Unsupervised Pre-Training of Image Features on Non-Curated Data

Mathilde Caron[1,2], Piotr Bojanowski[1], Julien Mairal[2], and Armand Joulin[1]

[1]Facebook AI Research
[2]Univ. Grenoble Alpes, Inria, CNRS, Grenoble INP, LJK, 38000 Grenoble, France

Abstract

Pre-training general-purpose visual features with convolutional neural networks without relying on annotations is a challenging and important task. Most recent efforts in unsupervised feature learning have focused on either small or highly curated datasets like ImageNet, whereas using non-curated raw datasets was found to decrease the feature quality when evaluated on a transfer task. Our goal is to bridge the performance gap between unsupervised methods trained on curated data, which are costly to obtain, and massive raw datasets that are easily available. To that effect, we propose a new unsupervised approach which leverages self-supervision and clustering to capture complementary statistics from large-scale data. We validate our approach on 96 *million images from YFCC100M [42], achieving state-of-the-art results among unsupervised methods on standard benchmarks, which confirms the potential of unsupervised learning when only non-curated raw data are available. We also show that pre-training a supervised VGG-16 with our method achieves* 74.9% *top-*1 *classification accuracy on the validation set of ImageNet, which is an improvement of* +0.8% *over the same network trained from scratch. Our code is available at* https://github.com/facebookresearch/DeeperCluster.

1. Introduction

Pre-trained convolutional neural networks, or convnets, are important components of image recognition applications [7, 8, 38, 46]. They improve the generalization of models trained on a limited amount of data [39] and speed up the training on applications when annotated data is abundant [20]. Convnets produce good generic representations when they are pre-trained on large supervised datasets like ImageNet [11]. However, designing such fully-annotated datasets has required a significant effort from the research community in terms of data cleansing and manual labeling.

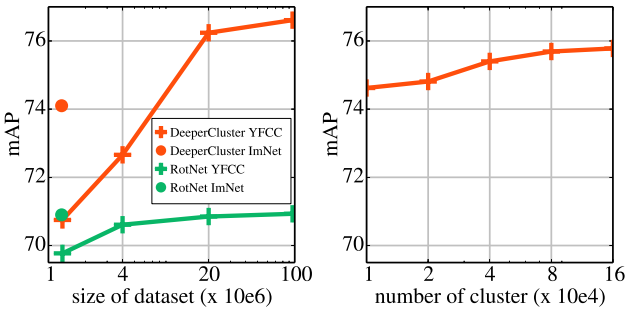

Figure 1: Influence of amount of data (*left*) and number of clusters (*right*) on the features quality. We report validation mAP on Pascal VOC classification task (FC68 setting).

Scaling up the annotation process to datasets that are orders of magnitude bigger raises important difficulties. Using raw metadata as an alternative has been shown to perform comparatively well [23, 41], even surpassing ImageNet pre-training when trained on billions of images [30]. However, metadata are not always available, and when they are, they do not necessarily cover the full extent of a dataset. These difficulties motivate the design of methods that learn transferable features without using any annotation.

Recent works describing unsupervised approaches have reported performances that are closing the gap with their supervised counterparts [6, 15, 51]. However, the best performing unsupervised methods are trained on ImageNet, a curated dataset made of carefully selected images to form well-balanced and diversified classes [11]. Simply discarding the labels does not undo this careful selection, as it only removes part of the human supervision. Because of that, previous works that have experimented with non-curated raw data report a degradation of the quality of features [6, 12]. In this work, we aim at learning good visual representations from unlabeled and non-curated datasets. We focus on the YFCC100M dataset [42], which contains 99 million images from the Flickr photo-sharing website. This dataset is unbalanced, with a "long-tail" distribution of

hashtags contrasting with the well-behaved label distribution of ImageNet (see Appendix). For example, *guenon* and *baseball* correspond to labels with 1300 associated images in ImageNet, while there are respectively 226 and 256,758 images associated with these hashtags in YFCC100M. Our goal is to understand if trading manually-curated data for scale leads to an improvement in the feature quality.

We propose a new unsupervised approach specifically designed to leverage large amount of raw data. Indeed, training on large-scale non-curated data requires (i) model complexity to increase with dataset size; (ii) model stability to data distribution changes. A simple yet effective solution is to combine methods from two domains of unsupervised learning: clustering and self-supervision. Since clustering methods, like DeepCluster [6], build supervision from *inter-image* similarities, the task at hand becomes inherently more complex when the number of images increases. In addition, DeepCluster captures finer relations between images when the number of clusters scales with the dataset size. Clustering approaches infer target labels at the same time as features are learned. Thus, target labels evolve during training, making clustering-based approaches unstable. Furthermore, these methods are sensitive to data distribution as they rely directly on cluster structure in the underlying data. Explicitly dealing with unbalanced category distribution might be a solution but it assumes that we know the distribution of the latent classes. We design our method without this assumption. On the other hand, self-supervised learning [10] consists in designing a pretext task by predicting pseudo-labels automatically extracted from input signals [12]. In other words, self-supervised approaches, like RotNet [15], leverage *intra-image* statistics to build supervision, which are often independent of the data distribution. However, the dataset size has little impact on the nature of the task and on the performance of the resulting features (see Figure 1). A solution to leveraging larger datasets require manually increasing the difficulty of the self-supervision task [19]. Our approach automatically increases complexity through the clustering strategy.

Method	intra-image statistics	inter-images statistics.	stable to distribution change
Self-Sup (RotNet)	✓	✗	✓
Deep Clustering	✓	✓	✗

Table 1: Training on non-curated large-scale data requires model complexity to increase with dataset size and model stability to data distribution changes. A simple solution is to combine self-supervision and clustering.

