Learning To Classify Images Without Labels

Wouter Van Gansbeke[1⋆] Simon Vandenhende[1⋆] Stamatios Georgoulis[2]
Marc Proesmans[1] Luc Van Gool[1,2]

[1]KU Leuven/ESAT-PSI [2]ETH Zurich/CVL, TRACE

Abstract. Is it possible to automatically classify images without the use of ground-truth annotations? Or when even the classes themselves, are not a priori known? These remain important, and open questions in computer vision. Several approaches have tried to tackle this problem in an end-to-end fashion. In this paper, we deviate from recent works, and advocate a two-step approach where feature learning and clustering are decoupled. First, a self-supervised task from representation learning is employed to obtain semantically meaningful features. Second, we use the obtained features as a prior in a learnable clustering approach. In doing so, we remove the ability for cluster learning to depend on low-level features, which is present in current end-to-end learning approaches. Experimental evaluation shows that we outperform state-of-the-art methods by huge margins, in particular +26.9% on CIFAR10, +21.5% on CIFAR100-20 and +11.7% on STL10 in terms of classification accuracy. Furthermore, results on ImageNet show that our approach is the first to scale well up to 200 randomly selected classes, obtaining 69.3% top-1 and 85.5% top-5 accuracy, and marking a difference of less than 7.5% with fully-supervised methods. Finally, we applied our approach to all 1000 classes on ImageNet, and found the results to be very encouraging. The code is made publicly available **here**.

Keywords: Unsupervised Learning, Image Classification, Clustering.

1 Introduction and prior work

Image classification is the task of assigning a semantic label from a predefined set of classes to an image. For example, an image depicts a cat, a dog, a car, an airplane, etc., or abstracting further an animal, a machine, etc. Nowadays, this task is typically tackled by training convolutional neural networks [28,44,19,51,46] on large-scale datasets [9,30] that contain annotated images, i.e. images with their corresponding semantic label. Under this supervised setup, the networks excel at learning discriminative feature representations that can subsequently be clustered into the predetermined classes. What happens, however, when there is no access to ground-truth semantic labels at training time? Or going further, the semantic classes, or even their total number, are not *a priori* known? The desired goal in this case is to group the images into clusters, such that images within

⋆ Authors contributed equally

the same cluster belong to the same or similar semantic classes, while images in different clusters are semantically dissimilar. Under this setup, unsupervised or self-supervised learning techniques have recently emerged in the literature as an alternative to supervised feature learning.

Representation learning methods [11,39,55,35,15] use self-supervised learning to generate feature representations solely from the images, omitting the need for costly semantic annotations. To achieve this, they use pre-designed tasks, called pretext tasks, which do not require annotated data to learn the weights of a convolutional neural network. Instead, the visual features are learned by minimizing the objective function of the pretext task. Numerous pretext tasks have been explored in the literature, including predicting the patch context [11,33], inpainting patches [39], solving jigsaw puzzles [35,37], colorizing images [55,29], using adversarial training [12,13], predicting noise [3], counting [36], predicting rotations [15], spotting artifacts [23], generating images [41], using predictive coding [38,20], performing instance discrimination [49,18,14,32], and so on. Despite these efforts, representation learning approaches are mainly used as the first pretraining stage of a two-stage pipeline. The second stage includes fine-tuning the network in a fully-supervised fashion on another task, with as end goal to verify how well the self-supervised features transfer to the new task. When annotations are missing, as is the case in this work, a clustering criterion (e.g. K-means) still needs to be defined and optimized independently. This practice is arguably suboptimal, as it leads to imbalanced clusters [4], and there is no guarantee that the learned clusters will align with the semantic classes.

As an alternative, *end-to-end learning* pipelines combine feature learning with clustering. A first group of methods (e.g. DEC [50], DAC [6], DeepCluster [4], DeeperCluster [5], or others [1,17,52]) leverage the architecture of CNNs as a prior to cluster images. Starting from the initial feature representations, the clusters are iteratively refined by deriving the supervisory signal from the most confident samples [6,50], or through cluster re-assignments calculated offline [4,5]. A second group of methods (e.g. IIC [24], IMSAT [21]) propose to learn a clustering function by maximizing the mutual information between an image and its augmentations. In general, methods that rely on the initial feature representations of the network are sensitive to initialization [6,50,4,5,22,17,52], or prone to degenerate solutions [4,5], thus requiring special mechanisms (like pretraining, cluster reassignment and feature cleaning) to avoid those situations. Most importantly, since the cluster learning depends on the network initialization, it is likely to latch onto low-level features, such as color, which is unwanted for the objective of semantic clustering [16]. To prevent the network from focusing on the low-level information, certain approaches [24,21] are tied to the use of specific preprocessing (e.g. Sobel filtering).

In this work we advocate for the use of a two-step approach for unsupervised classification, in contrast to recent end-to-end learning approaches. The proposed method leverages the advantages of both representation and end-to-end learning approaches, but at the same time it addresses their shortcomings:

- In a first step, we learn feature representations through a pretext task. In contrast to representation learning approaches that require K-means clustering after learning the feature representations, with no guarantee that the latter are semantically meaningful, we propose to mine the nearest neighbors of each image based on feature similarity. We empirically found that in most cases these nearest neighbors belong to the same semantic class (see Figure 2), rendering them appropriate for semantic clustering.
- In a second step, we integrate the semantically meaningful nearest neighbors as a prior into a learnable approach. We classify each image and its mined neighbors together by using a loss function that maximizes their dot product after softmax, pushing the network to produce both consistent and discriminative (one-hot) predictions. Unlike end-to-end approaches, the learned clusters depend on more meaningful features, rather than on the network architecture. This avoids the clustering from latching onto low-level features such as color at the beginning of training. Furthermore, because we encourage invariance w.r.t. the nearest neighbors, and not solely w.r.t. augmentations, we found no need to apply specific preprocessing to the input.

Experimental evaluation shows that our method outperforms prior work by huge margins across multiple datasets. Furthermore, we obtain promising results on the large-scale ImageNet dataset, without the use of ground-truth. This validates our assumption that separation between learning (semantically meaningful) features and clustering them is an arguably better approach over recent end-to-end works.

2 Method

The following sections present the cornerstones of our approach. First, we show how mining nearest neighbors from a pretext task can be used as a prior for semantic clustering. Also, we introduce additional constraints for selecting an appropriate pretext task, capable of producing semantically meaningful feature representations. Second, we integrate the obtained prior into a novel loss function to classify each image and its nearest neighbors together. Additionally, we mitigate the problem of noise inherent in the nearest neighbor selection with a self-labeling approach. We believe that each of these contributions are relevant for unsupervised image classification.

2.1 Representation learning for semantic clustering

In the supervised learning setup, each sample can be associated with its correct cluster by using the available ground-truth labels. In particular, the mapping between the images $\mathcal{D} = \{X_1, \ldots, X_{|\mathcal{D}|}\}$ and the semantic classes $\mathcal{C}$ can generally be learned by minimizing a cross-entropy loss. However, when we do not have access to such ground-truth labels, we need to define a prior to obtain an estimate of which samples are likely to belong together, and which are not.

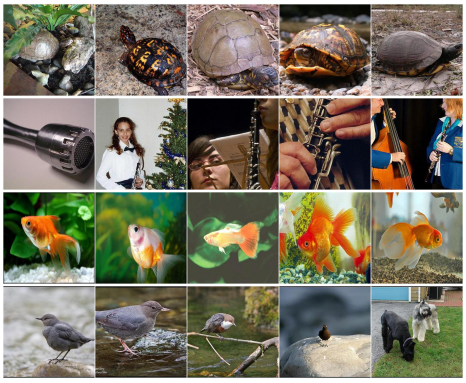


Fig. 1: Images (first column) and their nearest neighbors (other columns) sampled under pretext task [49].

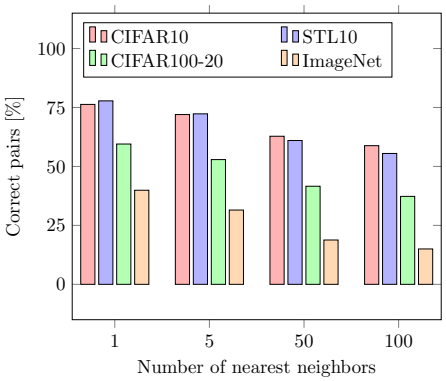


Fig. 2: Neighboring samples tend to be instances of the same semantic class.

