useful and common technique to improve performance of machine learning algorithms. Usually, random image transformations such as cropping, flipping, rotation, scaling and photometric transformations are applied to images in order to expand the dataset with new and unique images. In our task of unsupervised clustering, these random transformations are essential, because they provide several instances of the same image that appear different but share the same semantic meaning as they contain the same object. Let g denote a random image transformation. In our method, we use the

center crop of an image when minimizing Equation (3.1) w.r.t P. When minimizing the same equation w.r.t θ, we first apply g to the image. Why is this algorithmic ingredient useful? When training the ConvNet, it must find common patterns between the original images and transformed images when fitting them to the same target. These common patterns are likely to appear in other images in the dataset belonging to the same class. This pushes the network to map images that contain the same objects closer to each other, in a similar manner to the beneficial effect of self-supervision.

3.4 Auxiliary Task

While optimizing the clustering objective (3.1), the ConvNet model simultaneously learns image representation and partitions the images. The success of unsupervised clustering is highly correlated with the quality of the learnt representation. It has been repeatedly shown that self-supervision methods can significantly improve the quality of representations in an unsupervised learning scenario. To benefit from this idea, we employ RotNet [20], which is a self-supervised learning algorithm that learns image features by training a ConvNet to predict image rotations. Specifically, images are rotated by r degrees where $r \in \{0^\circ, 90^\circ, 180^\circ, 270^\circ\}$, and the model is subsequently trained to predict their rotation by optimizing the cross-entropy loss. RotNet produces competitive performance in representation learning benchmarks, and has been shown to benefit training in other tasks, when incorporated into a model as an auxiliary task [9, 19, 35]. We incorporate RotNet into our method, modifying the ConvNet training procedure to alternate between optimizing the main clustering task and this secondary auxiliary task.

3.5 Refinement Stage

As we have no prior knowledge regarding the size of the clusters, we begin by assuming that clusters' sizes are equal. When this assumption cannot be justified, we propose to augment the algorithm with an additional step, performed after the main training is concluded. In this step the assumption is relaxed, while target points are iteratively reassigned based on the outcome of k-means applied to $f_\theta(x_1), ..., f_\theta(x_n)$, and assigning image x_i to target μ_j with label $j \in [K]$ derived from the outcome of k-means. This ingredient is similar to DeepCluster [7], proposed by Caron et al. as an approach for representation learning, where they perform the clustering on the latent vectors of the

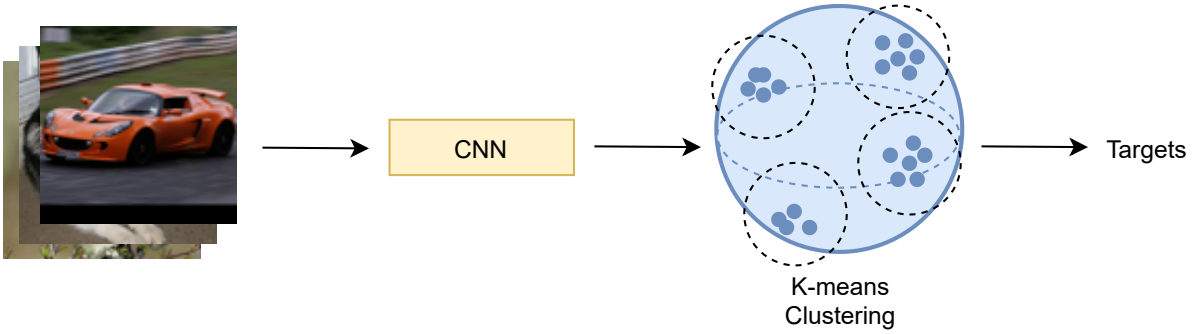

Figure 3.2: Illustration of the final refinement stage of our method. Images $x_1, ..., x_n$ are reassigned targets at the start of each epoch by applying K-means clustering on $f_\theta(x_1), ..., f_\theta(x_n)$ and assigning target μ_j to image x_i with cluster assignment $j \in [K]$.

model and not the final output layer. A possible alternative method may start with this stage and discard the first one altogether, as this approach makes no assumption on the size of the clusters. However, we found that starting off with reassigning labels based on k-means is not competitive and produces less accurate clusters. For example, training on MNIST results in low accuracy of 81%.