The novelty of our method lies in the combination of these two paradigms (Table 1) so that they benefit from one another. Our approach, DeeperCluster, automatically generates targets by clustering the features of the entire dataset, under constraints derived from self-supervision. Due to the "long-tail" distribution of raw non-curated data, processing huge datasets and learning a large number of targets is necessary, making the problem challenging from a computational point of view. For this reason, we propose a hierachical formulation that is suitable for distributed training. This enables the discovery of latent categories present in the "tail" of the image distribution. While our framework is general, in practice we focus on combining the large rotation classification task of Gidaris *et al*. [15] with the clustering approach of Caron *et al*. [6]. Figure 1 left shows that as we increase the number of training images, the quality of features improves to the point where it surpasses those trained without labels on curated datasets. More importantly, we evaluate the quality of our approach as a pre-training step for ImageNet classification. Pre-training a supervised VGG-16 with our unsupervised approach leads to a top-1 accuracy of 74.9%, which is an improvement of $+0.8\%$ over a model trained from scratch. This shows the potential of unsupervised pre-training on large non-curated datasets as a way to improve the quality of visual features.

2. Related Work

Self-supervision. Self-supervised learning builds a pretext task from the input signal to train a model without annotation [10]. Many pretext tasks have been proposed [22, 31, 44, 48], exploiting, amongst others, spatial context [12, 24, 33, 34, 36], cross-channel prediction [27, 28, 52, 53], or the temporal structure of videos [1, 35, 43]. Some pretext tasks explicitly encourage the representations to be either invariant or discriminative to particular types of input tranformations. For example, Dosovitskiy *et al*. [13] consider each image and its transformations as a class to enforce invariance to data transformations. In this paper, we build upon the work of Gidaris *et al*. [15] where the model encourages features to be discriminative for large rotations. Recently, Kolesnikov *et al*. [25] have conducted an extensive benchmark of self-supervised learning methods on different convnet architectures. As opposed to our work, they use curated datasets for pre-training.

Deep clustering. Clustering, along with density estimation and dimensionality reduction, is a family of standard unsupervised learning methods. Various attempts have been made to train convnets using clustering [2, 3, 6, 29, 45, 49, 50]. Our paper builds upon the work of Caron *et al*. [6], in which k-means is used to cluster the visual representations. Unlike our work, they mainly focus on training their approach using ImageNet without labels. Recently, Noroozi *et al*. [34] show that clustering can also be used as a form of distillation to improve the performance of networks

trained with self-supervision. As opposed to our work, they use clustering only as a post-processing step and does not leverage the complementarity between clustering and self-supervision to further improve the quality of features.

Learning on non-curated datasets. Some methods [9, 17, 32] aim at learning visual features from non-curated data streams. They typically use metadata such as hashtags [23, 41] or geolocalization [47] as a source of noisy supervision. In particular, Mahajan *et al*. [30] train a network to classify billions of Instagram images into predefined and clean sets of hashtags. They show that with little human effort, it is possible to learn features that transfer well to ImageNet, even achieving state-of-the-art performance if fine-tuned. As opposed to our work, they use an extrinsic source of supervision that had to be cleaned beforehand.

3. Preliminaries

In this work, we refer to the vector obtained at the penultimate layer of the convnet as a *feature* or *representation*. We denote by f_θ the feature-extracting function, parametrized by a set of parameters θ. Given a set of images, our goal is then to learn a "good" mapping f_{θ^*}. By "good", we mean a function that produces general-purpose visual features that are useful on downstream tasks.

3.1. Self-supervision

In self-supervised learning, a pretext task is used to extract target labels directly from data [12]. These targets can take a variety of forms. They can be categorical labels associated with a multiclass problem, as when predicting the transformation of an image [15, 51] or the ordering of a set of patches [33]. Or they can be continuous variables associated with a regression problem, as when predicting image color [52] or surrounding patches [36]. In this work, we are interested in the former. We suppose that we are given a set of N images $\{x_1, \dots, x_N\}$ and we assign a pseudo-label y_n in $\mathcal{Y}$ to each input x_n. Given these pseudo-labels, we learn the parameters θ of the convet jointly with a linear classifier V to predict pseudo-labels by solving the problem

$$\min_{\theta,V} \frac{1}{N} \sum_{n=1}^{N} \ell(y_n, V f_\theta(x_n)), \tag{1}$$

where ℓ is a loss function. The pseudo-labels y_n are fixed during the optimization and the quality of the learned features entirely depends on their relevance.

Rotation as self-supervision. Gidaris *et al*. [15] have recently shown that good features can be obtained when training a convnet to discriminate between different image rotations. In this work, we focus on their pretext task, *RotNet*, since its performance on standard evaluation benchmarks is among the best in self-supervised learning. This pretext task corresponds to a multiclass classification problem with four categories: rotations in $\{0°, 90°, 180°, 270°\}$. Each input x_n in Eq. (1) is randomly rotated and associated with a target y_n that represents the angle of the applied rotation.

3.2. Deep clustering

Clustering-based approaches for deep networks typically build target classes by clustering visual features produced by convnets. As a consequence, the targets are updated during training along with the representations and are potentially different at each epoch. In this context, we define a latent pseudo-label z_n in $\mathcal{Z}$ for each image n as well as a corresponding linear classifier W. These clustering-based methods alternate between learning the parameters θ and W and updating the pseudo-labels z_n. Between two reassignments, the pseudo-labels z_n are fixed, and the parameters and classifier are optimized by solving

$$\min_{\theta,W} \frac{1}{N} \sum_{n=1}^{N} \ell(z_n, W f_\theta(x_n)), \tag{2}$$

which is of the same form as Eq. (1). Then, the pseudo-labels z_n can be reassigned by minimizing an auxiliary loss function. This loss sometimes coincides with Eq. (2) [3, 49] but some works proposed to use another objective [6, 50].