End-to-end learning approaches have utilized the architecture of CNNs as a prior [52,6,50,17,4,5], or enforced consistency between images and their augmentations [24,21] to disentangle the clusters. In both cases, the cluster learning is known to be sensitive to the network initialization. Furthermore, at the beginning of training the network does not extract high-level information from the image yet. As a result, the clusters can easily latch onto low-level features (e.g. color, texture, contrast, etc.), which is suboptimal for semantic clustering. To overcome these limitations of end-to-end learning approaches, we employ representation learning as a means to obtain a better prior for semantic clustering.

In representation learning, a pretext task τ learns in a self-supervised fashion an embedding function Φ_θ - parameterized by a neural network with weights θ - that maps images into feature representations. The literature offers several pretext tasks which can be used to learn such an embedding function Φ_θ (e.g. rotation prediction [15], affine or perspective transformation prediction [54], colorization [29], in-painting [39], instance discrimination [49,14,32], etc.). In practice, however, certain pretext tasks are based on specific image transformations, causing the learned feature representations to be covariant to the employed transformation. For example, when Φ_θ predicts the transformation parameters of an affine transformation, different affine transformations of the same image will result in distinct output predictions for Φ_θ. This renders the learned feature representations less appropriate for semantic clustering, where feature representations ought to be invariant to image transformations. To overcome this issue, we impose the pretext task τ to also minimize the distance between images X_i and their augmentations $T[X_i]$, which can be expressed as:

$$\min_{\theta} d(\Phi_\theta(X_i), \Phi_\theta(T[X_i])). \tag{1}$$

Any pretext task [49,32,14] that satisfies Equation 1 can consequently be used. For example, Figure 1 shows the results when retrieving the nearest neighbors

under an instance discrimination task [49] which satisfies Equation 1. We observe that similar features are assigned to semantically similar images. An experimental evaluation using different pretext tasks can be found in Section 3.2.

To understand why images with similar high-level features are mapped closer together by Φ_θ, we make the following observations. First, the pretext task output is conditioned on the image, forcing Φ_θ to extract specific information from its input. Second, because Φ_θ has a limited capacity, it has to discard low-level features which are not predictive of the high-level pretext task. For example, it is unlikely that Φ_θ can solve an instance discrimination task by only encoding color or a single pixel from the input image. As a result, images with similar high-level characteristics will lie closer together in the embedding space of Φ_θ.

We conclude that pretext tasks from representation learning can be used to obtain semantically meaningful features. Following this observation, we will use the features from a pretext task as a prior for clustering the images.

2.2 A semantic clustering loss

Mining nearest neighbors. In Section 2.1, we motivated that a pretext task from representation learning can be used to obtain semantically meaningful features. However, naively applying K-means on the obtained features can lead to cluster degeneracy [4]. A discriminative model can assign all its probability mass to the same cluster when learning the decision boundary. This leads to one cluster dominating the others. Instead, we opt for a better strategy.

Let us first consider the following experiment. Through representation learning, we train a model Φ_θ on the unlabeled dataset $\mathcal{D}$ to solve a pretext task τ, i.e. instance discrimination [49]. Then, for every sample $X_i \in \mathcal{D}$, we mine its K nearest neighbors in the embedding space Φ_θ. We define the set $\mathcal{N}_{X_i}$ as the neighboring samples of X_i in the dataset $\mathcal{D}$. Figure 2 quantifies the degree to which the mined nearest neighbors are instances of the same semantic cluster. We observe that this is largely the case across four datasets[1] (CIFAR10 [27], CIFAR100-20 [27], STL10 [7] and ImageNet [9]) for different values of K. Motivated by this observation, we propose to adopt the nearest neighbors obtained through representation learning as our prior for semantic clustering.

Loss function. We aim to learn a clustering function Φ_η - parameterized by a neural network with weights η - that classifies a sample X_i and its mined neighbors $\mathcal{N}_{X_i}$ together. The function Φ_η performs a soft assignment over the clusters $\mathcal{C} = \{1, \ldots, C\}$, with $\Phi_\eta(X_i) \in [0,1]^C$. The probability of sample X_i being assigned to cluster c is denoted as $\Phi_\eta^c(X_i)$. We learn the weights of Φ_η by minimizing the following objective:

$$\begin{aligned} \Lambda = -\frac{1}{|\mathcal{D}|} \sum_{X \in \mathcal{D}} \sum_{k \in \mathcal{N}_X} \log \langle \Phi_\eta(X), \Phi_\eta(k) \rangle + \lambda \sum_{c \in \mathcal{C}} \Phi_\eta'^c \log \Phi_\eta'^c, \\ \text{with } \Phi_\eta'^c = \frac{1}{|\mathcal{D}|} \sum_{X \in \mathcal{D}} \Phi_\eta^c(X). \end{aligned} \tag{2}$$

[1] The details for each dataset will be provided in the supplementary materials.

Here, $\langle \cdot \rangle$ denotes the dot product operator. The first term in Equation 2 imposes Φ_η to make consistent predictions for a sample X_i and its neighboring samples $\mathcal{N}_{X_i}$. Note that, the dot product will be maximal when the predictions are one-hot (confident) and assigned to the same cluster (consistent). To avoid Φ_η from assigning all samples to a single cluster, we include an entropy term (the second term in Equation 2), which spreads the predictions uniformly across the clusters $\mathcal{C}$. If the probability distribution over the clusters $\mathcal{C}$ is known in advance, which is not the case here, this term can be replaced by KL-divergence.

Remember that, the exact number of clusters in $\mathcal{C}$ is generally unknown. However, similar to prior work [50,6,24], we choose C equal to the number of ground-truth clusters for the purpose of evaluation. In practice, it should be possible to obtain a rough estimate of the amount of clusters[2]. Based on this estimate, we can overcluster to a larger amount of clusters, and enforce the class distribution to be uniform. We refer to Section 3.4 for a concrete experiment.

Implementation details. For the practical implementation of our loss function, we approximate the dataset statistics by sampling batches of sufficiently large size. During training we randomly augment the samples X_i and their neighbors $\mathcal{N}_{X_i}$. For the corner case $K = 0$, only consistency between samples and their augmentations is imposed. We set $K \geq 1$ to capture more of the cluster's variance, at the cost of introducing noise, i.e. not all samples and their neighbors belong to the same cluster. Section 3.2 experimentally shows that choosing $K \geq 1$ significantly improves the results compared to only enforcing consistency between samples and their augmentations, as in [24,21].

Discussion. In contrast to [40,25,47,2,34,56,50] we do not include a reconstruction criterion into the loss, since this is not explicitly required by our target task. After all, we are only interested in a few bits of information encoded from the input signal, rather than the majority of information that a reconstruction criterion typically requires. It is worth noting that the consistency in our case is enforced at the level of individual samples through the dot product term in the loss, rather than on an approximation of the joint distribution over the classes [24,21]. We argue that this choice allows to express the consistency in a more direct way. Finally, we do not directly cluster the features learned from the pretext task through K-means [50,4,17], which is known to give degenerate results [24].

2.3 Fine-tuning through self-labeling

The semantic clustering loss in Section 2.2 imposed consistency between a sample and its neighbors. More specifically, each sample was combined with $K \geq 1$ neighbors, some of which inevitably do not belong to the same semantic cluster. These false positive examples lead to predictions, for which the network is less certain. At the same time, we experimentally observed that samples with highly confident predictions ($p_{max} \approx 1$) tend to be classified to the proper cluster. In

[2] As an example, say you want to cluster various animal species observed in a national park. In this case, we can rely on prior domain knowledge to make an estimate.