CHAPTER 4

EXPERIMENTAL EVALUATION

We empirically evaluate our proposed algorithm for image clustering on a variety of datasets and compare the results to previous methods discussed in chapter 2. We specify the implementation details of our method, the evaluation protocol and analyze the results. Subsequently, we report the results of an ablation study evaluating the various ingredients of the algorithm, which demonstrate how they contribute to its success

4.1 Datasets

The datasets we evaluate our method on are commonly used as benchmarks for image clustering methods and contain varying amounts of classes and samples, and a variety of resolutions. We are most interested in the datasets that consist of natural images, which all but one do. The six datasets used in our empirical study: MNIST [33], CIFAR-10 [31], the 20 superclasses of CIFAR-100 [31], STL-10 [10], ImageNet-10 (a subset of ImageNet [12]) and Tiny-ImageNet [12], more details are presented in Table 4.1.

Table 4.1: The image datasets used in our experiments.

Name	Classes	Samples	Dimension
MNIST	10	70,000	28×28
CIFAR-10	10	60,000	32×32×3
CIFAR-100	20	60,000	32×32×3
STL-10	10	13,000	96×96×3
ImageNet-10	10	13,000	96×96×3
Tiny-ImageNet	200	100,000	64×64×3

4.2 Implementation Details

4.2.1 Architectures

For the MNIST experiments we use a small VGG model [46] with batch normalization [25]. Each block in this neural network consists of one convolution layer, followed by a batch normalization layer and ReLU activation function, and ends with a max pooling layer. Our model has four blocks. For all other experiments we use a ResNet model [24] with 18 layers. These base models are followed by a linear prediction layer, that outputs the cluster assignments. When trained on the auxiliary task, the base model is also followed by another linear head, which predicts the image rotation.

4.2.2 Training Details

The network is trained with stochastic gradient descent with learning rate 0.05 and momentum of 0.9. We apply weight decay of 0.0001 for CIFAR-100 and Tiny-ImageNet, and 0.0005 for all other datasets. We use batch size 128 and perform random image augmentations which include cropping, flipping and color jitter. When training on the auxiliary rotation task, we rotate each image to all four orientations, resulting in an effective batch size of 512. We train the network for 400 epochs and decay learning rate by a factor of 5 after 350 epochs. For MNIST we train for 50 epochs and decay learning rate by a factor of 10 after 40 epochs. Training on CIFAR-10 takes 10.5 hours on a single GTX-1080 GPU.

