Università di Genova | MaLGa

Unsupervised learning

30/04/2025

Vito Paolo Pastore

Deep learning a.y. 2024/2025

Credits

These slides have been built upon the following tutorials or lecture:

- https://ranzato.github.io/publications/tutorial_deep_unsup_learning_part1_NeurIPS2018.pdf

Some slides from:

- Vittorio Murino
- Pietro Morerio

UniGe | MaLGa

Unsupervised learning

Types of Learning

	With Teacher	Without Teacher
Active	Reinforcement Learning / Active Learning	Intrinsic Motivation / Exploration
Passive	Supervised Learning	Unsupervised Learning

Why Learning without a teacher?

If the goal is to create intelligent systems that can succeed at a wide variety of tasks (RL or supervised), why not just teach them those tasks directly?

1. Targets/rewards can be difficult to obtain or define
2. Unsupervised learning feels more human
3. **Want rapid generalization to new tasks and situations**

What about Transfer learning?

- Teaching on one task and transferring to another (multi-task learning, one-shot learning...) kind of works
- E.g. Retraining speech recognition systems from a language with lots of data can improve performance on a related language with little data
- But never seems to transfer as far or as fast as we want it to
- Maybe there just isn't enough information in the targets/rewards to learn transferable skills?

The cherry on the cake

- The targets for supervised learning contain far less information than the input data;
- RL reward signals contain even less;
- Unsupervised learning gives us an essentially unlimited supply of information about the world: surely we should exploit that?

If intelligence was a cake, unsupervised learning would be the cake, supervised learning would be the icing on the cake, and reinforcement learning would be the cherry on the cake. – Yann LeCun

An example

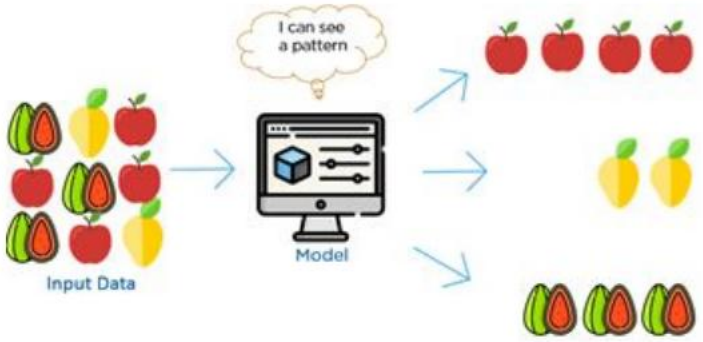
- ImageNet training set contains ~1.28M images, each assigned one of 1000 labels
- If labels are equally probable, complete set of randomly shuffled labels contains ~log2 (1000)*1.28M ≈ 12.8 Mbits
- Complete set of images uncompressed at 128 x128 contains ~500 Gbits: > 4 orders of magnitude more
- A large conv net (~30M weights) can memorise randomised ImageNet labellings. Could it memorise randomised pixels?

understanding deep learning requires rethinking generalization, zhang et. al. 2016

Unsupervised learning

- Given a dataset D of inputs x, learn to predict… what?

$$\mathcal{D} = \{x\}$$
$$L(\mathcal{D}) = ???$$



- Basic challenge of unsupervised learning is that the task is undefined
- Want a single task that will allow the network generalise to many other tasks (which ones?)

A possibility? Density modeling

- Simplest approach: do maximum likelihood on the data instead of the targets

$$\mathcal{D} = \{x\}$$
$$L(\mathcal{D}) = \sum_{x \in \mathcal{D}} -\log p(x)$$

- Goal is to learn the 'true' distribution from which the data was drawn
- Means attempting to learn everything about the data

Generative models are unsupervised

- Modelling densities also gives us a generative model of the data (as long as we can draw samples)
- Allows us to ‘see’ what the model has and hasn’t learned

"Diffusion Models Beat GANs on Image Synthesis"
Dhariwal & Nichol, OpenAI, 2021

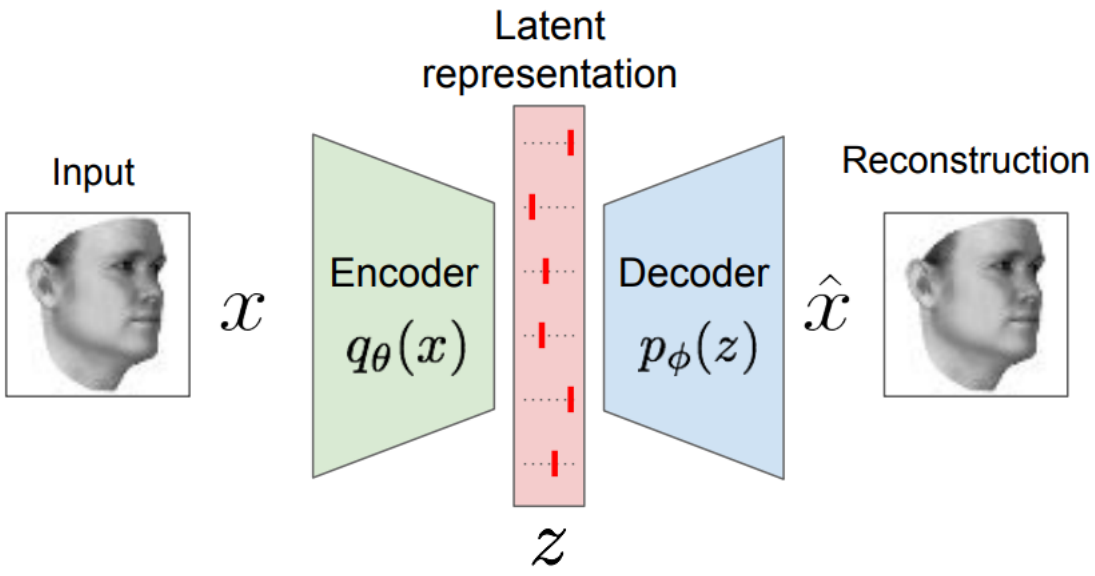
Generative models are unsupervised

- Modelling densities also gives us a generative model of the data (as long as we can draw samples)
- Allows us to ‘see’ what the model has and hasn’t learned

"Diffusion Models Beat GANs on Image Synthesis"
Dhariwal & Nichol, OpenAI, 2021

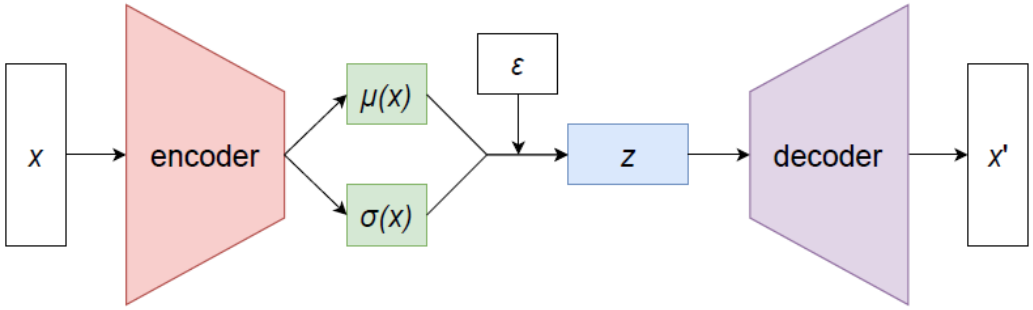
Unsupervised representations (1)

- Latent space of an autoencoder



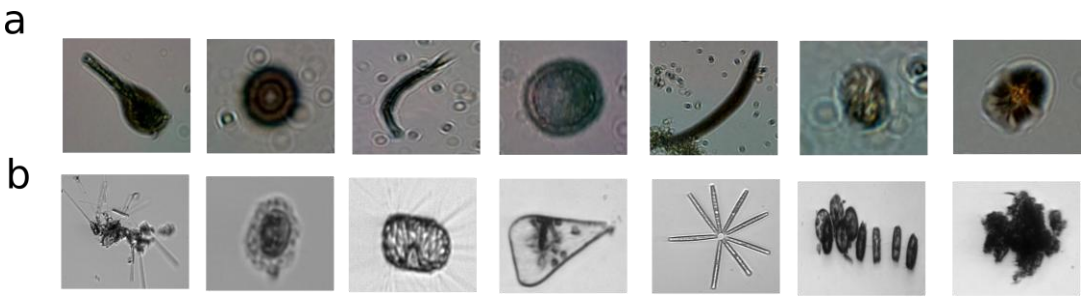
Unsupervised representations (2)