Algorithm 1 Semantic Clustering by Adopting Nearest neighbors (SCAN)

Input: Dataset $\mathcal{D}$, Clusters $\mathcal{C}$, Task τ, Neural Nets Φ_θ and Φ_η, Neighbors $\mathcal{N}_\mathcal{D} = \{\}$.
Optimize Φ_θ with task τ. ▷ Pretext Task Step, Sec. 2.1
for $X_i \in \mathcal{D}$ **do**
    $\mathcal{N}_\mathcal{D} \leftarrow \mathcal{N}_\mathcal{D} \cup \mathcal{N}_{X_i}$, with $\mathcal{N}_{X_i} = K$ neighboring samples of $\Phi_\theta(X_i)$.
end for
while SCAN-loss decreases **do** ▷ Clustering Step, Sec. 2.2
    Update Φ_η with SCAN-loss, i.e. $\Lambda(\Phi_\eta(\mathcal{D}), \mathcal{N}_\mathcal{D}, C)$ in Eq. 2
end while
while $Len(Y)$ increases **do** ▷ Self-Labeling Step, Sec. 2.3
    Y $\leftarrow$ ($\Phi_\eta(\mathcal{D}) >$ threshold)
    Update Φ_η with cross-entropy loss, i.e. $H(\Phi_\eta(\mathcal{D}), Y)$
end while
Return: $\Phi_\eta(\mathcal{D})$ ▷ $\mathcal{D}$ is divided over C clusters

fact, the highly confident predictions that the network forms during clustering can be regarded as "prototypes" for each class (see Section 3.5). Unlike prior work [6,4,50], this allows us to select samples based on the confidence of the predictions in a more reliable manner. Hence, we propose a self-labeling approach [43,31] to exploit the already well-classified examples, and correct for mistakes due to noisy nearest neighbors.

In particular, during training confident samples are selected by thresholding the probability at the output, i.e. $p_{max} >$ threshold. For every confident sample, a pseudo label is obtained by assigning the sample to its predicted cluster. A cross-entropy loss is used to update the weights for the obtained pseudo labels. To avoid overfitting, we calculate the cross-entropy loss on strongly augmented versions of the confident samples. This self-labeling step allows the network to correct itself, as it gradually becomes more certain, adding more samples to the mix. We refer to Section 3.2 for a concrete experiment.

Algorithm 1 summarizes all the steps of the proposed method. We further refer to our approach as SCAN, i.e. Semantic Clustering by Adopting Nearest neighbors.

3 Experiments

3.1 Experimental setup

Datasets. Our experimental evaluation is performed on CIFAR10 [27], CIFAR100-20 [27], STL10 [7] and ImageNet [9]. We focus on the smaller datasets first. The results on ImageNet are discussed separately in Section 3.5. Following prior work [24,6,50,52], we train and evaluate on the full dataset. The results are reported as the mean and standard deviation on 10 separate runs. All experiments are performed using the same backbone, augmentations, pretext task and hyperparameters.

Training setup. We use a standard ResNet-18 backbone. For every sample, the 5 nearest neighbors are determined through an instance discrimination task based on noise contrastive estimation (NCE) [49][3]. The selected pretext task satisfies the feature invariance constraint from Equation 1, i.e. every image is classified as a separate instance independent of the used transformation. As already mentioned, during training the consistency is also imposed between images and their random augmentations. To speed up training, we transfer the weights, obtained from the pretext task, to initiate the clustering step (Section 2.2). We perform the clustering step for 100 epochs using batches of size 128. The weight on the entropy term is set to $\lambda = 2$. A higher weight avoids the premature grouping of samples early on during training. The results seem to be insensitive to small changes of λ. After the clustering step, we train for another 100 epochs using the self-labeling procedure with threshold 0.99 (Section 2.3). We use a weighted cross-entropy loss to compensate for the imbalance between confident samples across clusters. The weights are inversely proportional to the number of occurrences in the batch after thresholding. We use an Adam optimizer [25] with initial learning rate 10^{-4} and weight decay 10^{-4}. The standard data augmentations are random flips, random crops and jitter. The strong augmentations are composed of four randomly selected transformations from AutoAugment [8]. The transformation parameters are uniformly sampled between fixed intervals. For more details we refer the interested reader to the supplementary materials.
Validation criterion During the clustering step, we select the best model based on the lowest loss. During the self-labeling step, we save the weights of the model when the amount of confident samples plateaus. We follow these practices as we do not have access to a labeled validation set.

3.2 Ablation studies

Method. We quantify the performance gains w.r.t. the various parts of our method through an ablation study on CIFAR10 in Table 1. K-means clustering of the NCE pretext features results in low accuracy (35.9%). This is to be expected since the cluster assignments are imbalanced (Figure 3), and not guaranteed to align well with the actual semantic classes. We consider the effect of the nearest neighbors consistency by replacing the dot product loss by a sample-wise entropy loss, which is minimized to encourage the predictions to be one-hot. The latter can only rely on the prior imposed by the network architecture to assign each sample to its correct cluster. We conclude that the SCAN-loss (see Equation 2) gives better results compared to relying on the backbone itself to group the right samples together (62.7% vs. 19.2%).

Applying strong augmentations to the samples and their nearest neighbors further improves the results (62.7% vs. 72.5%). The strong augmentations shrink the solution space by imposing additional invariances, thereby boosting generalization. After the first epoch, we observe that the model makes more confident

[3] Note that, some recent works [18,32] have built and improved upon this method. As a consequence, implementing the pretext task following any of these is also expected to boost our results accordingly.

Table 1: Ablation Method on CIFAR10

Setup	**ACC** (Avg ± Std)
Pretext + K-means	35.9 ± 3.6
Sample + Batch Entropy Loss	19.2 ± 0.9
SCAN-Loss (Standard Augs) (Ours)	62.7 ± 3.3
SCAN-Loss (Strong Augs) (Ours)	72.5 ± 3.5
SCAN-Loss + Self-Labeling (Ours)	83.5 ± 4.1

Table 2: Ablation Pretext Task on CIFAR10

Pretext Task	**ACC** (Avg ± Std)
RotNet [15]	74.3 ± 3.9
Feature Decoupling [14]	83.0 ± 3.4
NCE [49]	83.5 ± 4.1

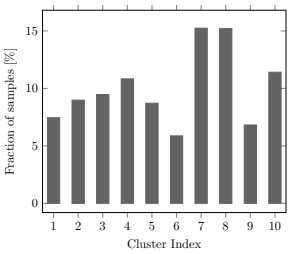


Fig. 3: K-means cluster assignments are imbalanced.

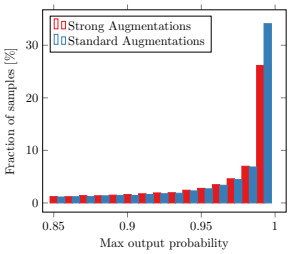


Fig. 4: Output probabilities after one epoch during the clustering step.

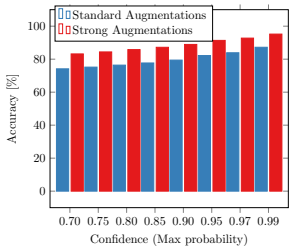


Fig. 5: Accuracy vs. confidence after the complete clustering step.

predictions when using standard augmentations (see Figure 4), while more classification errors are made (see Figure 5) after completing the clustering step. On the one hand, strong augmentations make the network less confident, but on the other hand, this helps the network to avoid learning degenerate features which can reduce the loss early on during training. The latter can be beneficial for avoiding such local optima, since the SCAN-loss is a non-convex function (see Equation 2).

Fine-tuning the network weights through self-labeling further improves the results (72.51% to 83.50%). Figure 5 confirms our assumption that confident samples are better classified after the clustering step. The self-labeling step allows the network to correct itself as it gradually becomes more certain (see Figure 6).

Pretext task. In addition to the instance discrimination task (NCE [49]) from before, we consider RotNet [15] and Feature Decoupling [14] as possible pretext tasks to mine the nearest neighbors. RotNet is trained to predict image rotations, while Feature Decoupling jointly tackles instance discrimination and rotation prediction. Since RotNet learns to discriminate between multiple orientations, the distance between an image X_i and its augmentations $T[X_i]$ is not minimized in the embedding space (Equation 1). Differently, the instance discrimination task in NCE and Feature Decoupling satisfies the invariance criterion from Equation 1. Table 2 shows the results on CIFAR10.

A first observation is that the proposed method is not tied to a specific pretext task. All cases report high accuracy (+70%). Second, however, pretext

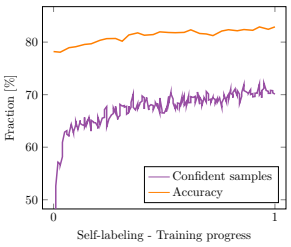


Fig. 6: Accuracy and the number of confident samples increase during self-labeling.