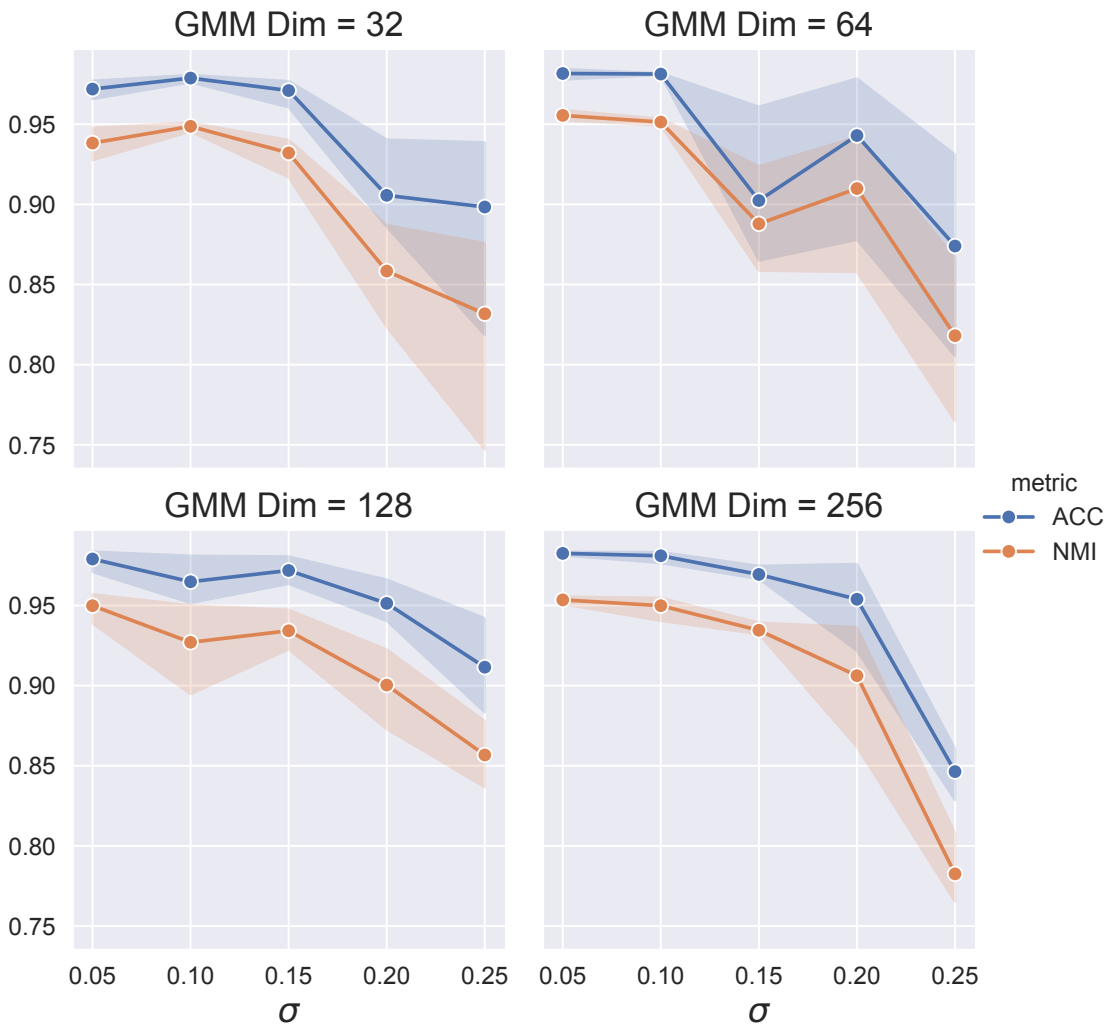

Figure 4.1: Comparison of clustering performance on MNIST with different Mixture of Gaussians initializations. We compare different dimensions for the target vectors and different coefficient parameters (σ) for the covariance matrices of the gaussians. These results do not include performing the refinement stage.

4.2.3 Mixture of Gaussians

We examined several initialization heuristics to determine the Gaussian means $\{\mu_k\}$ in the GMM distribution defined in (3.2) and the covariance matrices $\{\Sigma_k\}$. A comparison of different initialization schemes is provided in Figure 4.1, where all vectors lie on the d-dimensional unit sphere. Gaussian means $\{\mu_k\}$ are sampled from a multi-variate uniform distribution within the range $[-0.1, 0.1]$ and projected onto the unit sphere. We always set $\Sigma_k = \sigma \cdot I_{K\times K}\ \forall k \in [K]$. We compare different values for the dimension d and

the variance parameter σ. Smaller variance usually performs best with the added benefit of similar performance for different choices of dimension d. We therefore opted to use K different one-hot vectors in $\mathbb{R}^K$ for $\{\mu_k\}$ with variance $\sigma = 0$, as this achieved good performance while reducing the number of free hyperparameters.

4.3 Evaluation Metrics

To evaluate clustering performance we adopt two commonly used scores: Normalized Mutual Information (NMI), and Clustering Accuracy (ACC). Clustering accuracy measures the accuracy of the hard-assignment to clusters, with respect to the best permutation of the dataset's ground-truth labels. The best permutation can be found using the Hungarian algorithm [32]. Normalized Mutual Information measures the mutual information between the ground-truth labels and the predicted labels based on the clustering method. The range of both scores is [0, 1], where a larger value indicates more precise clustering results. We formally define these metrics as:

$$ACC(c, y, X) = \max_{\pi \in S_K} \frac{1}{|X|} \sum_{x \in X} 1_{y(x)=\pi(c(x))} \tag{4.1}$$

$$NMI(c, y, X) = \frac{1}{|X|} \sum_{x \in X} \frac{I(y(x), c(x))}{\sqrt{H(y(x))H(c(x))}} \tag{4.2}$$

Where X denotes the dataset on which clustering is performed, $c(x) \in [K]$ denotes the cluster assignment of sample x, $y(x) \in [K]$ denotes the ground truth label of sample x, S_K denotes the set of all possible permutations on K elements, I denotes mutual information and H denotes the entropy function.