Updating the targets with k-means. In this work, we focus on the framework of Caron *et al*. [6], *DeepCluster*, where latent targets are obtained by clustering the activations with k-means. More precisely, the targets z_n are updated by solving the following optimization problem:

$$\min_{C \in \mathbb{R}^{d \times k}} \sum_{n=1}^{N} \left[\min_{z_n \in \{0,1\}^k \text{ s.t. } z_n^\top \mathbf{1}=1} \|C z_n - f_\theta(x_n)\|_2^2 \right], \tag{3}$$

C is the matrix where each column corresponds to a centroid, k is the number of centroids, and z_n is a binary vector with a single non-zero entry. This approach assumes that the number of clusters k is known *a priori*; in practice, we set it by validation on a downstream task (see Sec. 5.3). The latent targets are updated every T epochs of stochastic gradient descent steps when minimizing the objective (2).

Note that this alternate optimization scheme is prone to trivial solutions and controlling the way optimization procedures of both objectives interact is crucial. Re-assigning empty clusters and performing a batch-sampling based on an uniform distribution over the cluster assignments are workarounds to avoid trivial parametrization [6].

4. Method

In this section, we describe how we combine self-supervised learning with deep clustering in order to scale

up to large numbers of images and targets.

4.1. Combining self-supervision and clustering

We assume that the inputs $x_1, \ldots, x_N$ are rotated images, each associated with a target label y_n encoding its rotation angle and a cluster assignment z_n. The cluster assignment changes during training along with the visual representations. We denote by $\mathcal{Y}$ the set of possible rotation angles and by $\mathcal{Z}$, the set of possible cluster assignments. A way of combining self-supervision with deep clustering is to add the losses defined in Eq. (1) and Eq. (2). However, summing these losses implicitly assumes that classifying rotations and cluster memberships are two independent tasks, which may limit the signal that can be captured. Instead, we work with the Cartesian product space $\mathcal{Y} \times \mathcal{Z}$, which can potentially capture richer interactions between the two tasks. We get the following optimization problem:

$$\min_{\theta, W} \frac{1}{N} \sum_{n=1}^{N} \ell(y_n \otimes z_n, W f_\theta(x_n)). \tag{4}$$

Note that any clustering or self-supervised approach with a multiclass objective can be combined with this formulation. For example, we could use a self-supervision task that captures information about tiles permutations [33] or frame ordering in a video [43]. However, this formulation does not scale in the number of combined targets, i.e., its complexity is $O(|\mathcal{Y}||\mathcal{Z}|)$. This limits the use of a large number of cluster or a self-supervised task with a large output space [51]. In particular, if we want to capture information contained in the tail of the distribution of non-curated dataset, we may need a large number of clusters. We thus propose an approximation of our formulation based on a scalable hierarchical loss that it is designed to suit distributed training.

4.2. Scaling up to large number of targets

Hierarchical losses are commonly used in language modeling where the goal is to predict a word out of a large vocabulary [5]. Instead of making one decision over the full vocabulary, these approaches split the process in a hierarchy of decisions, each with a smaller output space. For example, the vocabulary can be split into clusters of semantically similar words, and the hierarchical process would first select a cluster and then a word within this cluster.

Following this line of work, we partition the target labels into a 2-level hierarchy where we first predict a super-class and then a sub-class among its associated target labels. The first level is a partition of the images into S super-classes and we denote by y_n the super-class assignment vector in $\{0,1\}^S$ of the image n and by y_{ns} the s-th entry of y_n. This super-class assignment is made with a linear classifier V on top of the features. The second-level of the hierarchy is obtained by partitioning *within each super-class*. We denote

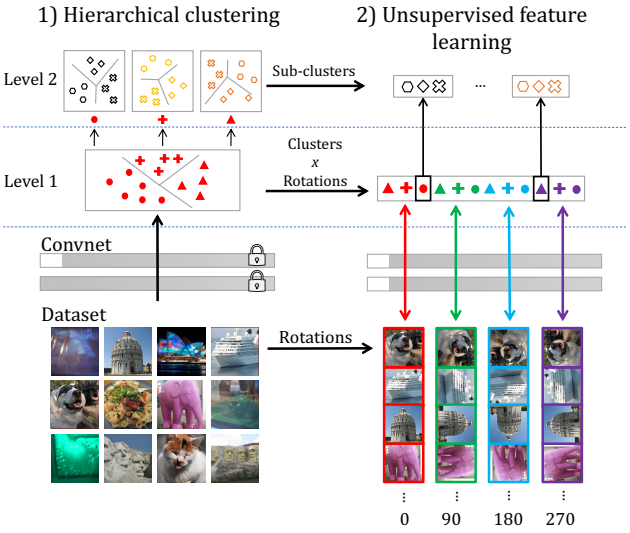

Figure 2: DeeperCluster alternates between a hierachical clustering of the features and learning the parameters of a convnet by predicting both the rotation angle and the cluster assignments in a single hierachical loss.

by z_n^s the vector in $\{0,1\}^{k_s}$ of the assignment into k_s sub-classes for an image n belonging to super-class s. There are S sub-class classifiers $W_1, \ldots, W_S$, each predicting the sub-class memberships within a super-class s. The parameters of the linear classifiers $(V, W_1, \ldots, W_S)$ and θ are jointly learned by minimizing the following loss function:

$$\frac{1}{N} \sum_{n=1}^{N} \left[\ell\big(V f_\theta(x_n), y_n\big) + \sum_{s=1}^{S} y_{ns} \ell\left(W_s f_\theta(x_n), z_n^s\right) \right], \tag{5}$$

where ℓ is the negative log-softmax function. Note that an image that does not belong to the super-class s does not belong either to any of its k_s sub-classes.