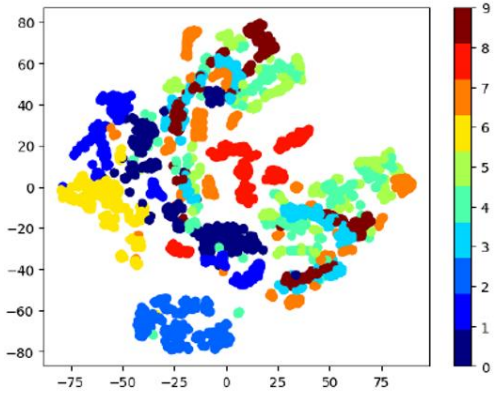
- Latent space of a variational autoencoder



Unsupervised representations (3)

- What about real-world fine-grained potential scenarios? Let's look at an example





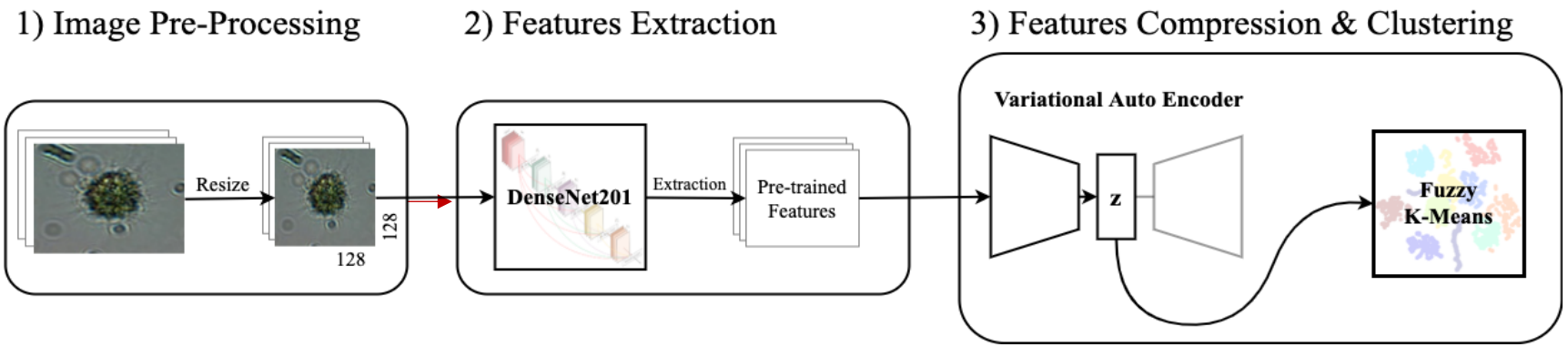
(a) Original images.

- Features should ideally be as much discriminative as possible

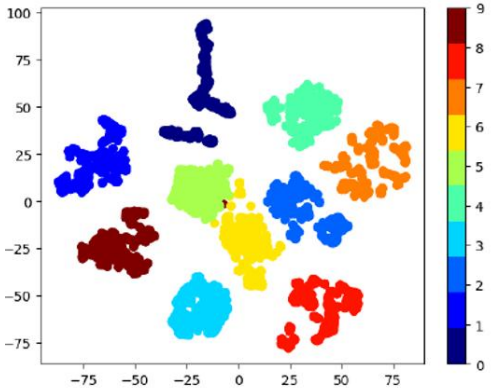
- Latent spaces of generative models trained on images -> may not be enough.

Unsupervised representations (4)

- What about real-world fine-grained potential scenarios?



- We can exploit transfer learning with a pre-trained model (e.g., on ImageNet);
- Train a generative model (e.g., a VAE) to compress deep features

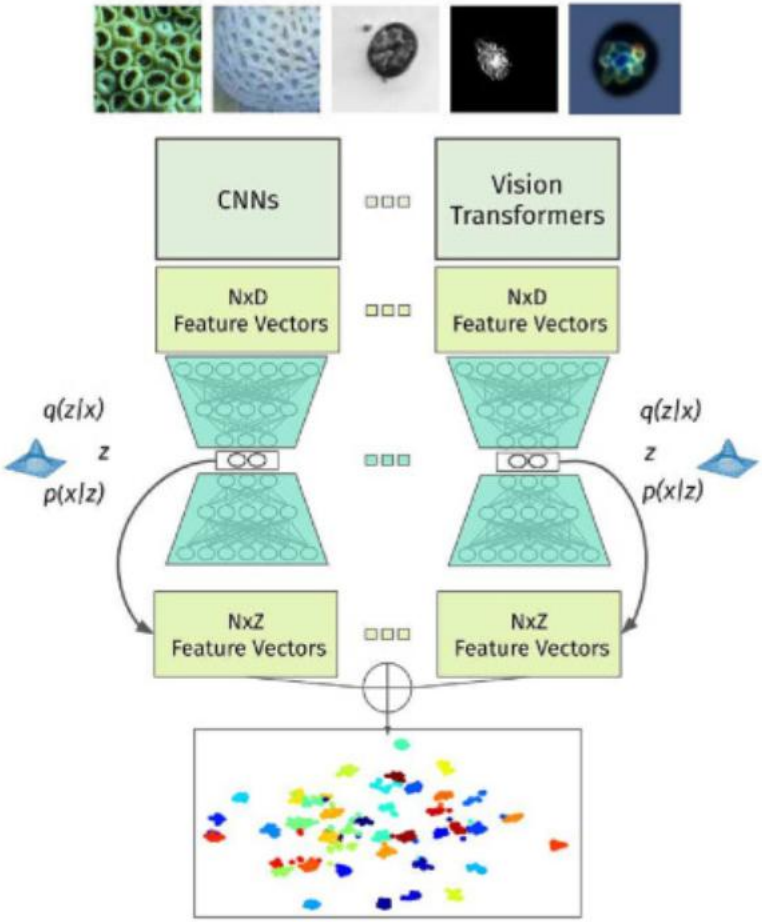


(b) Pre-trained features.

Unsupervised representations (5)

- What about real-world fine-grained potential scenarios?

- How can we can further improve the unsupervised representation?;
- A possibility is by ensembling VAEs trained on top of different models;

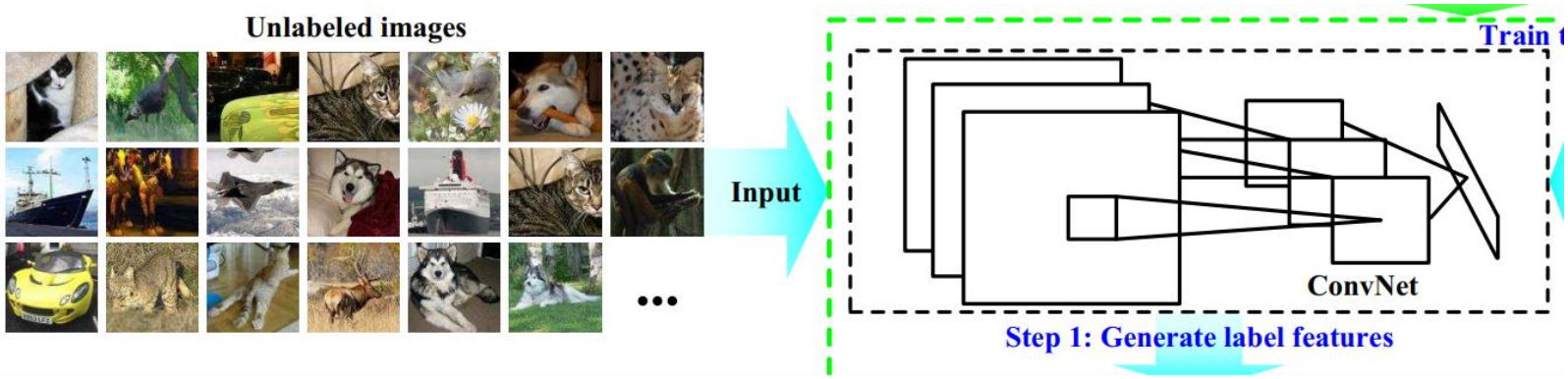


UniGe | MaLGa

Deep clustering

Deep Adaptive Image Clustering (1)

- It is a single-stage ConvNet-based method to cluster images
- Image clustering task framed a binary pairwise-classification problem to judge whether pairs of images belong to the same clusters



Deep Adaptive Image Clustering (2)

- Given a set of unlabeled images $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^{n}$

$r_{ij} = 1$ *if* x_i *and* x_j *belong to the same cluster*

- Denote training data as: $\mathcal{D} = \{(\mathbf{x}_i, \mathbf{x}_j, r_{ij})\}_{i=1,j=1}^{n}$ $\mathbf{x}_i,\ \mathbf{x}_j \in \mathcal{X}$ $r_{ij} \in \mathcal{Y}$

Unknown!