4.4 Results

The results of our method when applied to the six image datasets are reported in Table 4.4. Clearly, our clustering algorithm is able to separate unlabeled images into distinct groups of semantically similar images with high accuracy, improving the state-of-the-art in the five datasets of natural images. Examples of clustering results on the STL-10 dataset of natural images are shown in Figure 4.2.

As baselines, we report clustering results of classical algorithms such as k-means (KM) [3] and spectral clustering (SC) [53], and a simple deep learning approach of

Table 4.2: Unsupervised clustering results. The results of our method are shown below the separation line. For each dataset, we show the average result over five runs, standard error (ste) and the best run. Above the separation line we list state of the art results for comparison, see review in Chapter 2. Unreported results are marked with (-).

	MNIST		CIFAR-10		CIFAR-100		STL-10		ImNet-10		Tiny-ImNet	
	NMI	ACC	NMI	ACC	NMI	ACC	NMI	ACC	NMI	ACC	NMI	ACC
KM	0.499	0.572	0.087	0.228	0.083	0.129	0.124	0.192	0.119	0.241	0.065	0.025
SC	0.663	0.696	0.103	0.247	0.090	0.136	0.098	0.159	0.151	0.274	0.063	0.022
AE	0.725	0.812	0.239	0.313	0.100	0.164	0.249	0.303	0.210	0.317	0.131	0.041
DEC	0.772	0.843	0.257	0.301	0.136	0.185	0.276	0.359	0.282	0.381	0.115	0.037
JULE	0.913	0.964	0.192	0.272	0.103	0.137	0.182	0.277	0.175	0.300	0.102	0.033
DAC	0.935	0.978	0.396	0.522	0.185	0.238	0.249	0.303	0.394	0.527	0.190	0.066
IIC	**0.978**	**0.992**	0.513	0.617	0.224	0.257	0.431	0.499	-	-	-	-
DCCM	-	-	0.496	0.623	0.285	0.327	0.376	0.482	0.608	0.710	0.224	0.108
Ours avg.	0.971	0.990	**0.703**	**0.820**	**0.418**	**0.446**	**0.593**	**0.694**	**0.719**	**0.811**	**0.274**	**0.119**
Ours ste	±.000	±.000	±.011	±.019	±.003	±.006	±.005	±.013	±.008	±.012	±.001	±.001
Ours best	0.973	0.991	0.720	0.843	0.423	0.464	0.609	0.741	0.732	0.830	0.277	0.121

applying k-means to the latent space of an autoencoder (AE). We also compare ourselves to state-of-the-art methods such as DEC [51], JULE [52], DAC [8], IIC [27] and DCCM [49]. Compared to previous state-of-the-art, we improve clustering accuracy on CIFAR-10 by 20%, CIFAR-100 by 12%, STL-10 by 20%, ImageNet-10 by 10% and Tiny-ImageNet by 1%.

In the results reported in Table 4.4, the refinement stage was invoked only when using the MNIST dataset. A more complete ablation study of the refinement stage is reported in Table 4.5. The auxiliary task of RotNet, which was shown to be beneficial when learning natural images, was used to enhance the clustering of all the datasets except MNIST. For reference, we used the same image augmentations as in [27], which uses a larger ResNet-34 as the backbone for the model. We use centrally cropped images for evaluation.

4.4.1 Benefits of Auxiliary Task

Applying the Sobel filter to an image emphasizes edges and discards colors. This pre-processing is commonly done in the context of unsupervised representation learning and clustering algorithms, presumably to avoid sub-optimal solutions based on trivial cues such as color [6, 27]. We observed an interesting interaction between Sobel filtering and

Table 4.3: Clustering performance on CIFAR-10, showing the combined effect of pre-processing with the Sobel filter and adding a rotation loss. First row: no pre-processing and no rotation loss, second row: pre-processing and no rotation loss, third row: no pre-processing with a rotation loss, fourth row: both.

