SCAN: 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. Can we automatically group images into semantically meaningful clusters when ground-truth annotations are absent? The task of unsupervised image classification remains an important, and open challenge in computer vision. Several recent 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 large margins, in particular +26.6% on CIFAR10, +25.0% on CIFAR100-20 and +21.3% on STL10 in terms of classification accuracy. Furthermore, our method is the first to perform well on a large-scale dataset for image classification. In particular, we obtain promising results on ImageNet, and outperform several semi-supervised learning methods in the low-data regime without the use of any ground-truth annotations. The code is made publicly available **here**.

Keywords: Unsupervised Learning, Self-Supervised 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 [28,44,19,53,47] on large-scale datasets [11,30] 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

⋆ Authors contributed equally

goal in this case is to group the images into clusters, such that images within 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 [13,39,58,35,16] 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 [13,33], inpainting patches [39], solving jigsaw puzzles [35,37], colorizing images [58,29], using adversarial training [14,15], predicting noise [3], counting [36], predicting rotations [16], spotting artifacts [23], generating images [41], using predictive coding [38,20], performing instance discrimination [51,18,7,48,32], 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 [52], DAC [6], DeepCluster [4], DeeperCluster [5], or others [1,17,54]) 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 [6,52], 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 [6,52,4,5,22,17,54], or prone to degenerate solutions [4,5], thus requiring special mechanisms (e.g. pretraining, cluster reassignment and feature cleaning) to avoid those situations. Most importantly, since the cluster learning depends on the network initialization, they are likely to latch onto low-level features, like color, which is unwanted for the objective of semantic clustering. To partially alleviate this problem, some works [24,21,4] are tied to the use of specific preprocessing (e.g. Sobel filtering).

In this work we advocate a two-step approach for unsupervised image classification, in contrast to recent end-to-end learning approaches. The proposed method, named SCAN (Semantic Clustering by Adopting Nearest neighbors), 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, which is known to lead to cluster degeneracy [4], 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. 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 large margins across multiple datasets. Furthermore, we report promising results on the large-scale ImageNet dataset. 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 show how to 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.

End-to-end learning approaches have utilized the architecture of CNNs as a prior [54,6,52,17,4,5], or enforced consistency between images and their augmentations [24,21] to disentangle the clusters. In both cases, the cluster learning is

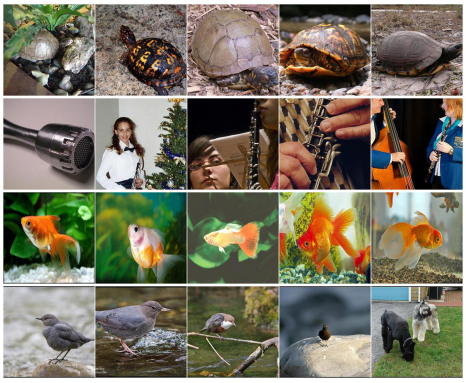


Fig. 1: Images (first column) and their nearest neighbors (other columns) [51].

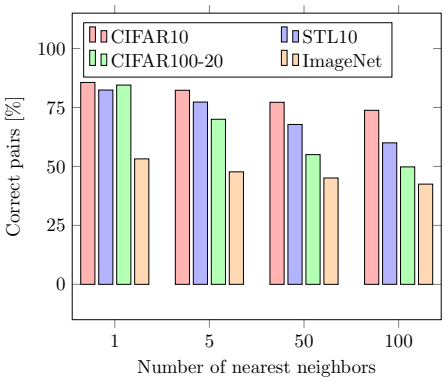


Fig. 2: Neighboring samples tend to be instances of the same semantic class.

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, 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 [16], affine or perspective transformation prediction [57], colorization [29], in-painting [39], instance discrimination [51,18,7,32], 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 [51,18,7,32] 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 [51] 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 information from its input that is 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 leverage the pretext features 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 [7,18]. 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 [9] and ImageNet [11]) for different values of K. Motivated by this observation, we propose to adopt the nearest neighbors obtained through the pretext task τ 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 terminates in a softmax function to perform 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:

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

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

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

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 [52,6,24], 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. Unlike [40,25,49,2,34,59,52] 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.

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 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 [6,4,52], 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 [43,31,46] to exploit the already well-classified examples, and correct for mistakes due to noisy nearest neighbors.

[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

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. The 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 it as SCAN, i.e. Semantic Clustering by Adopting Nearest neighbors.