- The objective function is:

$$\min_{\mathbf{w}} \mathbf{E}(\mathbf{w}) = \sum_{i,j} L(r_{ij}, g(\mathbf{x}_i, \mathbf{x}_j; \mathbf{w}))$$

- And specifically:

$$L(r_{ij}, g(\mathbf{x}_i, \mathbf{x}_j; \mathbf{w})) =$$
$$-r_{ij} \log(g(\mathbf{x}_i, \mathbf{x}_j; \mathbf{w})) - (1 - r_{ij}) \log(1 - g(\mathbf{x}_i, \mathbf{x}_j; \mathbf{w})).$$

Deep Adaptive Image Clustering (3)

- First, , Images are represented with label features , exploiting the last fully connected layer with k neurons (k is the number of clusters)

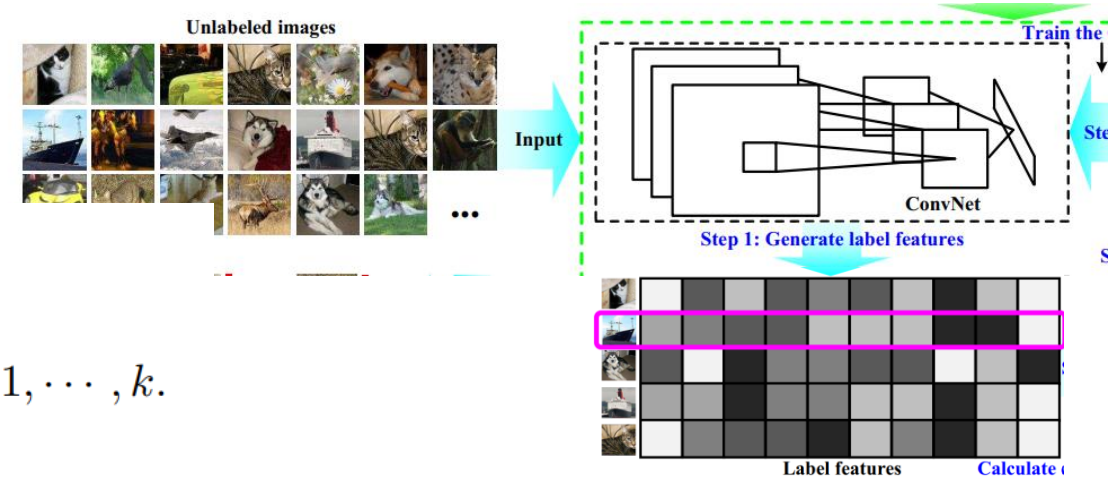
$$\mathcal{L} = \{\mathbf{l}_i \in \mathbb{R}^k\}_{i=1}^n$$

- Imposing a constraint such that $\forall\, i,\ \| \mathbf{l}_i \|_2 = 1,\ \text{and}\ l_{ih} \geq 0,\ h = 1, \cdots, k,$
- Cosine similarity is used as $g(\mathbf{x}_i, \mathbf{x}_j; \mathbf{w})$
- Thus $g(\mathbf{x}_i, \mathbf{x}_j; \mathbf{w}) = f(\mathbf{x}_i; \mathbf{w}) \cdot f(\mathbf{x}_j; \mathbf{w}) = \mathbf{l}_i \cdot \mathbf{l}_j$

With the constraint above:

$$\min_{\mathbf{w}} \mathbf{E}(\mathbf{w}) = \sum_{i,j} L(r_{ij}, \mathbf{l}_i \cdot \mathbf{l}_j),$$

$$\text{s.t.}\ \forall\, i,\ \| \mathbf{l}_i \|_2 = 1,\ \text{and}\ l_{ih} \geq 0,\ h = 1, \cdots, k.$$

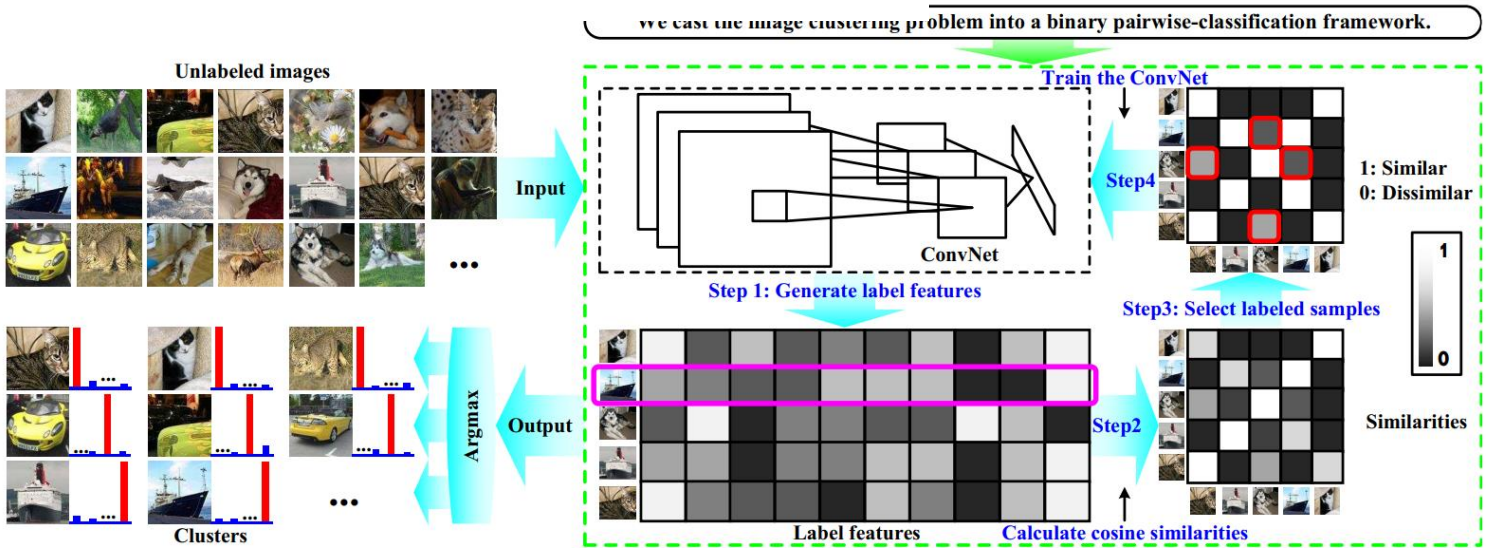


Deep Adaptive Image Clustering (4)

Exploiting a threshold on similarity we get similar images ($r_{ij} = 1$) and Dissimilar ($r_{ij} = 0$)

Two adaptive thresholds are used for selecting training samples:

$$r_{ij} := \begin{cases} 1, & \text{if } \mathbf{l}_i \cdot \mathbf{l}_j \geq u(\lambda), \\ 0, & \text{if } \mathbf{l}_i \cdot \mathbf{l}_j < l(\lambda), \\ \text{None}, & \text{otherwise}, \end{cases} \quad i,\ j = 1, \cdots, n,$$



Deep Adaptive Image Clustering (5)