Sobel	Rotation loss	NMI	ACC
		0.428 ± .005	0.492 ± .003
✓		0.463 ± .003	0.560 ± .006
	✓	0.703 ± .011	0.820 ± .019
✓	✓	0.610 ± .010	0.725 ± .020

training with the auxiliary task of predicting image rotations. Without the auxiliary task, Sobel filtering indeed improves clustering performance as seen in Table 4.3. In contrast, when training with an auxiliary task and adding the rotation loss, pre-processing with the Sobel filter degrades the algorithms performance. Furthermore, without the rotation loss the learning rate has to be reduced to 0.01 for training to converge. The reason may be that trivial cues such as color are not beneficial for the task of predicting image rotations, and therefore the auxiliary task forces the ConvNet to learn features that focus on the object in the image. Once the focus is on the object, additional cues such as color can be beneficial for clustering, and as a result pre-processing with the Sobel filter is detrimental to the algorithm's performance.

4.4.2 Features Evaluation

Our algorithm borrows some of its ingredients from NAT and RotNet. However, while these two methods address representation learning, the final goal of our method is clustering. Nevertheless, we compare our method to NAT and RotNet in two ways. First, we examine the clustering capabilities of the methods by applying k-means to the penultimate layer of the networks. Second, we evaluate the learnt features by training a linear classifier with the image labels on top of the frozen features of the networks. We use the same architecture and image transformations as our model for both methods. We follow the training procedure from [30] for training RotNet and [6] for training NAT.

More specifically, we train the linear classifier with stochastic gradient descent with learning rate 0.1, momentum of 0.9, weight decay of 0.00001, batch size of 128, cosine annealing for learning rate scheduling, and 100 training epochs. Results with CIFAR-10 and CIFAR-100 are reported in Table 4.4. As shown our method outperforms the others

in all cases except one, where NAT+RotNet performs better when clustering CIFAR-10 image features. As a reference for the linear classifier performance, we also evaluate a model pretrained with ImageNet (first row in Table 4.4). Note that we use the same image augmentations as for training the unsupervised methods, including 20×20 cropping, which may degrade performance for this model.

Table 4.4: Evaluation of unsupervised feature learning methods on CIFAR-10 and CIFAR-100. We use the penultimate layer of the network as image features and test performance with two procedures. We perform k-means clustering on the image features and train a linear classifier using the image labels. As a reference, we report results using an imagenet-pretrained ResNet-18.

	CIFAR-10			CIFAR-100		
	K-means		Linear	K-means		Linear
	NMI	ACC	ACC	NMI	ACC	ACC
ImNet	0.321	0.407	0.782	0.247	0.281	0.646
NAT	0.044 ± .001	0.162 ± .001	0.315 ± .002	0.037 ± .001	0.095 ± .001	0.177 ± .001
RotNet	0.329 ± .011	0.349 ± .012	0.740 ± .002	0.261 ± .006	0.284 ± .013	0.543 ± .001
NAT+Rot	0.413 ± .005	**0.511 ± .002**	0.764 ± .001	0.190 ± .007	0.232 ± .006	0.499 ± .002
Ours	**0.428 ± .011**	0.397 ± .018	**0.869 ± .002**	**0.395 ± .002**	**0.347 ± .007**	**0.662 ± .001**

4.4.3 Refinement Stage

We compare clustering performance with and without the proposed refinement stage in Table 4.5. MNIST is the only dataset with class imbalance, as its smallest class has 6313 samples while its largest has 7877. Reassuringly, the refinement stage helps the algorithm achieve near perfect clustering with accuracy of 99.0%.

Table 4.5: Comparison of clustering performance before and after the refinement stage.

		MNIST		CIFAR-10		CIFAR-100		STL-10		ImNet-10		Tiny-ImNet	
		NMI	ACC	NMI	ACC	NMI	ACC	NMI	ACC	NMI	ACC	NMI	ACC
Before	avg.	0.950	0.981	0.703	0.820	0.418	0.446	0.593	0.694	0.719	0.811	0.274	0.119
	ste	±.002	±.001	±.011	±.019	±.003	±.006	±.005	±.013	±.008	±.012	±.001	±.001
After	avg.	0.971	0.990	0.715	0.829	0.422	0.446	0.596	0.696	0.725	0.815	0.254	0.095
	ste	±.000	±.000	±.009	±.021	±.002	±.005	±.005	±.013	±.008	±.012	±.001	±.002

Figure 4.2: Visualization of image clustering results on STL-10. Each column shows images from a different cluster. The top seven images in each column are examples of images from the same class successfully clustered together. The images in the bottom three rows illustrate failure cases, where the image is assigned to the wrong cluster (e.g., an airplane assigned to the 'bird' cluster).