Choice of super-classes. A natural partition would be to define the super-classes based on the target labels from the self-supervised task and the sub-classes as the labels produced by clustering. However, this would mean that each image of the entire dataset would be present in each super-class (with a different rotation), which does not take advantage of the hierarchical structure to use a bigger number of clusters.

Instead, we split the dataset into m sets by running k-means with m centroids on the full dataset every T epochs. We then use the Cartesian product between the assignment to these m clusters and the angle rotation classes to form the super-classes. There are $4m$ super-classes, each associated with the subset of data belonging to the corresponding cluster (N/m images if the clustering is perfectly balanced). These subsets are then further split with k-means into k subclasses. This is equivalent to running a hierarchical k-means

with rotation constraints on the full datasets to form our hierarchical loss. We typically use $m = 4$ and $k = 80$k, leading to a total of 320k different clusters split in 4 subsets. Our approach, "DeeperCluster", shares similarities with DeepCluster but is designed to scale to larger datasets. We alternate between clustering the non-rotated images features and training the network to predict both the rotation applied to the input data and its cluster assignment amongst the clusters corresponding to this rotation (Figure 2).

Distributed training. Building the super-classes based on data splits lends itself to a distributed implementation that scales well in the number of images. Specifically, when optimizing Eq. (5), we form as many distributed communication groups of p GPUs as the number of super-classes, i.e., $G = 4m$. Different communication groups share the parameters θ and the super-class classifier V, while the parameters of the sub-class classifiers $W_1, \dots, W_S$ are only shared within a communication group. Each communication group s deals only with the subset of images and the rotation angle associated with the super-class s.

Distributed k-means. Every T epochs, we recompute the super and sub-class assignments by running two consecutive k-means on the entire dataset. This is achieved by first randomly splitting the dataset across different GPUs. Each GPU is in charge of computing cluster assignments for its partition, whereas centroids are updated across GPUs. We reduce communication between GPUs by sharing only the number of assigned elements for each cluster and the sum of their features. The new centroids are then computed from these statistics. We observe empirically that k-means converges in 10 iterations. We cluster 96M features of dimension 4096 into $m = 4$ clusters using 64 GPUs (1 minute per iteration). Then, we split this pool of GPUs into 4 groups of 16 GPUs. Each group clusters around 23M features into 80k clusters (4 minutes per iteration).

4.3. Implementation details

The loss in Eq. (5) is minimized with mini-batch stochastic gradient descent [4]. Each mini-batch contains 3072 instances distributed accross 64 GPUs, leading to 48 instances per GPU per minibatch [18]. We use dropout, weight decay, momentum and a constant learning rate of 0.1. We reassign clusters every 3 epochs. We use the Pascal VOC 2007 classification task without finetuning as a downstream task to select hyper-parameters. In order to speed up experimentations, we initialize the network with RotNet trained on YFCC100M. Before clustering, we perform a whitening of the activations and ℓ_2-normalize each of them. We use standard data augmentations, i.e., cropping of random sizes and aspect ratios and horizontal flips [26]). We use the VGG-16 architecture [40] with batch normalization layers. Following [3, 6, 37], we pre-process images with a Sobel filtering. We train our models on the 96M images from YFCC100M [42] that we managed to download. We use this publicly available dataset for research purposes only.

Method	Data	Classif. FC6	Classif. ALL	Detect. FC6	Detect. ALL
ImageNet labels	INet	89.3	89.2	66.3	70.3
Random	–	10.1	49.6	5.4	55.6
Unsupervised on curated data					
Larsson *et al.* [28]	INet+Pl.	–	77.2†	49.2	59.7
Wu *et al.* [48]	INet	–	–	–	60.5†
Doersh *et al.* [12]	INet	54.6	78.5	38.0	62.7
Caron *et al.* [6]	INet	78.5	82.5	58.7	65.9†
Unsupervised on non-curated data					
Mahendran *et al.* [31]	YFCCv	–	76.4†	–	–
Wang and Gupta [43]	YT8M	–	–	–	60.2†
Wang *et al.* [44]	YT9M	59.4	79.6	40.9	63.2†
DeeperCluster	YFCC	**79.7**	**84.3**	**60.5**	**67.8**

Table 2: Comparison of DeeperCluster to state-of-the-art unsupervised feature learning on classification and detection on PASCAL VOC 2007. We disassociate methods using curated datasets and methods using non-curated datasets. We selected hyper-parameters for each transfer task on the validation set, and then retrain on both training and validation sets. We report results on the test set averaged over 5 runs. "YFFCv" stands for the videos contained in YFFC100M dataset. † numbers from their original paper.

5. Experiments

In this section we evaluate the quality of the features learned with DeeperCluster on a variety of downstream tasks, such as classification or object detection. We also provide insights about the impact of the number of images and clusters on the performance of our model.

5.1. Evaluating unsupervised features

We evaluate the quality of the features extracted from a convnet trained with DeeperCluster on YFCC100M by considering several standard transfer learning tasks, namely image classification, object detection and scene classification.