- Curriculum learning -> number of selected samples increases with time
- Penalty term $u(\lambda) - l(\lambda)$
- Considering that λ increases with clustering;
- Considering that:

$$u(\lambda) \propto -\lambda, \; l(\lambda) \propto \lambda$$

Objective function can become:

$$\min_{\mathbf{w},\lambda} \mathbf{E}(\mathbf{w}, \lambda) = \sum_{i,j} v_{ij} L(r_{ij}, \mathbf{l}_i \cdot \mathbf{l}_j) + u(\lambda) - l(\lambda),$$

$$\text{s.t. } l(\lambda) \leq u(\lambda),$$

$$v_{ij} \in \{0, 1\}, \; i, \; j = 1, \cdots, n,$$

$$\forall \, i, \; \| \, \mathbf{l}_i \, \|_2 = 1, \text{ and } l_{ih} \geq 0, \; h = 1, \cdots, k,$$

$$r_{ij} := \begin{cases} 1, & \text{if } \mathbf{l}_i \cdot \mathbf{l}_j \geq u(\lambda), \\ 0, & \text{if } \mathbf{l}_i \cdot \mathbf{l}_j < l(\lambda), \\ \text{None}, & \text{otherwise}, \end{cases} \quad i, \; j = 1, \cdots, n,$$

where $\mathbf{v}$ is an indicator coefficient, *i.e.*,

$$v_{ij} := \begin{cases} 1, & \text{if } r_{ij} \in \{0, 1\}, \\ 0, & \text{otherwise}, \end{cases} \quad i, \; j = 1, \cdots, n,$$

Deep Adaptive Image Clustering (5)

$$\min_{\mathbf{w},\lambda} \mathbf{E}(\mathbf{w},\lambda) = \sum_{i,j} v_{ij} L(r_{ij}, \mathbf{l}_i \cdot \mathbf{l}_j) + u(\lambda) - l(\lambda),$$

$$\text{s.t. } l(\lambda) \leq u(\lambda),$$

$$v_{ij} \in \{0,1\},\ i,\ j = 1, \cdots, n,$$

$$\forall\, i,\ \| \mathbf{l}_i \|_2 = 1,\ \text{and } l_{ih} \geq 0,\ h = 1, \cdots, k,$$

$$r_{ij} := \begin{cases} 1, & \text{if } \mathbf{l}_i \cdot \mathbf{l}_j \geq u(\lambda), \\ 0, & \text{if } \mathbf{l}_i \cdot \mathbf{l}_j < l(\lambda), \\ \text{None}, & \text{otherwise}, \end{cases} \quad i,\ j = 1, \cdots, n,$$

where $\mathbf{v}$ is an indicator coefficient, *i.e.*,

$$v_{ij} := \begin{cases} 1, & \text{if } r_{ij} \in \{0,1\}, \\ 0, & \text{otherwise}, \end{cases} \quad i,\ j = 1, \cdots, n,$$

W and λ are optimized alternately:

λ is fixed

$$\min_{\mathbf{w}} \mathbf{E}(\mathbf{w}) = \sum_{i,j} v_{ij} L(r_{ij}, f(\mathbf{x}_i; \mathbf{w}) \cdot f(\mathbf{x}_j; \mathbf{w})). \quad (10)$$

w is fixed

$$\min_{\lambda} \mathbf{E}(\lambda) = u(\lambda) - l(\lambda).$$

Updated through gradient descend

$$\lambda := \lambda - \eta \cdot \frac{\partial \mathbf{E}(\lambda)}{\partial \lambda}, \quad (12)$$

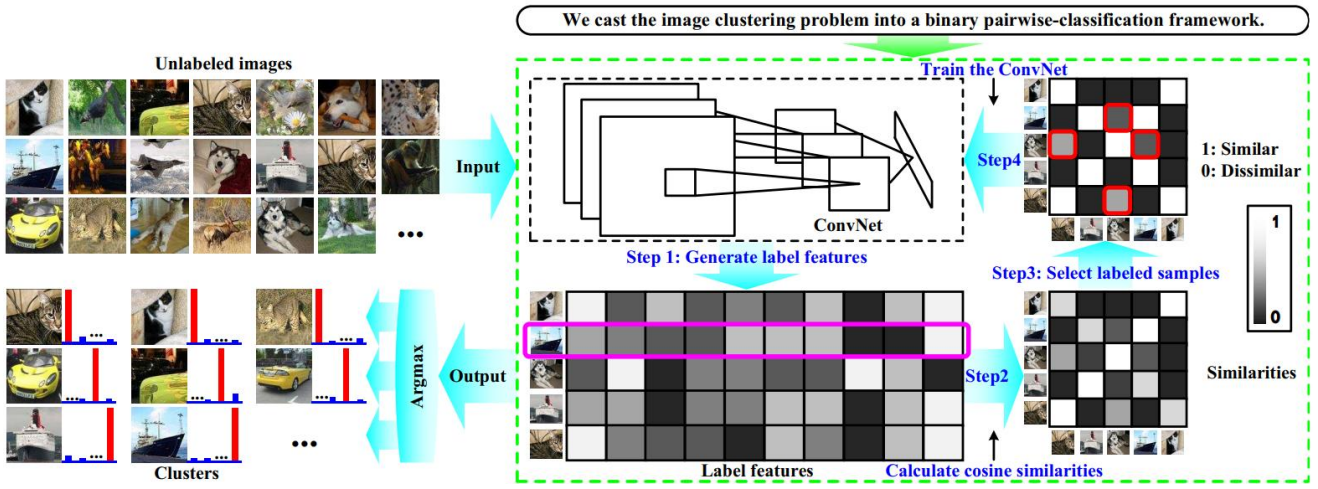
Updated through back prop

Deep Adaptive Image Clustering (6)