CHAPTER 5

SUMMARY AND CONCLUSIONS

For the task of unsupervised semantic image clustering, we presented an end-to-end deep clustering framework, that trains a ConvNet to align image embeddings with targets sampled from a Gaussian Mixture Model by solving a linear assignment problem using the Hungarian algorithm. Image transformations that preserve the semantic context of an image, such as cropping and flipping, are used to create several instances of an image that are mapped to the same target. This pushes the network to map images that contain the same objects closer to each other. The usage of fixed targets prevent the determinal phenomenon of target collapse, where all images would be assigned the same target, and consequently the same cluster. Motivated by the idea that a pleasing clustering of images requires an effective image representation to support this effort, we incorporated an additional auxiliary task to the training of the ConvNet - the prediction of image rotation, an effective self-supervised representation learning approach. Our ablation study shows that the contribution of this component is essential for the success of the method.

Even though the proposed method is quite simple, it yields a significant improvement on previous state-of-the-art methods on a variety of challenging benchmarks, further closing the gap between unsupervised and supervised learning. Furthermore, it is quite efficient and takes less time to train than previous state-of-the-art methods.

BIBLIOGRAPHY

[1] E. ACHTERT, C. BÖHM, AND P. KRÖGER, *Deli-clu: Boosting robustness, completeness, usability, and efficiency of hierarchical clustering by a closest pair ranking*, in Advances in Knowledge Discovery and Data Mining, W.-K. Ng, M. Kitsuregawa, J. Li, and K. Chang, eds., Berlin, Heidelberg, 2006, Springer Berlin Heidelberg, pp. 119–128.

[2] M. ANKERST, M. M. BREUNIG, H.-P. KRIEGEL, AND J. SANDER, *Optics: Ordering points to identify the clustering structure*, SIGMOD Rec., 28 (1999), p. 49,Äì60.

[3] D. ARTHUR AND S. VASSILVITSKII, *K-means++: The advantages of careful seeding*, in Proc. of the Annu. ACM-SIAM Symp. on Discrete Algorithms, vol. 8, 2007, pp. 1027–1035.

[4] Y. BENGIO, A. COURVILLE, AND P. VINCENT, *Representation learning: A review and new perspectives*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 35 (2013), pp. 1798–1828.

[5] Y. BENGIO, P. LAMBLIN, D. POPOVICI, AND H. LAROCHELLE, *Greedy layer-wise training of deep networks*, in Advances in Neural Information Processing Systems (NeurIPS), vol. 19, 2007.

[6] P. BOJANOWSKI AND A. JOULIN, *Unsupervised learning by predicting noise*, in International Conference on Machine Learning (ICML), 2017.

[7] M. CARON, P. BOJANOWSKI, A. JOULIN, AND M. DOUZE, *Deep clustering for unsupervised learning of visual features*, in European Conference on Computer Vision (ECCV), 2018.

[8] J. CHANG, L. WANG, G. MENG, S. XIANG, AND C. PAN, *Deep adaptive image clustering*, in IEEE/CVF International Conference on Computer Vision (ICCV), 2017, pp. 5880–5888.

[9] T. CHEN, X. ZHAI, M. RITTER, M. LUfÇIfÁ, AND N. HOULSBY, *Self-supervised gans via auxiliary rotation loss*, in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 12146–12155.

[10] A. COATES, A. NG, AND H. LEE, *An analysis of single-layer networks in unsupervised feature learning*, Journal of Machine Learning Research (JMLR) - Proceedings Track, 15 (2011), pp. 215–223.

[11] 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), 39 (1977), pp. 1–22.

[12] J. DENG, W. DONG, R. SOCHER, L. LI, KAI LI, AND LI FEI-FEI, *Imagenet: A large-scale hierarchical image database*, in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2009, pp. 248–255.