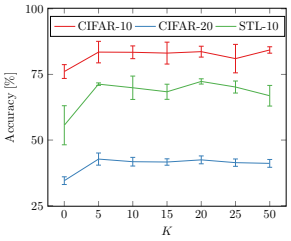


Fig. 7: Influence of the used number of neighbors K.

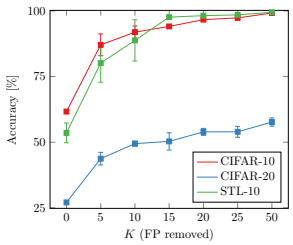


Fig. 8: Results without false positives in the nearest neighbors.

tasks which satisfy the invariance criterion are better suited to mine the nearest neighbors, i.e. 83.0% and 83.5% for Feature Decoupling and NCE vs. 74.3% for RotNet. This confirms our hypothesis from Section 2.1, that it is beneficial to select a pretext task which imposes invariance between images and their augmentations.

Number of neighbors. Figure 7 shows the influence of using a different number of nearest neighbors K. The results are not very sensitive to the value of K, and even remain stable when increasing K to 50. This is beneficial, since we do not have to fine-tune the value of K on every new dataset. We also consider the corner case $K = 0$, i.e. we only enforce consistent predictions for images and their augmentations. The performance decreases on all three datasets compared to $K = 5$, 76.2% vs 83.5% on CIFAR10, 34.5% vs 42.8% on CIFAR100-20 and 55.6% vs 70.2% on STL10. This confirms our initial assumption that better representations can be learned by enforcing coherent predictions between a sample and its nearest neighbors.

Convergence. Figure 8 shows the results when removing the false positives from the nearest neighbors, i.e. sample-pairs which belong to a different class. The results can be considered as an upper-bound for the proposed method in terms of classification accuracy. A desirable characteristic is that the clusters quickly align with the ground truth, obtaining near perfect results on CIFAR10 and STL10 with a relatively small increase in the number of used neighbors K. The lower performance improvement on CIFAR100-20 can be explained by the ambiguity of the superclasses used to measure the accuracy. For example, there is not exactly one way to group categories like omnivores or carnivores together.

3.3 Comparison with the state-of-the-art

Comparison. Table 3 compares our method to the state-of-the-art on three different benchmarks. We evaluate the results based on clustering accuracy (ACC),

Table 3: State-of-the-art comparison: We report average results after 100 epochs (*), and the results of the best model after 100 (†) and 1000 (‡) epochs, following the validation criterion from Section 3.1.

Dataset	**CIFAR10**			**CIFAR100-20**			**STL10**		
Metric	ACC	NMI	ARI	ACC	NMI	ARI	ACC	NMI	ARI
K-means [48]	22.9	8.7	4.9	13.0	8.4	2.8	19.2	12.5	6.1
SC [53]	24.7	10.3	8.5	13.6	9.0	2.2	15.9	9.8	4.8
Triplets [42]	20.5	–	–	9.94	–	–	24.4	–	–
JULE [52]	27.2	19.2	13.8	13.7	10.3	3.3	27.7	18.2	16.4
AEVB [26]	29.1	24.5	16.8	15.2	10.8	4.0	28.2	20.0	14.6
SAE [34]	29.7	24.7	15.6	15.7	10.9	4.4	32.0	25.2	16.1
DAE [47]	29.7	25.1	16.3	15.1	11.1	4.6	30.2	22.4	15.2
SWWAE [56]	28.4	23.3	16.4	14.7	10.3	3.9	27.0	19.6	13.6
AE [2]	31.4	23.4	16.9	16.5	10.0	4.7	30.3	25.0	16.1
GAN [40]	31.5	26.5	17.6	15.1	12.0	4.5	29.8	21.0	13.9
DEC [50]	30.1	25.7	16.1	18.5	13.6	5.0	35.9	27.6	18.6
ADC [17]	32.5	–	–	16.0	–	–	53.0	–	–
DeepCluster [4]	37.4	–	–	18.9	–	–	33.4	–	–
DAC [6]	52.2	40.0	30.1	23.8	18.5	8.8	47.0	36.6	25.6
IIC [24]	**61.7**	**51.1**	**41.1**	**25.7**	**22.5**	**11.7**	**59.6**	**49.6**	**39.7**
SCAN* (Ours)(Avg)	83.5 ± 4.1	73.1 ± 2.5	69.7 ± 3.9	42.8 ± 2.3	42.5 ± 1.1	26.3 ± 1.3	70.2 ± 1.3	63.2 ± 0.9	54.8 ± 1.6
SCAN† (Ours)	85.2	74.3	71.5	44.9	43.5	27.5	71.3	64.9	57.4
SCAN‡ (Ours)	**88.6**	**80.3**	**77.9**	**47.2**	**45.5**	**30.3**	**71.3**	**64.9**	**57.4**
Absolute [%]	**+26.9**	**+29.2**	**+36.8**	**+21.5**	**+23.0**	**+18.6**	**+11.7**	**+15.3**	**+17.7**
Relative [%]	+43.6	+57.1	+89.5	+83.7	+102.2	+159.0	+19.6.8	+30.8	+44.6
SCAN† (Overcluster)	**83.8**	**73.9**	**68.9**	**52.5**	**45.8**	**31.6**	**75.0**	**63.8**	**57.2**

normalized mutual information (NMI) and adjusted rand index (ARI). The proposed method consistently outperforms prior work by large margins on all three metrics, e.g. +26.9% on CIFAR10, +21.5% on CIFAR100-20 and +11.7% on STL10 in terms of accuracy. We do not provide further comparison with methods from representation learning (e.g. [18,32]) as they require supervised fine-tuning, or the application of K-means which was already shown to be suboptimal in Section 3.2.

Other advantages. In contrast to prior work [6,24,21], we did not perform any dataset specific fine-tuning. Furthermore, the results on CIFAR10 were obtained within 3 hours on a single GPU, including the pretext task training, the clustering step and the self-labeling. As a comparison, prior SOTA requires at least a day of train time.

3.4 Overclustering

So far we assumed to have knowledge about the number of ground-truth classes. The model predictions were evaluated using a hungarian matching algorithm. However, what happens if the number of clusters does not match the number of ground-truth classes anymore. Table 3 reports the results when we overestimate the number of ground-truth classes by a factor of 2, e.g. we cluster CIFAR10 into 20 rather than 10 classes. The classification accuracy remains stable for CIFAR10 (85.2% to 83.8%), and improves for CIFAR100-20 (44.9% to 52.5%)

and STL10 (71.3% to 75.0%)[4]. We conclude that the approach does not require knowledge of the exact amount of clusters. Furthermore, we hypothesize that the increased performance on STL10 and CIFAR100-20 is related to the higher intra-class variance. More specifically, STL10 contains higher resolution images, while CIFAR100-20 groups multiple object categories together in superclasses. In this case, an overclustering approach might be better suited to explain the intra-class variance.

3.5 ImageNet

Setup. We are the first to consider the problem of unsupervised classification on the large-scale ImageNet [9] dataset[5]. Instead of directly training on the full dataset, we first consider smaller subsets of 50, 100 and 200 randomly selected classes. The sets of 50 and 100 classes are subsets of the 100 and 200 classes respectively. We repeat the experiment five times, each time using a different set of randomly selected classes. The selected classes are added to the supplementary materials, so future research can use them as a benchmark. We reuse the training setup from before (Section 3.1). The remainder of this section discusses the most important results on ImageNet. Additional experiments can be found in the supplementary materials. The results are reported on the held-out validation set, and we encourage future works to do the same.

Quantitative evaluation. We select the first subset of 50, 100 and 200 ImageNet classes to evaluate our approach. Table 4 compares our results against K-means, IIC [24] (prior SOTA) and supervised learning. Again, we outperform other unsupervised methods by large margins on all test metrics. Furthermore, we reduce the performance gap with supervised learning to less than 7.5% when learning to cluster 200 semantic classes. We refer to the supplementary materials for a confusion matrix which shows a clear diagonal. Most of the mistakes can be traced back to classes which are hard to disentangle, e.g. 'Giant Schnauzer' and 'Flat-coated Retriever' are both black dog breeds.

Prototypical behavior. We visualize the different clusters after training the model with the SCAN-loss. Specifically, we find the samples closest to the mean embedding of the confident samples in every cluster. The results are shown together with the name of the ground-truth class in Fig. 9. The discriminative features of each object are easily visible in every image. Therefore, we regard the obtained samples as "prototypes" of the different clusters. Notice that the performed experiment aligns well with prototypical networks [45], which present classes as the mean embedding of its supports.