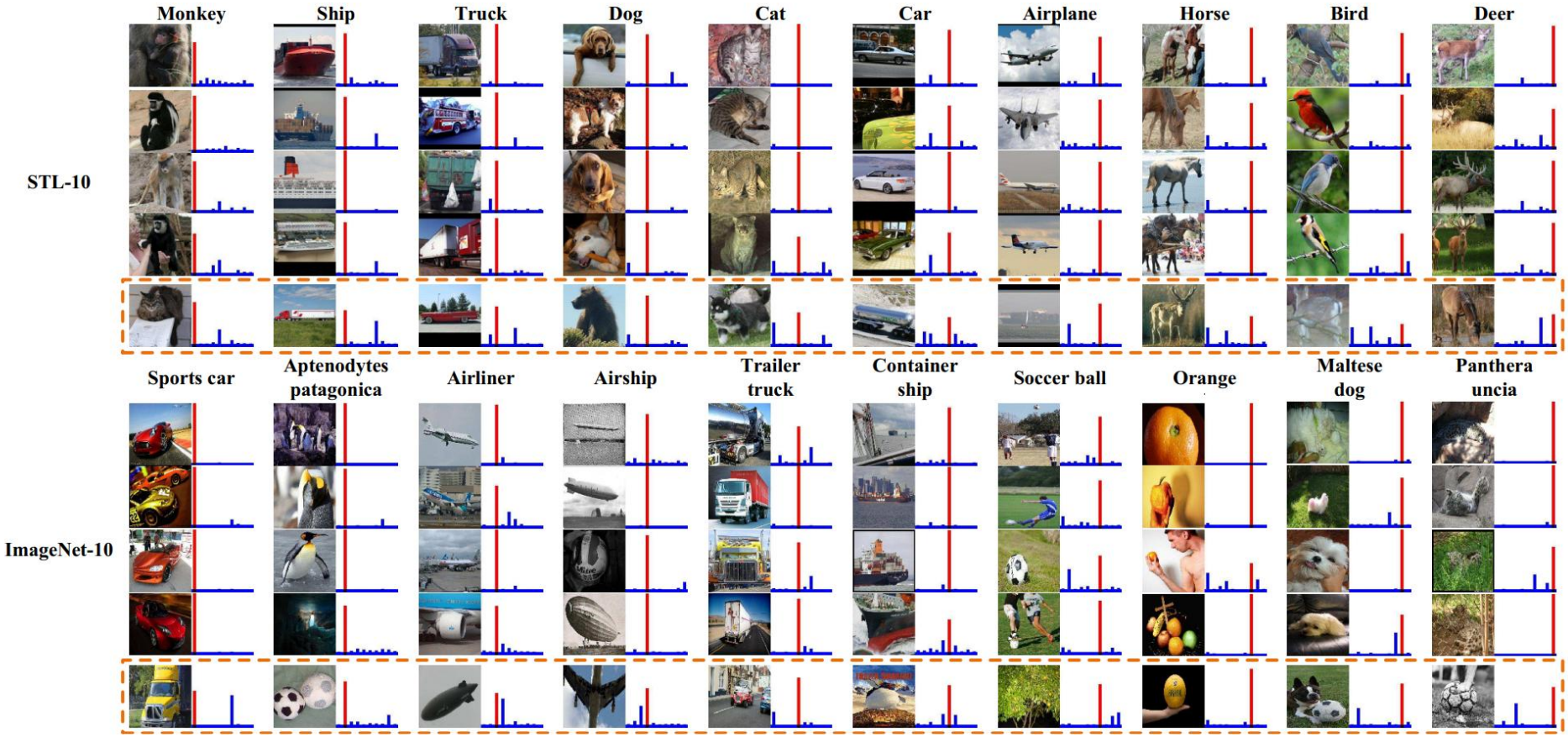
Algorithm 1 Deep Adaptive Clustering

Input: Dataset $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^{n}$, $f_{\mathbf{w}}$, λ, $u(\lambda)$, $l(\lambda)$, η, m.
Output: Cluster label c_i of $\mathbf{x}_i \in \mathcal{X}$.

Randomly initialize $\mathbf{w}$;
repeat
  for all $k \in \{1, 2, \cdots, \lfloor \frac{n}{m} \rfloor\}$ **do**
    Sample batch $\mathcal{X}_k$ from $\mathcal{X}$; // m images per batch
    Select training samples from $\mathcal{X}_k$; // Eq. (6)
    Calculate the indicator parameter $\mathbf{v}$; // Eq. (8)
    Update $\mathbf{w}$ by minimizing Eq. (10);
  end for
  Update λ according to Eq. (12);
until $l(\lambda) > u(\lambda)$
for all $\mathbf{x}_i \in \mathcal{X}$ **do**
  $\mathbf{l}_i := f(\mathbf{x}_i; \mathbf{w})$;
  $c_i := \arg\max_h(l_{ih})$;
end for



Deep Adaptive Image Clustering (8)



UniGe | MaLGa

Deep cluster (Unsupervised pretext tasks)

Deep cluster

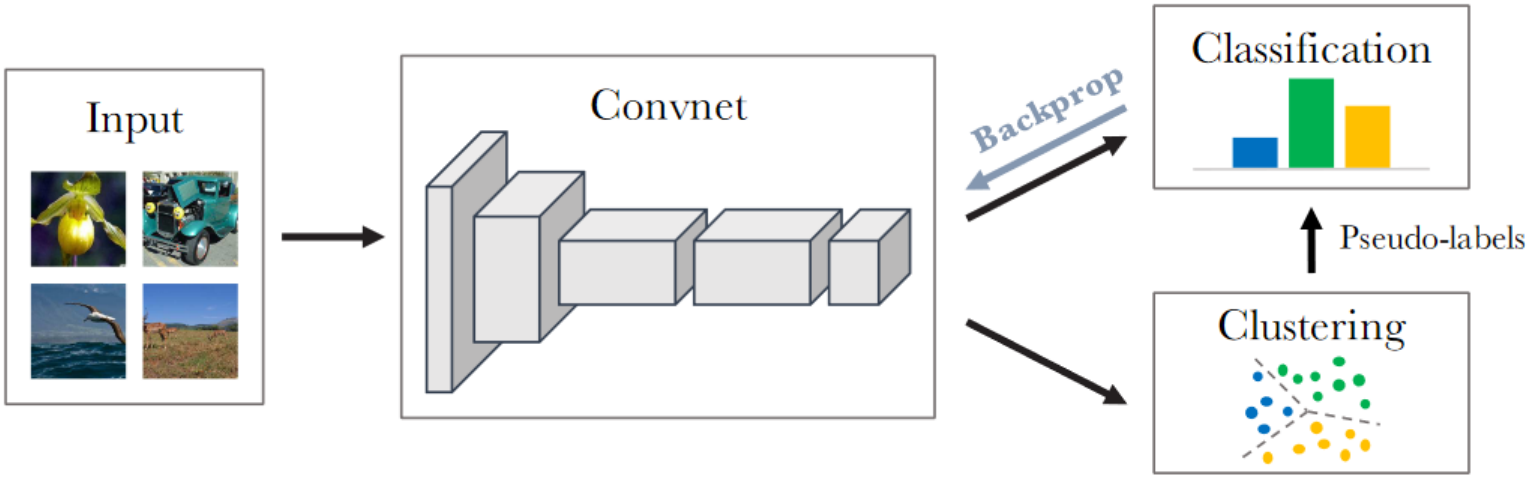
Many self-supervised methods use *pretext* tasks to generate surrogate labels and formulate an unsupervised learning problem as a supervised one. Some examples include rotation prediction, image colorization, jigsaw puzzles etc. However, such pretext tasks are domain-dependent and require expertise to design them.

DeepCluster is an unsupervised method that brings a different approach: this method doesn't require domain-specific knowledge and can be used to learn deep representations for scenarios where annotated data is scarce.

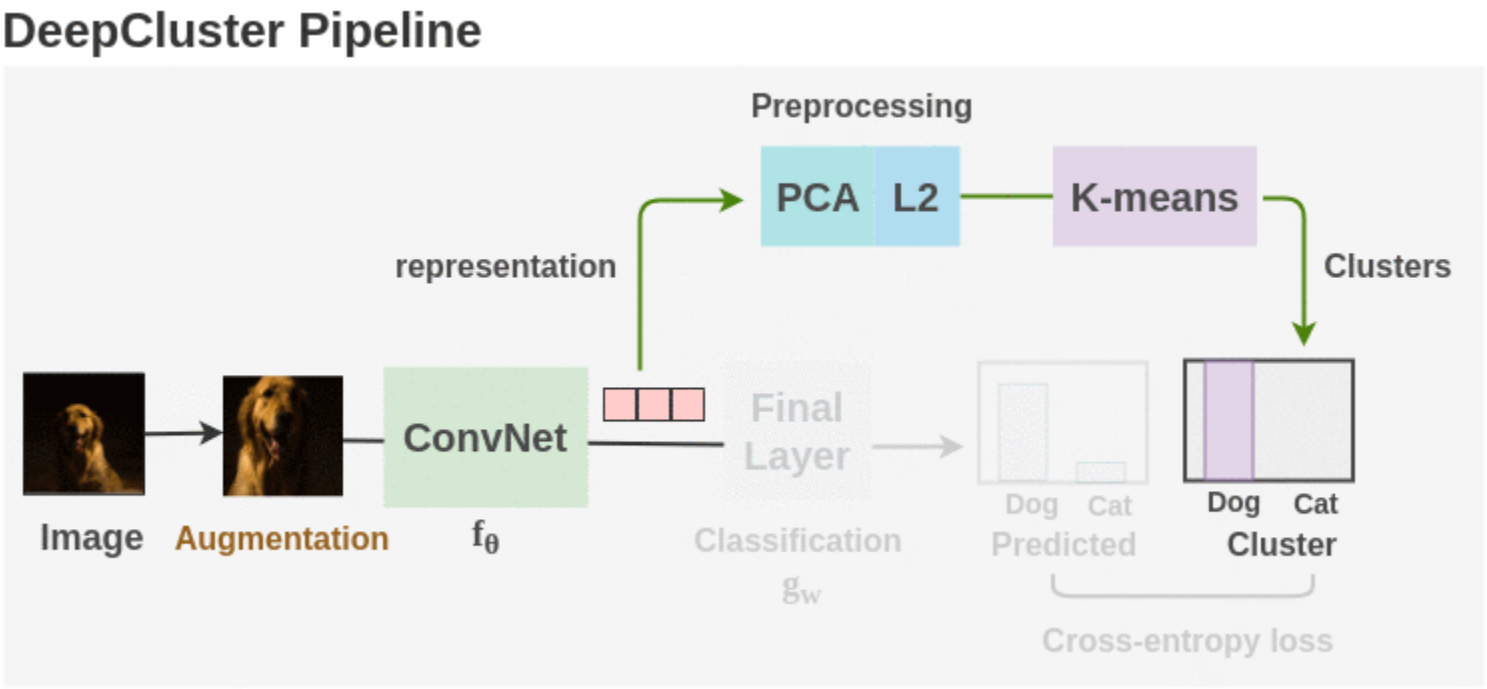
DeepCluster combines two pieces: unsupervised clustering and deep neural networks.