Pascal VOC 2007 [14]. This dataset has small training and validation sets (2.5k images each), making it close to the setting of real applications where models trained using large computational resources are adapted to a new task with a small number of instances. We report numbers on the

classification and detection tasks with finetuning ("ALL") or by only retraining the last three fully connected layers of the network ("FC68"). The FC68 setting gives a better measure of the quality of the evaluated features since fewer parameters are retrained. For classification, we use the code of Caron *et al*. [6][1] and for detection, `fast-rcnn` [16][2]. For classification, we train the models for $150k$ iterations, starting with a learning rate of 0.002 decayed by a factor 10 every $20k$ iterations, and we report results averaged over 10 random crops. For object detection, we train our network for $150k$ iterations, dividing the step-size by 10 after the first $50k$ steps with an initial learning rate of 0.01 (FC68) or 0.002 (ALL) and a weight decay of 0.0001. Following Doersch *et al*. [12], we use the multiscale configuration, with scales $[400, 500, 600, 700]$ for training and $[400, 500, 600]$ for testing. In Table 2, we compare DeeperCluster with two sets of unsupervised methods that use a VGG-16 network: those trained on curated datasets and those trained on non-curated datasets. Previous unsupervised methods that worked on unucurated datasets with a VGG-16 use videos: Youtube8M ("YT8M"), Youtube9M ("YT9M") or the videos from YFCC100M ("YFFCv"). Our approach achieves state-of-the-art performance among all the unsupervised method that uses a VGG-16 architecture, even those that use ImageNet as a training set. The gap with a supervised network is still important when we freeze the convolutions (6% for detection and 10% for classification) but drops to less than 5% for both tasks with finetuning.

Linear classifiers on ImageNet [11] and Places205 [54]. ImageNet ("INet") and Places205 ("Pl.") are two large scale image classification datasets: ImageNet's domain covers objects and animals (1.3M images) and Places205's domain covers indoor and outdoor scenes (2.5M images). We train linear classifiers with a logistic loss on top of frozen convolutional layers at different depths. To reduce influence of feature dimension in the comparison, we average-pool the features until their dimension is below $10k$ [52]. This experiment probes the quality of the features extracted at each convolutional layer. In Figure 3, we observe that DeeperCluster matches the performance of a supervised network for all layers on Places205. On ImageNet, it also matches supervised features up to the 4th convolutional block; then the gap suddenly increases to around 20%. It is not surprising since the supervised features are trained on ImageNet itself, while ours are trained on YFCC100M.

5.2. Pre-training for ImageNet

In the previous section, we can observe that a VGG-16 trained on YFCC100M has similar or better low level features than the same network trained on ImageNet with supervision. In this experiment, we want to check whether these low-level features pre-trained on YFCC100M without supervision can serve as a good initialization for fully-supervised ImageNet classification. To this end, we pre-train a VGG-16 on YFCC100M using either DeeperCluster or RotNet. The resulting weights are then used as initialization for the training of a network on ImageNet with supervision. We merge the Sobel weights of the network pre-trained with DeeperCluster with the first convolutional layer during the initialization. We then train the networks on ImageNet with mini-batch SGD for 100 epochs, a learning rate of 0.1, a weight decay of 0.0001, a batch size of 256 and dropout of 0.5. We reduce the learning rate by a factor of 0.2 every 20 epochs. Note that this learning rate decay schedule slightly differs from the ImageNet classification PyTorch default implementation[3] where they train for 90 epochs and decay the learning rate by 0.1 at epochs 30 and 60. We give in Appendix the results with this default schedule (with unchanged conclusions). In Table 3, we compare the performance of a network trained with a standard intialization ("Supervised") to one initialized with a pre-training obtained from either DeeperCluster ("Supervised + DeeperCluster pre-training") or RotNet ("Supervised + RotNet pre-training") on YFCC100M. We see that our pre-training improves the performance of a supervised network by $+0.8\%$, leading to 74.9% top-1 accuracy. This means that our pre-training captures important statistics from YFCC100M that transfers well to ImageNet.

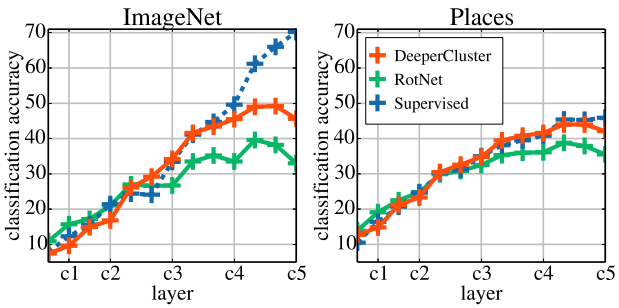

Figure 3: Accuracy of linear classifiers on ImageNet and Places205 using the activations from different layers as features. We compare a VGG-16 trained with supervision on ImageNet to VGG-16 trained with either RotNet or DeeperCluster on YFCC100M. Exact numbers are in Appendix.

5.3. Model analysis

In this final set of experiments, we analyze some components of our model. Since DeeperCluster derives from RotNet and DeepCluster, we first look at the difference between these methods and ours, when trained on curated and non-

[1] github.com/facebookresearch/deepcluster
[2] github.com/rbgirshick/py-faster-rcnn
[3] github.com/pytorch/examples/blob/master/imagenet/
[4] pytorch.org/docs/stable/torchvision/models

ImageNet	top-1	top-5
Supervised (PyTorch documentation[4])	73.4	91.5
Supervised (our code)	74.1	91.8
Supervised + RotNet pre-training	74.5	92.0
Supervised + DeeperCluster pre-training	**74.9**	**92.3**

Table 3: Accuracy on the validation set of ImageNet classification for a supervised VGG-16 trained with different initializations: we compare a network trained from a standard initialization to networks trained from pre-trained weights using either DeeperCluster or RotNet on YFCC100M.