[13] C. DOERSCH, A. GUPTA, AND A. A. EFROS, *Unsupervised visual representation learning by context prediction*, in IEEE/CVF International Conference on Computer Vision (ICCV), 2015, pp. 1422–1430.

[14] C. DOERSCH AND A. ZISSERMAN, *Multi-task self-supervised visual learning*, in IEEE/CVF International Conference on Computer Vision (ICCV), 2017, pp. 2051–2060.

[15] J. DONAHUE, P. KRAHENBUHL, AND T. DARRELL, *Adversarial feature learning*, (2017).

[16] J. DONAHUE AND K. SIMONYAN, *Large scale adversarial representation learning*, in Advances in Neural Information Processing Systems (NeurIPS), 2019.

[17] M. ESTER, H.-P. KRIEGEL, J. SANDER, AND X. XU, *A density-based algorithm for discovering clusters in large spatial databases with noise*, in Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, KDD'96, AAAI Press, 1996, p. 226,Äì231.

[18] C. GERLHOF, A. KEMPER, C. KILGER, AND G. MOERKOTTE, *Partition-based clustering in object bases: From theory to practice*, in International Conference on Foundations of Data Organization and Algorithms, Springer, 1993, pp. 301–316.

[19] S. GIDARIS, A. BURSUC, N. KOMODAKIS, P. PÉREZ, AND M. CORD, *Boosting few-shot visual learning with self-supervision*, in IEEE/CVF International Conference on Computer Vision (ICCV), 2019.

[20] S. GIDARIS, P. SINGH, AND N. KOMODAKIS, *Unsupervised representation learning by predicting image rotations*, in International Conference on Learning Representations (ICLR), 2018.

[21] I. J. GOODFELLOW, J. POUGET-ABADIE, M. MIRZA, B. XU, D. WARDE-FARLEY, S. OZAIR, A. COURVILLE, AND Y. BENGIO, *Generative adversarial nets*, in Advances in Neural Information Processing Systems (NeurIPS), vol. 2, 2014, pp. 2672–2680.

[22] K. C. GOWDA AND G. KRISHNA, *Agglomerative clustering using the concept of mutual nearest neighbourhood*, Pattern Recognition, 10 (1978), pp. 105–112.

[23] P. HÄUSSER, J. PLAPP, V. GOLKOV, E. ALJALBOUT, AND D. CREMERS, *Associative deep clustering: Training a classification network with no labels*, in German Conference on Pattern Recognition (GCPR), 2017.

[24] K. HE, X. ZHANG, S. REN, AND J. SUN, *Deep residual learning for image recognition*, in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770–778.

[25] S. IOFFE AND C. SZEGEDY, *Batch normalization: Accelerating deep network training by reducing internal covariate shift*, in International Conference on Machine Learning (ICML), 2015.

[26] A. K. JAIN, M. N. MURTY, AND P. J. FLYNN, *Data clustering: a review*, ACM computing surveys (CSUR), 31 (1999), pp. 264–323.

[27] X. JI, J. F. HENRIQUES, AND A. VEDALDI, *Invariant information clustering for unsupervised image classification and segmentation*, in IEEE/CVF International Conference on Computer Vision (ICCV), 2019, pp. 9865–9874.

[28] D. J. KETCHEN AND C. L. SHOOK, *The application of cluster analysis in strategic management research: an analysis and critique*, Strategic Management Journal, 17 (1996), pp. 441–458.

[29] D. KINGMA AND M. WELLING, *Auto-encoding variational bayes*, in International Conference on Learning Representations (ICLR), 2014.

[30] A. KOLESNIKOV, X. ZHAI, AND L. BEYER, *Revisiting self-supervised visual representation learning*, in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019.

[31] A. KRIZHEVSKY, *Learning multiple layers of features from tiny images*, University of Toronto, (2009).

[32] H. W. KUHN AND B. YAW, *The hungarian method for the assignment problem*, Naval Res. Logist. Quart, (1955), pp. 83–97.

[33] Y. LECUN, L. BOTTOU, Y. BENGIO, AND P. HAFFNER, *Gradient-based learning applied to document recognition*, in Proceedings of the IEEE, 1998, pp. 2278–2324.