It proposes an **end-to-end method** to jointly learn parameters of a deep neural network and the cluster assignments of its representations. The features are generated and clustered iteratively to get both a trained model and labels as output artifacts.

Introduction



DeepCluster pipeline



DeepCluster pipeline

- Unlabeled images are taken and **augmentations** are applied to them.

Then, an **ConvNet** architecture such as **AlexNet** or **VGG-16** is used as the feature extractor.

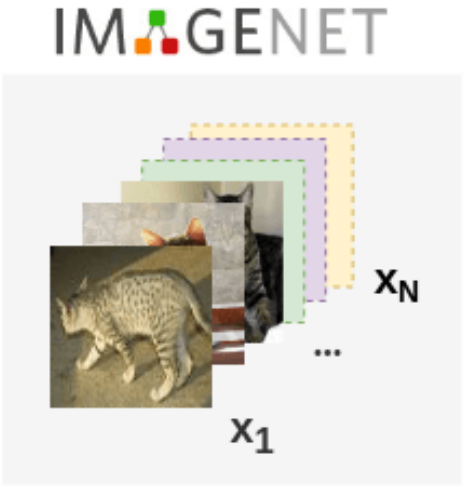
- Initially, the **ConvNet** is initialized with randomly weights and we take the **feature vector** from the layer before the final classification head.

- Then, **PCA** is used to reduce the dimension of the **feature vector** along with whitening and **L2 normalization**. Finally, the processed features are passed to **K-means** to get cluster assignment for each image.
- These cluster assignments are used as **pseudo-labels** and the ConvNet is trained to predict these clusters. Cross-entropy loss is used to gauge the performance of the model. The model is trained for 100 epochs with the **clustering** step occurring once per epoch.
- Finally, we can take the **representations** learned and use it for downstream tasks.

DeepCluster pipeline: going step by step

1. **Training Data**

   We take unlabeled images from the ImageNet dataset which consist of 1.3 million images uniformly distributed into 1000 classes. These images are prepared in mini-batches of 256.



DeepCluster pipeline: going step by step

The training set of N images can be denoted by:

$$X = \{x_1, x_2, \ldots, x_N\}$$

2. Data Augmentation

Transformations are applied to the images so that the features learned are invariant to augmentations.

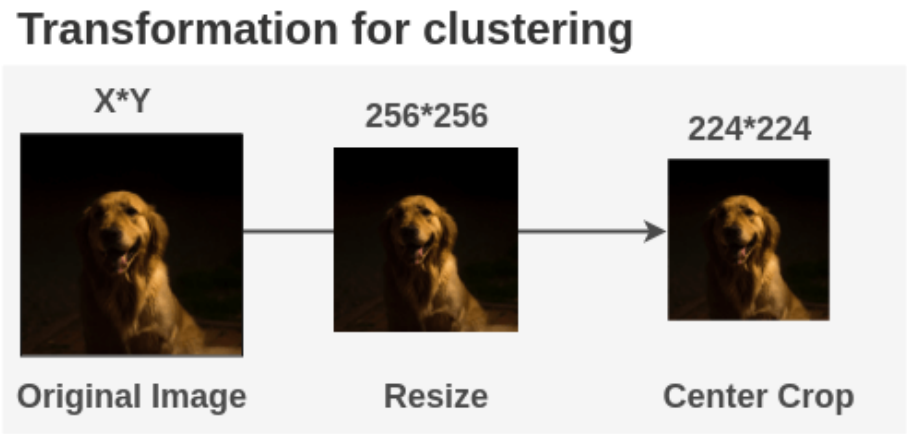
Two different augmentations are done:

- Case 1: when sending the image representations to the clustering algorithm and
- Case 2: when training the model to learn representations

DeepCluster pipeline: going step by step

Case 1: Transformation when doing clustering

When model representations are to be sent for clustering, random augmentations are not used. The image is simply resized to 256*256 and the center crop is applied to get 224*224 image. Then normalization is applied.



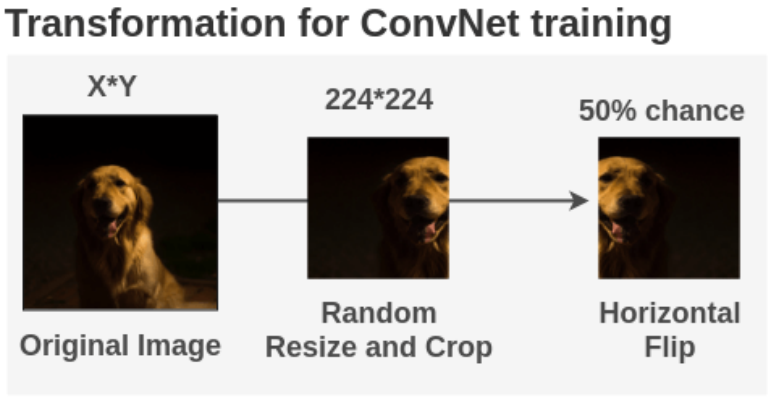
DeepCluster pipeline: going step by step

Case 2: Transformation when training model

When the model is trained on image and labels, then we use random augmentations.

The image is cropped to a random size and aspect ratio and then resized to 224*224.

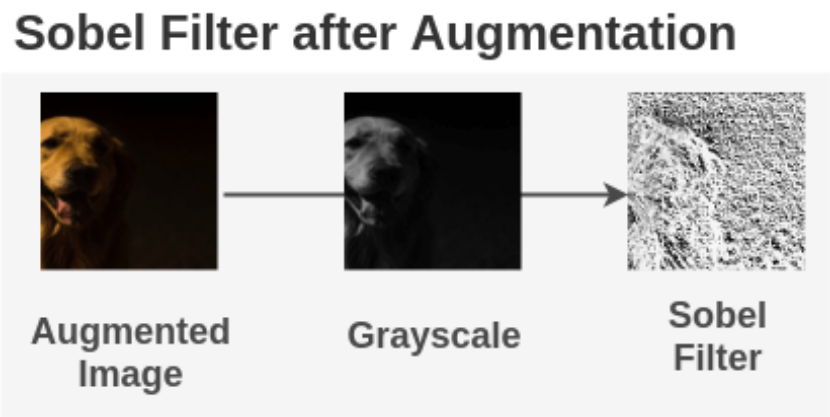
Then, the image is horizontally flipped with a 50% chance. Finally, we normalize the image with ImageNet mean and std



DeepCluster pipeline: going step by step

Sobel Transformation

Once we get the normalized image, we convert it into grayscale. Then, we increase the local contrast of the image using the Sobel filters.