[4] Since the overclustering case is evaluated using a many-to-one mapping, a direct comparison is not entirely fair. Still, we provide the comparison as an indication.

[5] DAC [6] also performs experiments on ImageNet, but only includes a few classes (15), which is arguably not much different from STL10.

Table 4: Validation set results for 50, 100 and 200 randomly selected classes from ImageNet. (*) Results obtained by running the publicly available code from [24]. (†) Results of applying K-means to the pretext task features from [49]

ImageNet	**50 Classes**				**100 Classes**				**200 Classes**			
Metric	Top-1	Top-5	NMI	ARI	Top-1	Top-5	NMI	ARI	Top-1	Top-5	NMI	ARI
Supervised	86.5	97.6	84.9	74.9	83.5	96.4	84.8	70.5	76.7	92.9	83.0	63.3
K-means†	27.1	-	23.1	11.9	22.8	-	23.1	9.6	16.7	-	22.9	6.6
IIC*	14.4	34.3	29.0	4.1	10.7	27.3	31.1	3.0	7.0	14.1	32.7	1.1
SCAN (Ours)	81.9	95.2	80.9	68.0	78.6	93.0	81.5	63.9	69.3	85.5	78.7	52.6

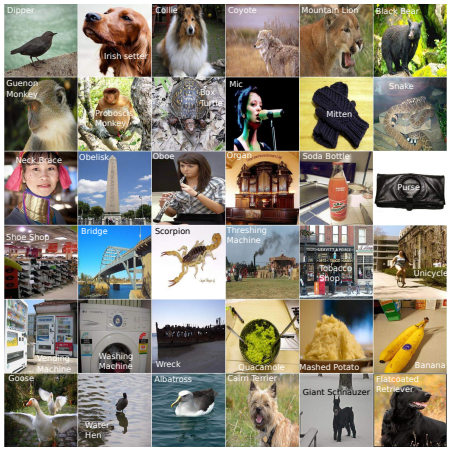


Fig. 9: Prototypes obtained by sampling the confident samples.

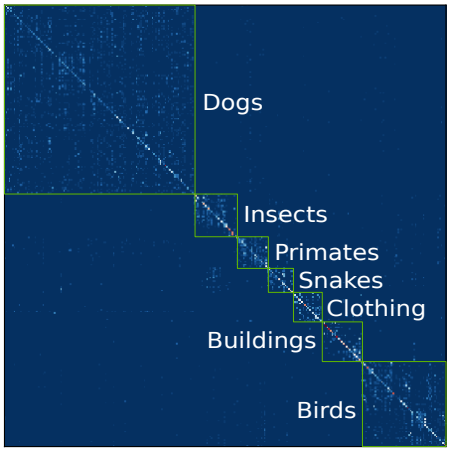


Fig. 10: Zoom on seven superclasses in the confusion matrix on ImageNet.

ImageNet - 1000 classes. Finally, the model is trained on the complete ImageNet dataset. Figure 11 shows images from the validation set which were assigned to the same cluster by our model. The obtained clusters are semantically meaningful, e.g. the first cluster contains cars, the second cluster contains images of (indoor) sports activities, while the third cluster shows images related to winter activities. Furthermore, the clusters capture a large variety of different backgrounds, viewpoints, etc. We conclude that (to a large extent) the model predictions are invariant to image features which do not alter the semantics. On the other hand, based on the ImageNet ground truth annotations, not all sample pairs should have been assigned to the same cluster. For example, the annotations discriminate between different primates, e.g. chimpanzee, baboon, langur, etc. We argue that there is not a single correct way of categorizing the images according to their semantics in case of ImageNet. Even for a human annotator, it is not straightforward to cluster each image according to the ImageNet classes without prior knowledge. Taking this into consideration, a quantitative analysis

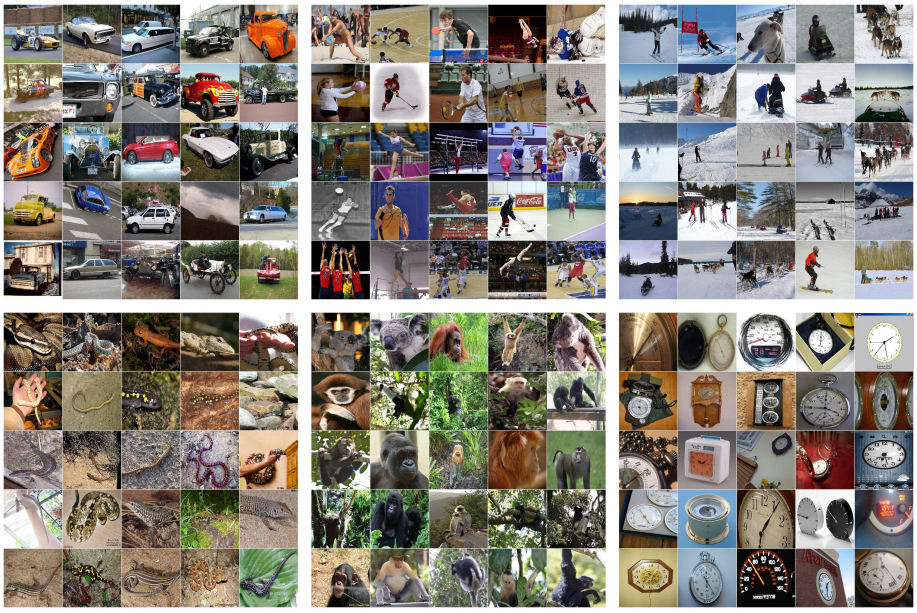


Fig. 11: Clusters extracted by our model on ImageNet (more in supplementary).

can be misguiding. However, we still evaluate our model using the ground-truth annotations for completeness (Top-1: 22%, Top-5: 39%, NMI: 60%, ARI: 11%).

Based on the ImageNet hierarchy we select class instances of the following superclasses: dogs, insects, primates, snake, clothing, buildings and birds. Figure 10 shows a confusion matrix of the selected classes. The confusion matrix has a block diagonal structure. The result show that the misclassified examples tend to be assigned to other clusters from within the same superclass, e.g. the model confuses two different dog breeds. We conclude that the model has learned to group images with similar semantics together, while its prediction errors can be attributed to the lack of annotations which could disentangle the fine-grained differences between some classes.

4 Conclusion

We presented a new framework to unsupervised image classification. The proposed approach comes with several advantages relative to recent works which adopted an end-to-end strategy. Experimental evaluation shows that the proposed method outperforms prior work by large margins, for a variety of datasets. Furthermore, positive results on ImageNet demonstrate that semantic clustering can be applied to large-scale datasets. Encouraged by these findings, we believe that our approach admits several extensions to other domains, e.g. semantic segmentation, semi-supervised learning and few-shot learning.

Acknowledgment. The authors thankfully acknowledge support by Toyota via the TRACE project and MACCHINA (KU Leuven, C14/18/065). Furthermore, we would like to thank Xu Ji for her valuable insights and comments. Finally, we thank Kevis-Kokitsi Maninis, Jonas Heylen and Mark De Wolf for their feedback.