[34] S. P. LLOYD, *Least squares quantization in pcm*, IEEE Trans. Inf. Theory, 28 (1982), pp. 129–136.

[35] M. LUfÇIfÁ, M. RITTER, M. TSCHANNEN, X. ZHAI, O. F. BACHEM, AND S. GELLY, *High-fidelity image generation with fewer labels*, in International Conference on Machine Learning (ICML), 2019.

[36] J. MASCI, U. MEIER, D. C. CIRESAN, AND J. SCHMIDHUBER, *Stacked convolutional auto-encoders for hierarchical feature extraction*, in International Conference on Artificial Neural Networks (ICANN), 2011.

[37] E. MIN, X. GUO, Q. LIU, G. ZHANG, J. CUI, AND J. LONG, *A survey of clustering with deep learning: From the perspective of network architecture*, IEEE Access, 6 (2018), pp. 39501–39514.

[38] 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, NIPS'01, Cambridge, MA, USA, 2001, MIT Press, p. 849,Äì856.

[39] M. NOROOZI AND P. FAVARO, *Unsupervised learning of visual representations by solving jigsaw puzzles*, in European Conference on Computer Vision (ECCV), 2016.

[40] R. OSTROVSKY, Y. RABANI, L. J. SCHULMAN, AND C. SWAMY, *The effectiveness of lloyd-type methods for the k-means problem*, in IEEE Symposium on Foundations of Computer Science (FOCS), 2006, pp. 165–176.

[41] D. PATHAK, P. KRAHENBUHL, J. DONAHUE, T. DARRELL, AND A. EFROS, *Context encoders: Feature learning by inpainting*, in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

[42] A. RADFORD, L. METZ, AND S. CHINTALA, *Unsupervised representation learning with deep convolutional generative adversarial networks*, (2015).

[43] P. J. ROUSSEEUW, *Silhouettes: A graphical aid to the interpretation and validation of cluster analysis*, Journal of Computational and Applied Mathematics, 20 (1987), pp. 53–65.

[44] M. ROUX, *A comparative study of divisive and agglomerative hierarchical clustering algorithms*, Journal of Classification, (2018).

[45] J. SHI AND J. MALIK, *Normalized cuts and image segmentation*, IEEE Trans. Pattern Anal. Mach. Intell., 22 (2000), p. 888,Äì905.

[46] K. SIMONYAN AND A. ZISSERMAN, *Very deep convolutional networks for large-scale image recognition*, in International Conference on Learning Representations (ICLR), 2015.

[47] P. VINCENT, H. LAROCHELLE, I. LAJOIE, Y. BENGIO, AND P.-A. MANZAGOL, *Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion*, Journal of Machine Learning Research (JMLR), 11 (2010), pp. 3371–3408.

[48] W. WANG, Y. WU, C. TANG, AND M. HOR, *Adaptive density-based spatial clustering of applications with noise (dbscan) according to data*, in International Conference on Machine Learning and Cybernetics (ICMLC), vol. 1, 2015, pp. 445–451.

[49] J. WU, K. LONG, F. WANG, C. QIAN, C. LI, Z. LIN, AND H. ZHA, *Deep comprehensive correlation mining for image clustering*, in IEEE/CVF International Conference on Computer Vision (ICCV), 2019, pp. 8150–8159.

[50] J. XIE, R. GIRSHICK, AND A. FARHADI, *Unsupervised deep embedding for clustering analysis*, in International Conference on Machine Learning (ICML), 2016, p. 478,Äì487.

[51] J. XIE, R. B. GIRSHICK, AND A. FARHADI, *Unsupervised deep embedding for clustering analysis*, in International Conference on Machine Learning (ICML), 2015.

[52] J. YANG, D. PARIKH, AND D. BATRA, *Joint unsupervised learning of deep representations and image clusters*, in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

[53] L. ZELNIK-MANOR AND P. PERONA, *Self-tuning spectral clustering*, in Proceedings of the 17th International Conference on Neural Information Processing Systems, NIPS'04, Cambridge, MA, USA, 2004, MIT Press, pp. 1601–1608.

[54] J. ZHAO, M. MATHIEU, R. GOROSHIN, AND Y. LECUN, *Stacked what-where auto-encoders*, 2015.