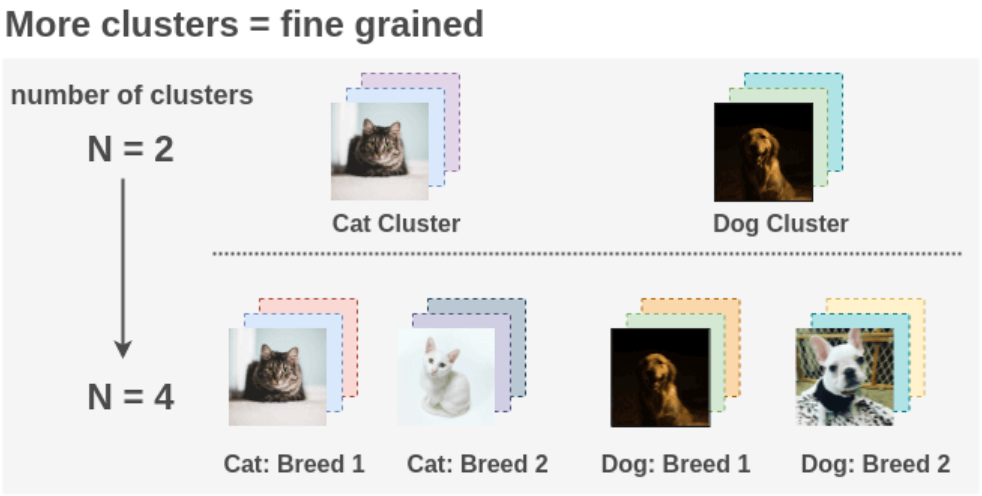


DeepCluster pipeline: going step by step

3. Decide Number of Clusters (Classes)

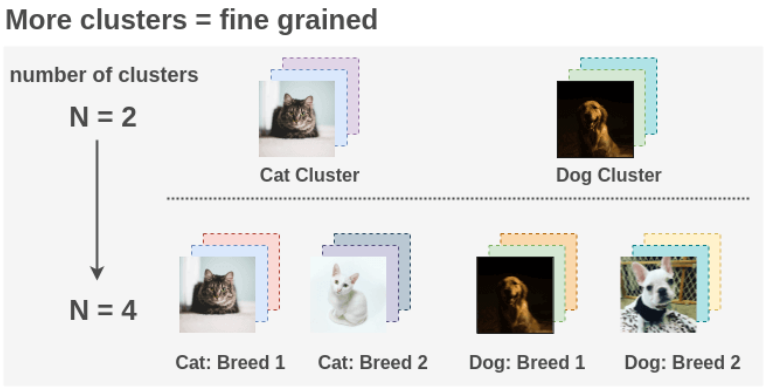
To perform clustering, we need to decide the number of clusters.

This will be the number of classes the model will be trained on.



DeepCluster pipeline: going step by step

3. Decide Number of Clusters (Classes)



By default, ImageNet has 1000 classes, but the paper uses 10,000 clusters as this gives more fine-grained grouping of the unlabeled images.

For example, if you previously had a grouping of cats and dogs and you increase clusters, then groupings of breeds of the cat and dog could be created.

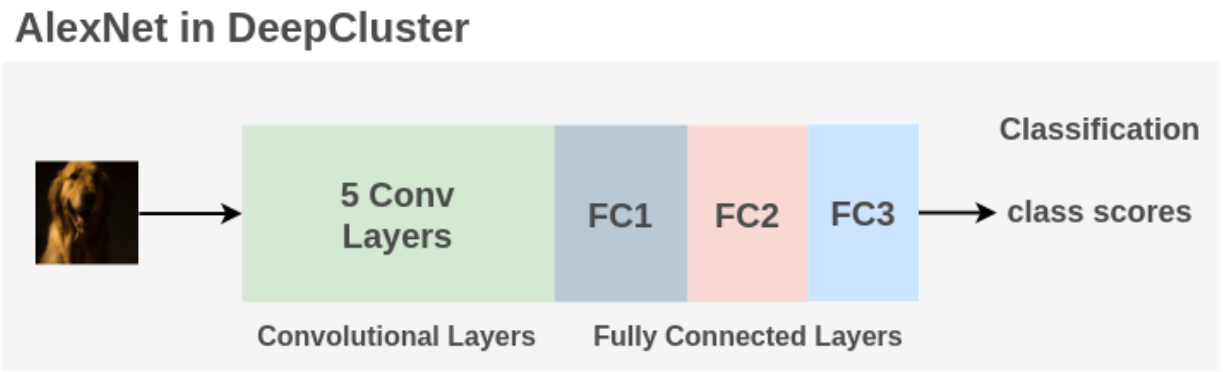
DeepCluster pipeline: going step by step

4. Model Architecture

The paper primarily uses **AlexNet** architecture consisting of **5 convolutional layers** and 3 fully connected layers.

The Local Response Normalization layers are removed and Batch Normalization is applied instead. Dropout is also added.

Alternatively, it has also tried replacing AlexNet by **VGG-16** with batch normalization to see impact on performance

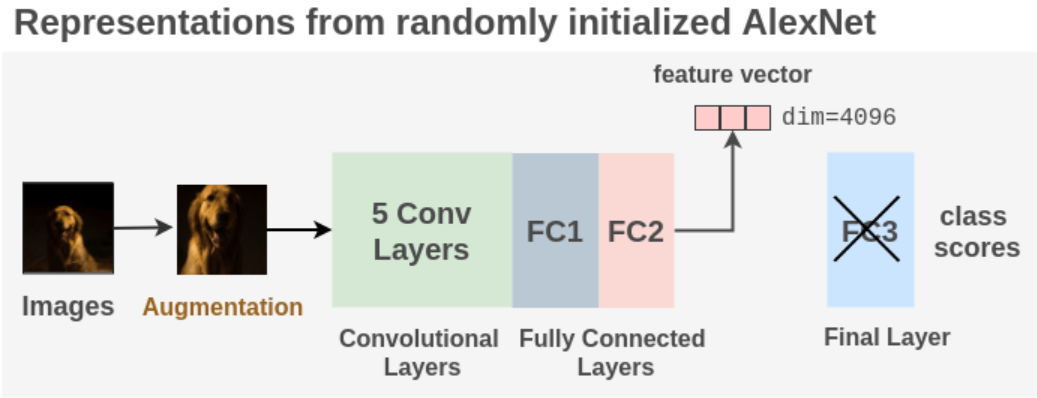


DeepCluster pipeline: going step by step

5. Generating the initial labels

To generating initial labels for the model to train on, we initialize AlexNet with random weights and the last fully connected layer FC3 removed.

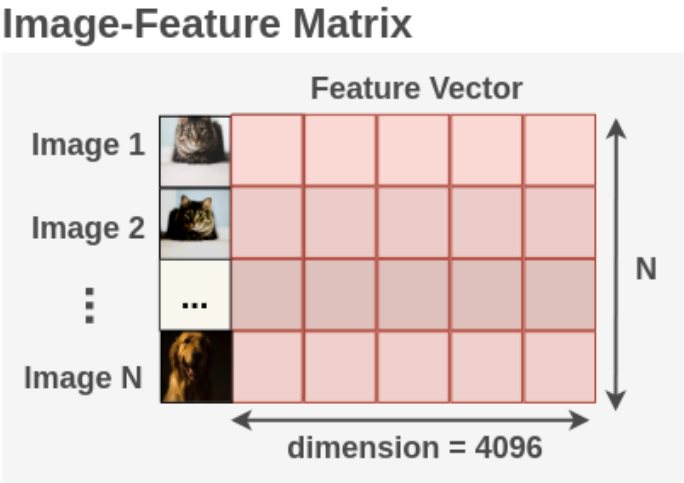
We perform a forward pass on the model on images and take the feature vector coming from the second fully connected layer FC2 of the model on an image. This feature vector has a dimension of 4096.



DeepCluster pipeline: going step by step

This process is repeated for all images in the batch for the whole dataset.

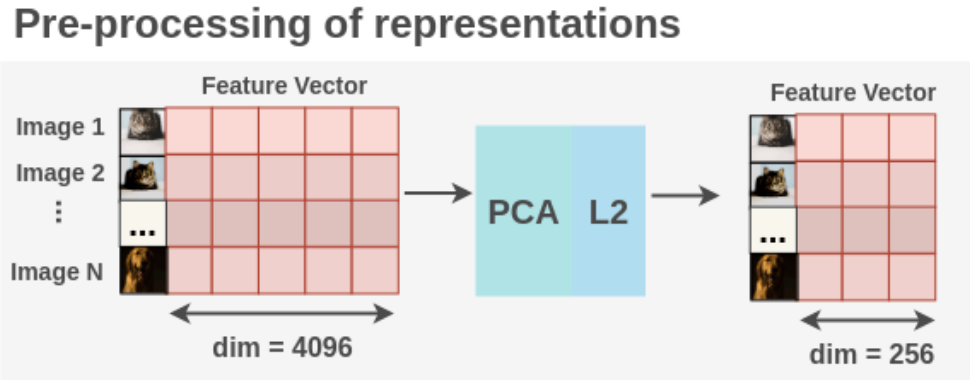
Thus, if we have N total images, we will have an image-feature matrix of [N, 4096].