Method	Data	ImageNet	Places	VOC2007
Supervised	ImageNet	70.2	45.9	84.8
Wu *et al*. [48]	ImageNet	39.2	36.3	-
RotNet	ImageNet	32.7	32.6	60.9
DeepCluster	ImageNet	**48.4**	37.9	71.9
RotNet	YFCC100M	33.0	35.5	62.2
DeepCluster	YFCC100M	34.1	35.4	63.9
DeeperCluster	YFCC100M	45.6	**42.1**	**73.0**

Table 4: Comparaison between DeeperCluster, RotNet and DeepCluster when pre-trained on curated and non-curated dataset. We report the accuracy on several datasets of a linear classifier trained on top of features of the last convolutional layer. All the methods use the same architecture. DeepCluster does not scale to the full YFCC100M dataset, we thus train it on a random subset of 1.3M images.

curated datasets. We then report quantitative and qualitative evaluations of the clusters obtained with DeeperCluster.

Comparison with RotNet and DeepCluster. In Table 4, we compare DeeperCluster with DeepCluster and RotNet when a linear classifier is trained on top of the last convolutional layer of a VGG-16 on several datasets. For reference, we also report previously published numbers [48] with a VGG-16 architecture. We average-pool the features of the last layer resulting in representations of 8192 dimensions. Our approach outperforms both RotNet and DeepCluster, even when they are trained on curated datasets (except for ImageNet classification task where DeepCluster trained on ImageNet yields the best performance). More interestingly, we see that the quality of the dataset or its scale has little impact on RotNet while it has on DeepCluster. This is confirming that self-supervised methods are more robust than clustering to a change of dataset distribution.

Influence of dataset size and number of clusters. To measure the influence of the number of images on features, we train models with 1M, 4M, 20M, and 96M images and report their accuracy on the validation set of the Pascal VOC 2007 classification task (FC68 setting). We also train models on 20M images with a number of clusters that varies from 10k to 160k. For the experiment with a total of 160k clusters, we choose $m = 2$ which results in 8 super-classes. In Figure 1, we observe that the quality of our features improves when scaling both in terms of images and clusters. Interestingly, between 4M and 20M of YFCC100M images are needed to meet the performance of our method on ImageNet. Augmenting the number of images has a bigger impact than the number of clusters. Yet, this improvement is significant since it corresponds to a reduction of more than 10% of the relative error w.r.t. the supervised model.

Quality of the clusters. In addition to features, our method provides a clustering of the input images. We evaluate the quality of these clusters by measuring their correlation with existing partitions of the data. In particular, YFCC100M comes with many different metadata. We consider hashtags, users, camera and GPS coordinates. If an image has several hashtags, we pick as label the least frequent one in the total hashtag distribution. We also measure the correlation of ours clusters with labels predicted by a classifier trained on ImageNet categories. We use a ResNet-50 network [21], pre-trained on ImageNet, to classify the YFCC100M images and we select those for which the confidence in prediction is higher than 75%. This evaluation omits a large amount of the data but gives some insight about the quality of our clustering in object classification.

In Figure 4, we show the evolution during training of the normalized mutual information (NMI) between our clustering and different metadata, and the predicted labels from ImageNet. The higher the NMI, the more correlated our clusters are to the considered partition. For reference, we compute the NMI for a clustering of RotNet features (as it corresponds to weights at initialization) and of a supervised model. First, it is interesting to observe that our clustering is improving over time for every type of metadata. One important factor is that most of these commodities are correlated since a given user takes pictures in specific places with probably a single camera and use a preferred fixed set of hashtags. Yet, these plots show that our model captures in the input signal enough information to predict these metadata at least as well as the features trained with supervision.

We visually assess the consistency of our clusters in Figure 5. We display 9 random images from 8 manually picked clusters. The first two clusters contain a majority of images associated with tag from the head (first cluster) and from the tail (second cluster) in the YFC100M dataset. Indeed, 418.538 YFC100M images are associated with the tag *cat*

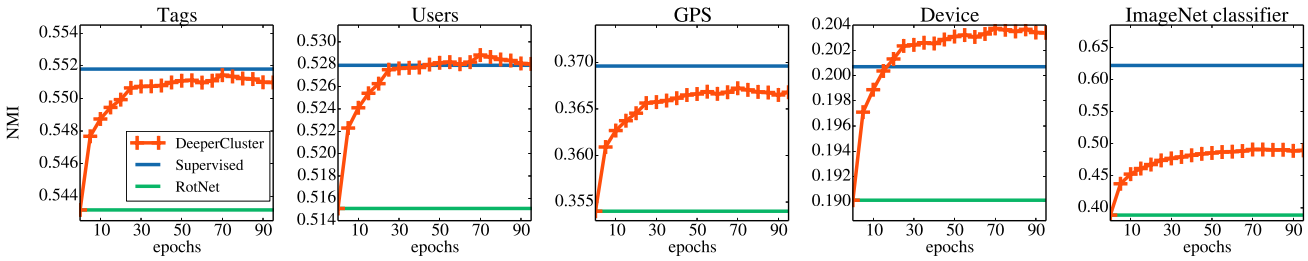

Figure 4: Normalized mutual information between our clustering and different sorts of metadata: hashtags, user IDs, geographic coordinates, and device types. We also plot the NMI with an ImageNet classifier labeling.