References

1. Asano, Y.M., Rupprecht, C., Vedaldi, A.: Self-labelling via simultaneous clustering and representation learning. In: ICLR (2020)
2. Bengio, Y., Lamblin, P., Popovici, D., Larochelle, H.: Greedy layer-wise training of deep networks. In: NIPS (2007)
3. Bojanowski, P., Joulin, A.: Unsupervised learning by predicting noise. In: ICML (2017)
4. Caron, M., Bojanowski, P., Joulin, A., Douze, M.: Deep clustering for unsupervised learning of visual features. In: ECCV (2018)
5. Caron, M., Bojanowski, P., Mairal, J., Joulin, A.: Unsupervised pre-training of image features on non-curated data. In: ICCV (2019)
6. Chang, J., Wang, L., Meng, G., Xiang, S., Pan, C.: Deep adaptive image clustering. In: ICCV (2017)
7. Coates, A., Ng, A., Lee, H.: An analysis of single-layer networks in unsupervised feature learning. In: JMLR (2011)
8. Cubuk, E.D., Zoph, B., Mane, D., Vasudevan, V., Le, Q.V.: Autoaugment: Learning augmentation strategies from data. In: CVPR (2019)
9. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: CVPR (2009)
10. DeVries, T., Taylor, G.W.: Improved regularization of convolutional neural networks with cutout. arXiv preprint arXiv:1708.04552 (2017)
11. Doersch, C., Gupta, A., Efros, A.A.: Unsupervised visual representation learning by context prediction. In: ICCV (2015)
12. Donahue, J., Krähenbühl, P., Darrell, T.: Adversarial feature learning. In: ICLR (2017)
13. Donahue, J., Simonyan, K.: Large scale adversarial representation learning. In: NIPS (2019)
14. Feng, Z., Xu, C., Tao, D.: Self-supervised representation learning by rotation feature decoupling. In: CVPR (2019)
15. Gidaris, S., Singh, P., Komodakis, N.: Unsupervised representation learning by predicting image rotations. In: ICLR (2018)
16. Greff, K., Kaufmann, R.L., Kabra, R., Watters, N., Burgess, C., Zoran, D., Matthey, L., Botvinick, M., Lerchner, A.: Multi-object representation learning with iterative variational inference. In: ICML (2019)
17. Haeusser, P., Plapp, J., Golkov, V., Aljalbout, E., Cremers, D.: Associative deep clustering: Training a classification network with no labels. In: German Conference on Pattern Recognition (2018)
18. He, K., Fan, H., Wu, Y., Xie, S., Girshick, R.: Momentum contrast for unsupervised visual representation learning. arXiv preprint arXiv:1911.05722 (2019)
19. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR (2016)

20. Hénaff, O.J., Razavi, A., Doersch, C., Eslami, S., Oord, A.v.d.: Data-efficient image recognition with contrastive predictive coding. arXiv preprint arXiv:1905.09272 (2019)
21. Hu, W., Miyato, T., Tokui, S., Matsumoto, E., Sugiyama, M.: Learning discrete representations via information maximizing self-augmented training. In: ICML (2017)
22. Huang, J., Dong, Q., Gong, S., Zhu, X.: Unsupervised deep learning by neighbourhood discovery. In: ICML (2019)
23. Jenni, S., Favaro, P.: Self-supervised feature learning by learning to spot artifacts. In: CVPR (2018)
24. Ji, X., Henriques, J.F., Vedaldi, A.: Invariant information clustering for unsupervised image classification and segmentation. In: ICCV (2019)
25. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
26. Kingma, D.P., Welling, M.: Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114 (2013)
27. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images (2009)
28. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: NIPS (2012)
29. Larsson, G., Maire, M., Shakhnarovich, G.: Colorization as a proxy task for visual understanding. In: CVPR (2017)
30. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: ECCV (2014)
31. McLachlan, G.J.: Iterative reclassification procedure for constructing an asymptotically optimal rule of allocation in discriminant analysis. Journal of the American Statistical Association (1975)
32. Misra, I., van der Maaten, L.: Self-supervised learning of pretext-invariant representations. arXiv preprint arXiv:1912.01991 (2019)
33. Nathan Mundhenk, T., Ho, D., Chen, B.Y.: Improvements to context based self-supervised learning. In: CVPR (2018)
34. Ng, A.: Sparse autoencoder. CS294A Lecture notes (2011)
35. Noroozi, M., Favaro, P.: Unsupervised learning of visual representations by solving jigsaw puzzles. In: ECCV (2016)
36. Noroozi, M., Pirsiavash, H., Favaro, P.: Representation learning by learning to count. In: ICCV (2017)
37. Noroozi, M., Vinjimoor, A., Favaro, P., Pirsiavash, H.: Boosting self-supervised learning via knowledge transfer. In: CVPR (2018)
38. Oord, A.v.d., Li, Y., Vinyals, O.: Representation learning with contrastive predictive coding. arXiv preprint arXiv:1807.03748 (2018)
39. Pathak, D., Krahenbuhl, P., Donahue, J., Darrell, T., Efros, A.A.: Context encoders: Feature learning by inpainting. In: CVPR (2016)
40. Radford, A., Metz, L., Chintala, S.: Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:1511.06434 (2015)
41. Ren, Z., Jae Lee, Y.: Cross-domain self-supervised multi-task feature learning using synthetic imagery. In: CVPR (2018)
42. Schultz, M., Joachims, T.: Learning a distance metric from relative comparisons. In: NIPS (2004)
43. Scudder, H.: Probability of error of some adaptive pattern-recognition machines. IEEE Transactions on Information Theory (1965)

44. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: ICLR (2015)
45. Snell, J., Swersky, K., Zemel, R.: Prototypical networks for few-shot learning. In: NIPS (2017)
46. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: CVPR (2016)
47. Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., Manzagol, P.A.: Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. JMLR (2010)
48. Wang, J., Wang, J., Song, J., Xu, X.S., Shen, H.T., Li, S.: Optimized cartesian k-means. IEEE Transactions on Knowledge & Data Engineering (2015)
49. Wu, Z., Xiong, Y., Yu, S.X., Lin, D.: Unsupervised feature learning via non-parametric instance discrimination. In: CVPR (2018)
50. Xie, J., Girshick, R., Farhadi, A.: Unsupervised deep embedding for clustering analysis. In: ICML (2016)
51. Xie, S., Girshick, R., Dollár, P., Tu, Z., He, K.: Aggregated residual transformations for deep neural networks. In: CVPR (2017)
52. Yang, J., Parikh, D., Batra, D.: Joint unsupervised learning of deep representations and image clusters. In: CVPR (2016)
53. Zelnik-Manor, L., Perona, P.: Self-tuning spectral clustering. In: NIPS (2005)
54. Zhang, L., Qi, G.J., Wang, L., Luo, J.: Aet vs. aed: Unsupervised representation learning by auto-encoding transformations rather than data. In: CVPR (2019)
55. Zhang, R., Isola, P., Efros, A.A.: Colorful image colorization. In: ECCV (2016)
56. Zhao, J., Mathieu, M., Goroshin, R., Lecun, Y.: Stacked what-where auto-encoders. arXiv preprint arXiv:1506.02351 (2015)

Supplementary Material

A Smaller datasets

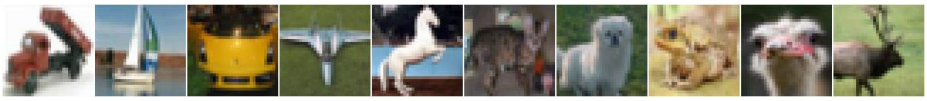
We include additional qualitative results on the smaller datasets, i.e. CIFAR10 [27], CIFAR100-20 [27] and STL10 [7]. All results are obtained with the models used in the state-of-the-art comparison in the main paper.

A.1 Prototypical examples

Figure S1 visualizes a prototype image for every cluster on CIFAR10, CIFAR100-20 and STL-10. The object of interest is clearly recognizable in the images. It is worth noting that the prototypical examples on CIFAR10 and STL10 can be matched with the ground-truth classes of the dataset. This is not the case for CIFAR100-20, e.g. *lawn-mower* and *tank* belong to the *vehicles 2* ground-truth superclass. We hypothesize that the reason can be found in the use of superclasses, which increases the intra-class variance. As a consequence, the model finds clusters which differ from the ground-truth clusters, which is arguably also the reason for the lower performance on CIFAR100-20. Note that we can reduce the mismatch between the predicted clusters and the ground-truth clusters by applying overclustering as shown in the main paper.

Fig. S1: Prototype images on the smaller datasets.

(a) CIFAR10



(b) STL10

(c) CIFAR100-20

Fig. S2: Low confidence predictions.

(a) CIFAR10

(b) STL10

(c) CIFAR100-20

A.2 Low confidence examples

Figure S2 shows examples for which the network produces low confidence predictions. In most cases, it is hard to determine the correct class label. The difficult examples include objects which are: only partially visible, occluded, under bad lighting conditions, etc.

B ImageNet

B.1 Results on smaller subsets

Quantitative results We sampled five random subsets of 50, 100 and 200 classes on ImageNet. The main paper only discusses the results on the first subset. Table S1 reports additional results across all five subsets, in order to show the influence of the selected classes on the clustering performance. The last column measures the average performance across the five subsets. We report similar performance levels across all subsets. The method generalizes since it is independent of the classes that are included in the subsets.