DeepCluster pipeline: going step by step

6. Clustering

Before performing clustering, dimensionality reduction is applied to the image-feature matrix.



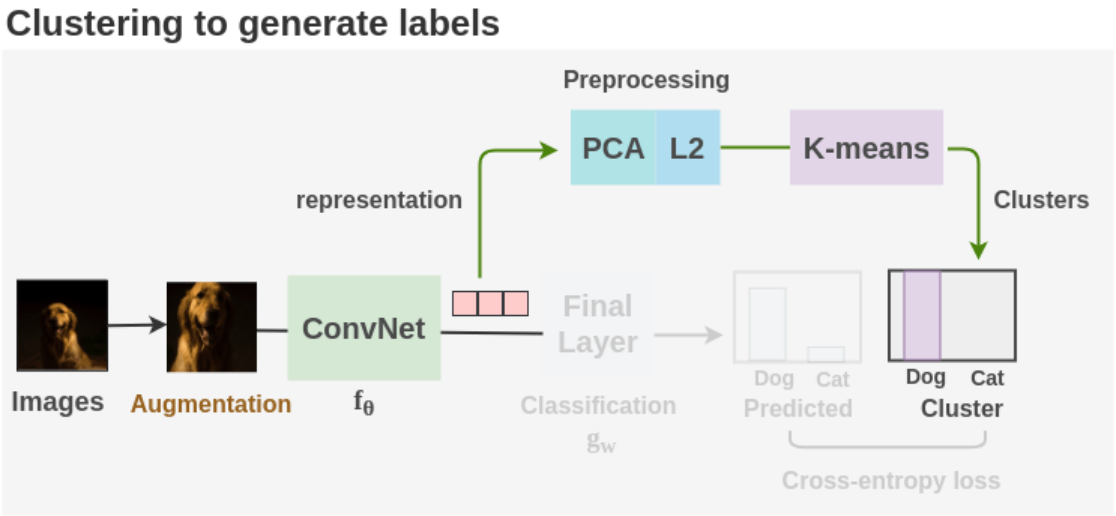
For dimensionality reduction Principal Component Analysis(PCA) is applied to the features to reduce them from 4096 dimensions to 256 dimensions. The values are also whitened.
Faiss library provides an efficient implementation of PCA which can be applied for some image-feature matrix x, it allows to perform this at scale.

DeepCluster pipeline: going step by step

Then, L2 normalization is applied to the values we get after PCA.

Thus, we finally get a matrix of (N, 256) for total N images.

Now, K-means clustering is applied to the pre-processed features to get images and their corresponding clusters. These clusters will act as the pseudo-labels on which the model will be trained.



DeepCluster pipeline: going step by step

It's used a particular implementation of k-means (Johnson's implementation from the paper "Billion-scale similarity search with GPUs", available in the FAISS library).

Since clustering has to be run on all the images, it takes one-third of the total training time.

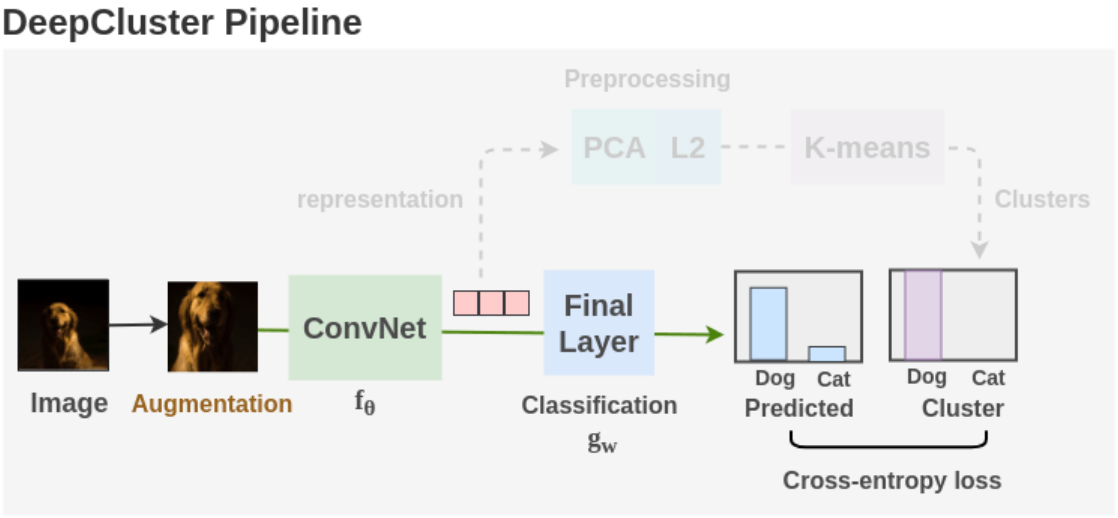
After clustering is done, new batches of images are created such that images from each cluster has an equal chance of being included. Random augmentations are applied to these images.

DeepCluster pipeline: going step by step

7. Representation Learning

Once we have the images and clusters, we train our ConvNet model like regular supervised learning.

We use a batch size of 256 and use cross-entropy loss to compare model predictions to the ground truth cluster label. The model learns useful representations



DeepCluster pipeline: going step by step

8. Switching between model training and clustering

The model is trained for 500 epochs. The clustering step is run once at the start of each epoch to generate pseudo-labels for the whole dataset.

Then, the regular training of ConvNet using cross-entropy loss is continued for all the batches.

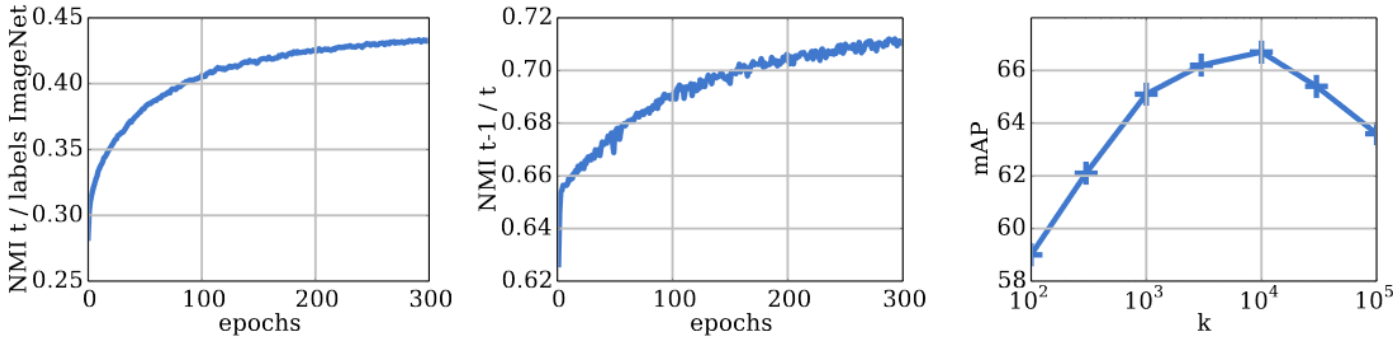
The paper uses SGD optimizer with momentum of 0.9, learning rate of 0.05 and weight decay of 10^{-5}

They trained it on Pascal P100 GPU*

* The code is available on GitHub

Validation

The information shared between two different assignments, A and B, of the same data is measured by the Normalized Mutual Information (NMI), defined as:



(a) Clustering quality (b) Cluster reassignment (c) Influence of k

$$\mathrm{NMI}(A;B) = \frac{\mathrm{I}(A;B)}{\sqrt{\mathrm{H}(A)\mathrm{H}(B)}}$$, where I denotes the mutual information and H the entropy

This measure can be applied to any assignment coming from the clusters or the true labels. If the two assignments A and B are independent, the NMI is equal to 0. If one of them is deterministically predictable from the other, the NMI is equal to 1.

In general, to recap …

Labeling huge amounts of data is too expensive, even collecting balanced, unbiased data is often unfeasible

Unsupervised learning is becoming more and more important in scientific and practical perspectives

(Pseudo-)Labeling (clustering) and training/fine-tuning is a viable class of approaches to explore

Self-supervised contrastive learning is another interesting alternative class of methods to be considered