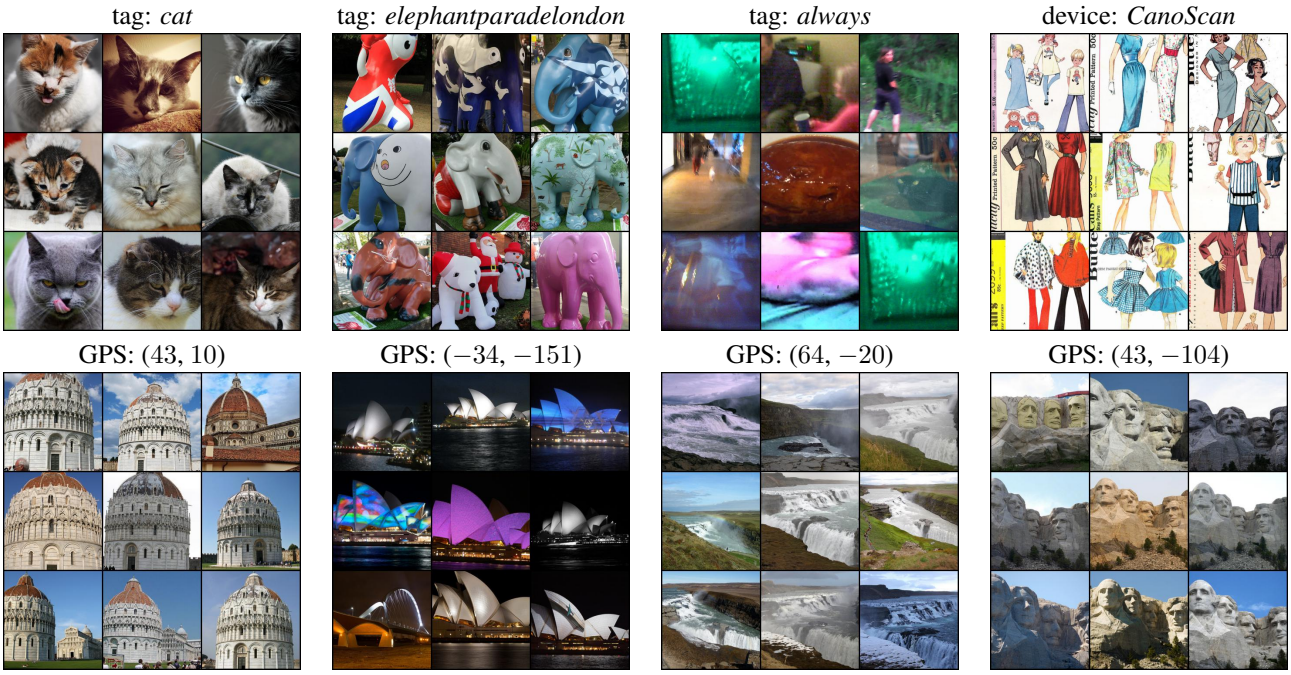

Figure 5: We randomly select 9 images per cluster and indicate the dominant cluster metadata. The bottom row depicts clusters pure for GPS coordinates but unpure for user IDs. As expected, they turn out to correlate with tourist landmarks. No metadata is used during training. For copyright reasons, we provide in Appendix the photographer username for each image.

whereas only 384 images contain the tag *elephantparadelondon* (0.0004% of the dataset). We also show a cluster for which the dominant hashtag does not corrolate visually with the content of the cluster. As already mentioned, this database is non-curated and contains images that basically do not depict anything semantic. The dominant metadata of the last cluster in the top row is the device ID *CanoScan*. As this cluster is about drawings, its images have been mainly taken with a scanner. Finally, the bottom row depict clusters that are pure for GPS coordinates but unpure for user IDs. It results in clusters of images taken by many different users in the same place: tourist landmarks.

6. Conclusion

In this paper, we present an unsupervised approach specifically designed to deal with large amount of non-curated data. Our method is well-suited for distributed training, which allows training on large datasets with $96M$ of images. With such amount of data, our approach surpasses unsupervised methods trained on curated datasets, which validates the potential of unsupervised learning in applications where annotations are scarce or curation is not trivial. Finally, we show that unsupervised pre-training improves the performance of a network trained on ImageNet.

Acknowledgement. Julien Mairal was funded by the ERC grant number 714381 (SOLARIS project).

References

[1] Pulkit Agrawal, Joao Carreira, and Jitendra Malik. Learning to see by moving. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2015. 2

[2] Miguel A Bautista, Artsiom Sanakoyeu, Ekaterina Tikhoncheva, and Bjorn Ommer. Cliquecnn: Deep unsupervised exemplar learning. In *Advances in Neural Information Processing Systems (NIPS)*, 2016. 2

[3] Piotr Bojanowski and Armand Joulin. Unsupervised learning by predicting noise. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2017. 2, 3, 5

[4] Léon Bottou. Stochastic gradient descent tricks. In *Neural networks: Tricks of the trade*, pages 421–436. Springer, 2012. 5

[5] Peter F Brown, Peter V Desouza, Robert L Mercer, Vincent J Della Pietra, and Jenifer C Lai. Class-based n-gram models of natural language. *Computational linguistics*, 18(4):467–479, 1992. 4

[6] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018. 1, 2, 3, 5, 6

[7] Joao Carreira, Pulkit Agrawal, Katerina Fragkiadaki, and Jitendra Malik. Human pose estimation with iterative error feedback. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 1

[8] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *arXiv preprint arXiv:1606.00915*, 2016. 1

[9] Xinlei Chen and Abhinav Gupta. Webly supervised learning of convolutional networks. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2015. 3

[10] Virginia R de Sa. Learning classification with unlabeled data. In *Advances in Neural Information Processing Systems (NIPS)*, 1994. 2

[11] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009. 1, 6

[12] Carl Doersch, Abhinav Gupta, and Alexei A Efros. Unsupervised visual representation learning by context prediction. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2015. 1, 2, 3, 5, 6

[13] Alexey Dosovitskiy, Philipp Fischer, Jost Tobias Springenberg, Martin Riedmiller, and Thomas Brox. Discriminative unsupervised feature learning with exemplar convolutional neural networks. *IEEE transactions on pattern analysis and machine intelligence*, 38(9):1734–1747, 2016. 2

[14] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010. 5

[15] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. In *International Conference on Learning Representations (ICLR)*, 2018. 1, 2, 3

[16] Ross Girshick. Fast r-cnn. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2015. 6