We provide the classes for every subset in separate txt-files. Future research can use the smaller ImageNet subsets to benchmark their results. Finally, we provide a comparison with K-means and supervised learning in Table S2. We used the pretext task features from [49] to cluster the images with K-means. The conclusions are similar as for the ImageNet experiments in the main paper.

Confusion matrix Figure S3 shows a confusion matrix for one of the ImageNet-50 subsets. Most of the mistakes can be found between classes that are hard to

disentangle, e.g. *'Giant Schnauzer'* and *'Flat-coated Retriever'* are both black dog breeds, *'Hatchet'* and *'Woodworking Plane'* are both tools, etc.

Prototype examples Figure S4 shows prototype images for a model trained on one of the ImageNet-50 subsets. The figure is an extended version of Figure 9 in the paper. For every cluster, we sampled one prototype image. Remarkably, the prototype images can be matched exactly with the 50 ground truth classes.

Low confidence examples Figure S5 shows examples for which the model produces low confidence predictions on one of the ImageNet-50 subset. In a number of cases, the low confidence output can be attributed to multiple objects being visible in the scene. Other cases can be explained by the partial visibility of the object, distracting elements in the scene, or ambiguity of the object of interest.

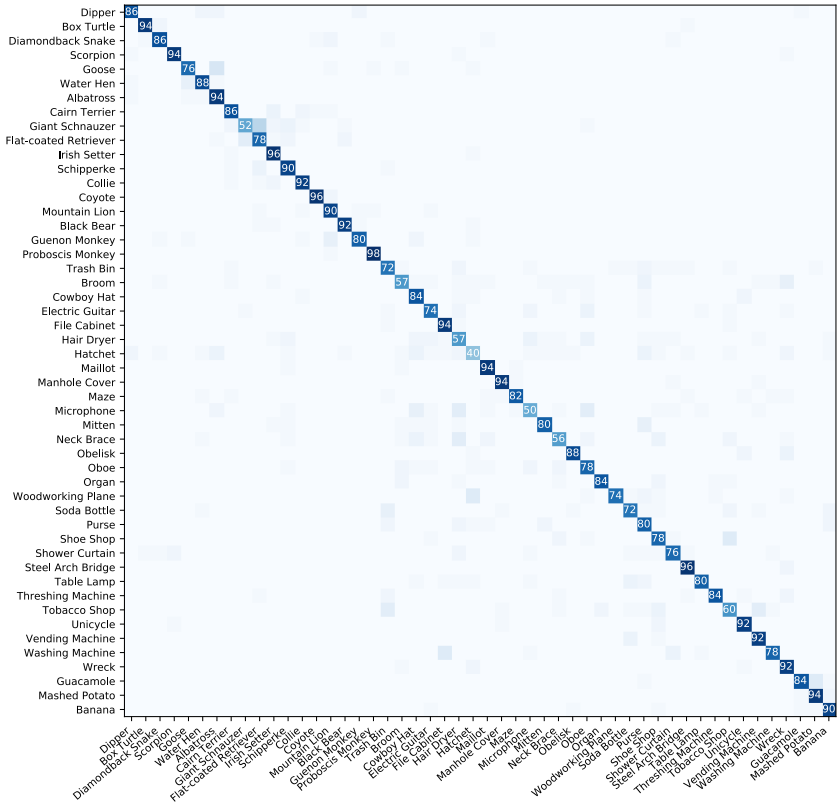


Fig. S3: Confusion matrix on ImageNet-50. Most mistakes can be found between classes that are visually similar.

Table S1: Additional results for randomly selected classes from ImageNet.

(a) Results on 50 classes.

Metric	Subset 1	Subset 2	Subset 3	Subset 4	Subset 5	Avg ± Std
Top - 1	81.9	78.6	79.0	81.3	78.2	79.8 ± 1.5
Top - 5	95.2	93.0	93.8	95.0	92.2	93.8 ± 1.1
NMI	80.9	77.6	79.0	81.7	79.3	79.7 ± 1.4
ARI	68.0	62.4	65.0	68.5	65.0	65.8 ± 2.2

(b) Results on 100 classes.

Metric	Subset 1	Subset 2	Subset 3	Subset 4	Subset 5	Avg ± Std
Top - 1	78.6	74.3	78.5	77.7	75.4	76.9 ± 1.7
Top - 5	93.0	90.5	91.9	93.3	92.0	92.1 ± 1.0
NMI	81.5	78.5	81.2	80.6	79.2	80.2 ± 1.2
ARI	63.9	58.7	63.9	62.3	60.0	61.8 ± 2.1

(c) Results on 200 classes.

Metric	Subset 1	Subset 2	Subset 3	Subset 4	Subset 5	Avg ± Std
Top - 1	69.3	66.9	69.6	69.5	67.3	68.5 ± 1.2
Top - 5	85.5	83.3	84.5	85.9	85.3	84.9 ± 0.9
NMI	78.7	76.4	78.1	78.1	76.6	77.6 ± 0.9
ARI	52.6	49.7	52.7	52.1	49.6	51.3 ± 1.4

Table S2: Validation set results for 50, 100 and 200 randomly selected classes from ImageNet. We report the average performance measured on five folds. † Results of applying K-means to the pretext task features from [49]

	50 Classes				**100 Classes**				**200 Classes**			
Metric	Top-1	Top-5	NMI	ARI	Top-1	Top-5	NMI	ARI	Top-1	Top-5	NMI	ARI
Supervised	85.2	96.7	83.6	72.9	81.5	95.4	83.0	68.2	77.6	93.2	82.2	61.8
K-means†	27.6	-	23.9	12.2	23.2	-	23.8	9.9	17.0	-	22.8	6.7
Ours	79.8	93.8	79.7	65.8	76.9	92.1	80.2	61.8	68.5	84.9	77.6	51.3

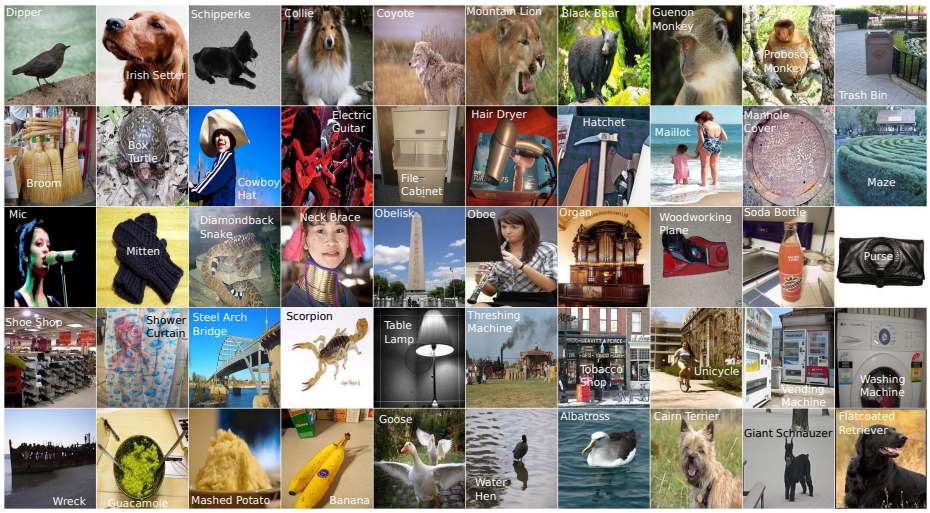


Fig. S4: Prototype images on ImageNet-50.

Fig. S5: Low confidence examples on ImageNet-50.

B.2 ImageNet - 1000 classes

Figures S6, S7 and S8 show images from the validation set that were assigned to the same cluster. The figures can be viewed together with Figure 11 in the main paper. Incorrect clusters are visualized in Figure S9. The failure cases are present when the network focuses too much on the background, or when the network cannot easily discriminate between the images in the same cluster. However, in most cases, we can still attach some semantic meaning to the clusters, e.g. animals in cages, white fences, etc.

C Experimental setup

C.1 Datasets

We performed our experimental evaluation on the CIFAR10 [27], CIFAR100-20 [27], STL10 [7] and ImageNet [9] datasets. In addition to the full ImageNet dataset, we also included subsets of 50, 100 and 200 randomly selected classes. To be able to capture the variance of the dataset, we sampled 5 folds of 50, 100

Fig. S6: Example clusters of ImageNet-1000 (1).

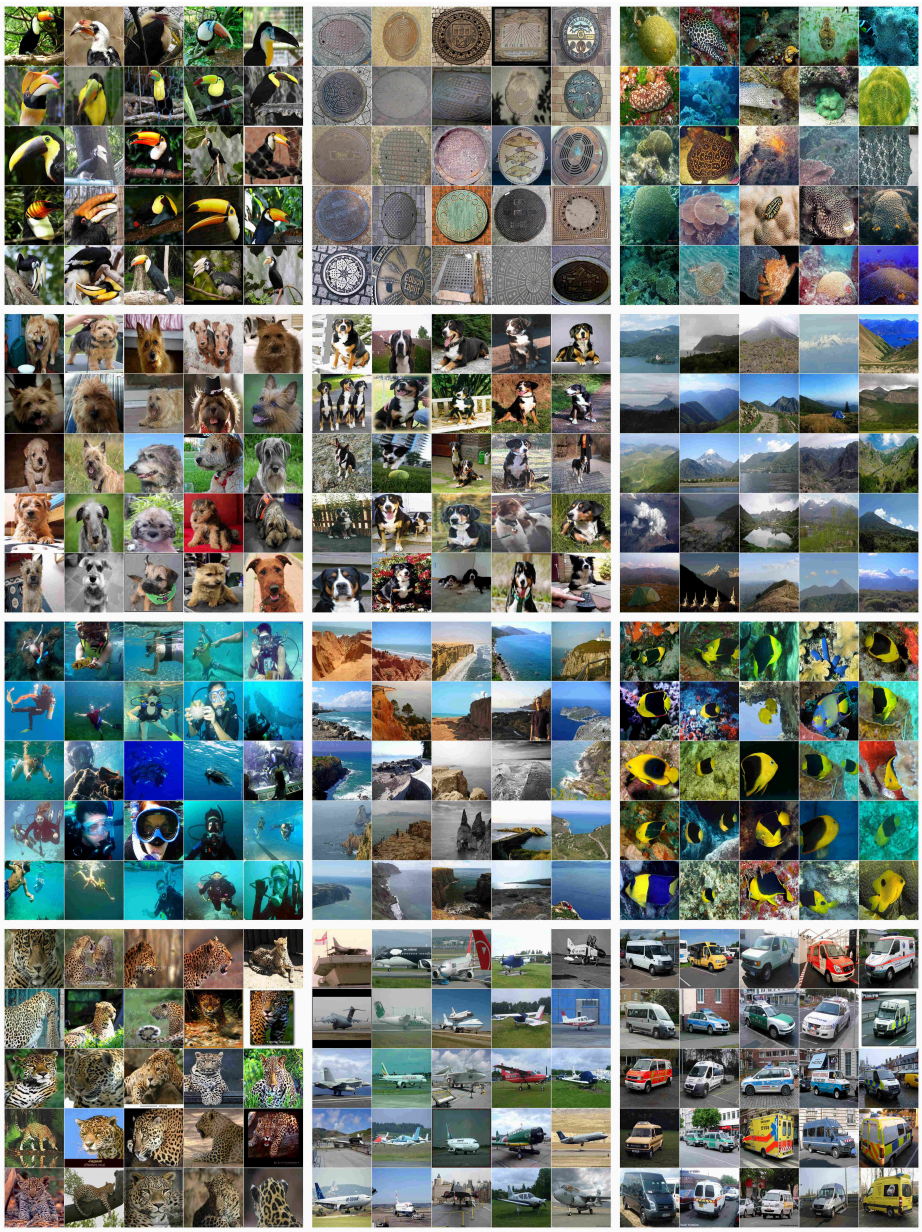


Fig. S7: Example clusters of ImageNet-1000 (2).

Fig. S8: Example clusters of ImageNet-1000 (3).

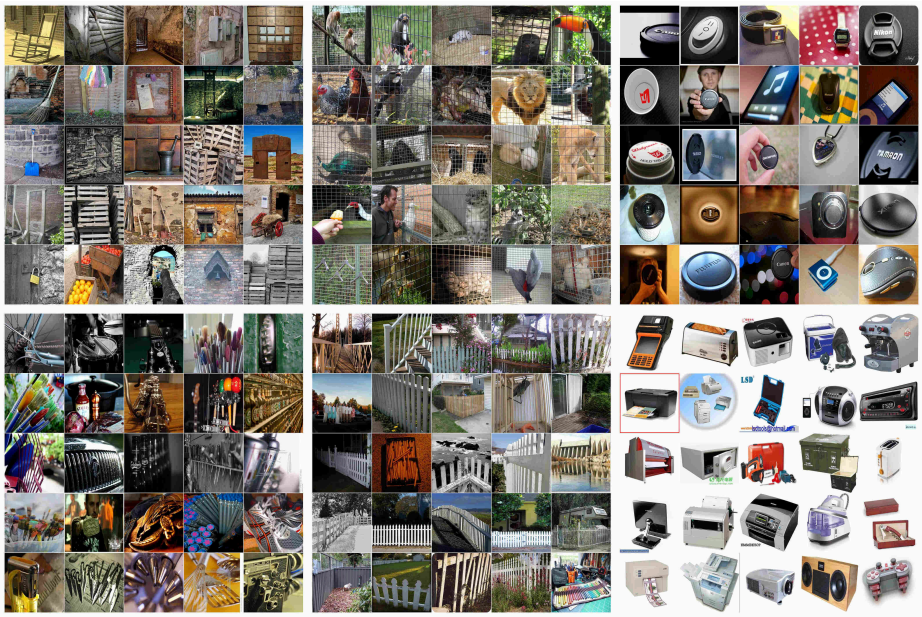


Fig. S9: Incorrect clusters of ImageNet-1000 predicted by our model.

and 200 classes. The sets of 50 and 100 classes are subsets of the 100 and 200 classes respectively. We provide the selected classes in the *imagenet_x_y.txt* files, with x the number of classes and y the fold. Table S3 provides an overview of the number of classes, the number of images and the aspect ratio of the used datasets.

Following prior work [24,6,50,52], we train and evaluate on the full dataset for CIFAR10, CIFAR100, and STL10. Notice that the twenty superclasses on CIFAR100 are used to evaluate the results, denoted as CIFAR100-20. On ImageNet, we use the regular train and validation split. We encourage future work to do the same.

Table S3: Datasets overview

Dataset	**Classes**	**Train images**	**Aspect ratio**
CIFAR10	10	60,000	32 x 32
CIFAR100-20	20	60,000	32 x 32
STL10	10	13,000	96 x 96
ImageNet-50	50	65,000	224 x 224
ImageNet-100	100	130,000	224 x 224
ImageNet-200	200	260,000	224 x 224
ImageNet	1000	1,281,167	224 x 224

Table S4: List of transformations.

Transformation	**Parameter**	**Interval**
Identity	-	-
Autocontrast	-	-
Equalize	-	-
Rotate	θ	$[-30, 30]$
Solarize	T	$[0, 256]$
Color	C	$[0.05, 0.95]$
Contrast	C	$[0.05, 0.95]$
Brightness	B	$[0.05, 0.95]$
Sharpness	S	$[0.05, 0.95]$
Shear X	R	$[-0.1, 0.1]$
Translation X	λ	$[-0.1, 0.1]$
Translation Y	λ	$[-0.1, 0.1]$
Posterize	B	$[4, 8]$
Shear Y	R	$[-0.1, 0.1]$

C.2 Augmentations

It is beneficial to apply strong augmentations to the samples X_i and their neighbors $\mathcal{N}_{X_i}$. We apply four random transformations from AutoAugment [8], followed by Cutout [10]. The transformation parameters are uniformly sampled

between fixed intervals. Table S4 provides a detailed overview. We apply the same augmentation strategy across all datasets.

Table S5: ImageNet hyperparameters.

Dataset	Batchsize	Entropy Weight
ImageNet-50	512	2.0
ImageNet-100	512	3.0
ImageNet-200	512	3.5
ImageNet	2048	4.0

C.3 Hyperparameters

The majority of experiments was performed using the same hyperparameters (see experiments section). However, due to the larger number of classes on ImageNet, we increased the batch size and the entropy weight λ. Our strategy for setting the entropy weight is to gradually increase its value, until there is no significant drop in entropy during training. A higher entropy weight avoids the network from minimizing the consistency loss by collapsing to a few classes. Table S5 reports the used hyperparameters on ImageNet.