[17] Lluis Gomez, Yash Patel, Marçal Rusiñol, Dimosthenis Karatzas, and CV Jawahar. Self-supervised learning of visual features through embedding images into text topic spaces. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 3

[18] Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, large minibatch sgd: Training imagenet in 1 hour. *arXiv preprint arXiv:1706.02677*, 2017. 5

[19] Priya Goyal, Dhruv Mahajan, Abhinav Gupta, and Ishan Misra. Scaling and benchmarking self-supervised visual representation learning. *arXiv preprint arXiv:1905.01235*, 2019. 2

[20] Kaiming He, Ross Girshick, and Piotr Dollár. Rethinking imagenet pre-training. *arXiv preprint arXiv:1811.08883*, 2018. 1

[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 7

[22] Simon Jenni and Paolo Favaro. Self-supervised feature learning by learning to spot artifacts. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2

[23] Armand Joulin, Laurens van der Maaten, Allan Jabri, and Nicolas Vasilache. Learning visual features from large weakly supervised data. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016. 1, 3

[24] Dahun Kim, Donghyeon Cho, Donggeun Yoo, and In So Kweon. Learning image representations by completing damaged jigsaw puzzles. In *Winter Conference on Applications of Computer Vision (WACV)*, 2018. 2

[25] Alexander Kolesnikov, Xiaohua Zhai, and Lucas Beyer. Revisiting self-supervised visual representation learning. *arXiv preprint arXiv:1901.09005*, 2019. 2

[26] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2012. 5

[27] Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. Learning representations for automatic colorization. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016. 2

[28] Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. Colorization as a proxy task for visual understanding. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2, 5

[29] Renjie Liao, Alex Schwing, Richard Zemel, and Raquel Urtasun. Learning deep parsimonious representations. In *Advances in Neural Information Processing Systems (NIPS)*, 2016. 2

[30] Dhruv Mahajan, Ross Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens van der Maaten. Exploring the limits of weakly supervised pretraining. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018. 1, 3

[31] Aravindh Mahendran, James Thewlis, and Andrea Vedaldi. Cross pixel optical flow similarity for self-supervised learning. *arXiv preprint arXiv:1807.05636*, 2018. 2, 5

[32] Karl Ni, Roger Pearce, Kofi Boakye, Brian Van Essen, Damian Borth, Barry Chen, and Eric Wang. Large-scale deep learning on the yfcc100m dataset. *arXiv preprint arXiv:1502.03409*, 2015. 3

[33] Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016. 2, 3, 4

[34] Mehdi Noroozi, Ananth Vinjimoor, Paolo Favaro, and Hamed Pirsiavash. Boosting self-supervised learning via knowledge transfer. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2

[35] Deepak Pathak, Ross Girshick, Piotr Dollár, Trevor Darrell, and Bharath Hariharan. Learning features by watching objects move. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2

[36] Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A Efros. Context encoders: Feature learning by inpainting. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 2, 3

[37] Mattis Paulin, Matthijs Douze, Zaid Harchaoui, Julien Mairal, Florent Perronin, and Cordelia Schmid. Local convolutional features with unsupervised training for image retrieval. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2015. 5

[38] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2015. 1

[39] Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. Cnn features off-the-shelf: an astounding baseline for recognition. In *CVPR workshops*, 2014. 1

[40] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 5

[41] Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Revisiting unreasonable effectiveness of data in deep learning era. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 843–852, 2017. 1, 3

[42] Bart Thomee, David A Shamma, Gerald Friedland, Benjamin Elizalde, Karl Ni, Douglas Poland, Damian Borth, and Li-Jia Li. Yfcc100m: The new data in multimedia research. *arXiv preprint arXiv:1503.01817*, 2015. 1, 5

[43] Xiaolong Wang and Abhinav Gupta. Unsupervised learning of visual representations using videos. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2015. 2, 4, 5

[44] Xiaolong Wang, Kaiming He, and Abhinav Gupta. Transitive invariance for self-supervised visual representation learning. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2017. 2, 5

[45] Xiaosong Wang, Le Lu, Hoo-Chang Shin, Lauren Kim, Mohammadhadi Bagheri, Isabella Nogues, Jianhua Yao, and Ronald M Summers. Unsupervised joint mining of deep features and image labels for large-scale radiology image categorization and scene recognition. In *Winter Conference on Applications of Computer Vision (WACV)*, 2017. 2

[46] Philippe Weinzaepfel, Jerome Revaud, Zaid Harchaoui, and Cordelia Schmid. Deepflow: Large displacement optical flow with deep matching. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2013. 1

[47] Tobias Weyand, Ilya Kostrikov, and James Philbin. Planet-photo geolocation with convolutional neural networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016. 3

[48] Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2, 5, 7

[49] Junyuan Xie, Ross Girshick, and Ali Farhadi. Unsupervised deep embedding for clustering analysis. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2016. 2, 3

[50] Jianwei Yang, Devi Parikh, and Dhruv Batra. Joint unsupervised learning of deep representations and image clusters. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 2, 3

[51] Liheng Zhang, Guo-Jun Qi, Liqiang Wang, and Jiebo Luo. Aet vs. aed: Unsupervised representation learning by auto-encoding transformations rather than data. *arXiv preprint arXiv:1901.04596*, 2019. 1, 3, 4

[52] Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016. 2, 3, 6

[53] Richard Zhang, Phillip Isola, and Alexei A Efros. Split-brain autoencoders: Unsupervised learning by cross-channel prediction. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2

[54] Bolei Zhou, Agata Lapedriza, Jianxiong Xiao, Antonio Torralba, and Aude Oliva. Learning deep features for scene recognition using places database. In *Advances in Neural Information Processing Systems (NIPS)*, 2014. 6