3 Experiments

3.1 Experimental setup

Datasets. The experimental evaluation is performed on CIFAR10 [27], CIFAR100-20 [27], STL10 [9] and ImageNet [11]. We focus on the smaller datasets first. The results on ImageNet are discussed separately in Section 3.5. Some prior works [24,6,52,54] trained and evaluated on the complete datasets. Differently, we train and evaluate using the train and val split respectively. Doing so, allows to study the generalization properties of the method for novel unseen examples. Note that this does not result in any unfair advantages compared to prior work. The results are reported as the mean and standard deviation from 10 different runs. Finally, 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 20 nearest neighbors are determined through an instance discrimination task based on noise contrastive estimation (NCE) [51]. We adopt the SimCLR [7] implementation for the instance discrimination task on the smaller datasets, and

the implementation from MoCo [8] on ImageNet. The selected pretext task satisfies the feature invariance constraint from Equation 1 w.r.t. the transformations applied to augment the input images. In particular, every image is disentangled as a unique instance independent of the applied transformation. 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 $\lambda = 5$. 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 200 epochs using the self-labeling procedure with threshold 0.99 (Section 2.3). A weighted cross-entropy loss compensates for the imbalance between confident samples across clusters. The class weights are inversely proportional to the number of occurrences in the batch after thresholding. The network weights are updated through Adam [25] with learning rate 10^{-4} and weight decay 10^{-4}. The images are strongly augmented by composing four randomly selected transformations from RandAugment [10] during both the clustering and self-labeling steps. The transformation parameters are uniformly sampled between fixed intervals. For more details visit 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 different parts of our method through an ablation study on CIFAR10 in Table 1. K-means clustering of the NCE pretext features results in the lowest accuracy (65.9%), and is characterized by a large variance (5.7%). This is to be expected since the cluster assignments can be imbalanced (Figure 3), and are not guaranteed to align with the ground-truth classes. Interestingly, applying K-means to the pretext features outperforms prior state-of-the-art methods for unsupervised classification based on end-to-end learning schemes (see Sec. 3.3). This observation supports our primary claim, i.e. it is beneficial to separate feature learning from clustering. Updating the network weights through the SCAN-loss - while augmenting the input images through SimCLR transformations - outperforms K-means (+15.9%). Note that the SCAN-loss is somewhat related to K-means, since both methods employ the pretext features as their prior to cluster the images. Differently, our loss avoids the cluster degeneracy issue. We also research the effect of using different augmentation strategies during training. Applying transformations from RandAgument (RA) to both the samples and their mined neighbors further improves the performance (78.7% vs. 81.8%). We hypothesize that strong augmentations help to reduce the solution space by imposing additional invariances.

Fine-tuning the network through self-labeling further enhances the quality of the cluster assignments (81.8% to 87.6%). During self-labeling, the network corrects itself as it gradually becomes more confident (see Figure 4). Importantly,

Table 1: Ablation Method CIFAR10

Setup	**ACC** (Avg ± Std)
Pretext + K-means	65.9 ± 5.7
SCAN-Loss (SimCLR)	78.7 ± 1.7
(1) Self-Labeling (SimCLR)	10.0 ± 0
(2) Self-Labeling (RA)	87.4 ± 1.6
SCAN-Loss (RA)	81.8 ± 1.7
(1) Self-Labeling (RA)	87.6 ± 0.4

Table 2: Ablation Pretext CIFAR10

Pretext Task	**Clustering**	**ACC** (Avg ± Std)
RotNet [16]	K-means	27.1 ± 2.1
	SCAN	74.3 ± 3.9
Inst. discr. [51]	K-means	52.0 ± 4.6
	SCAN	83.5 ± 4.1
Inst. discr. [7]	K-means	65.9 ± 5.7
	SCAN	87.6 ± 0.4

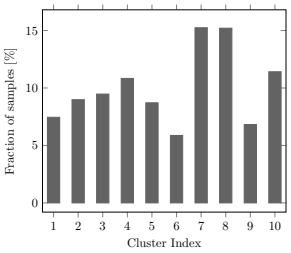


Fig. 3: K-means cluster assignments are imbalanced.

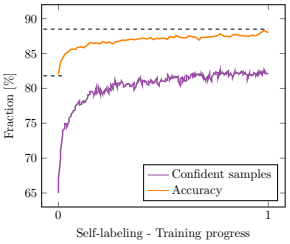


Fig. 4: Acc. and the number of confident samples during self-labeling.

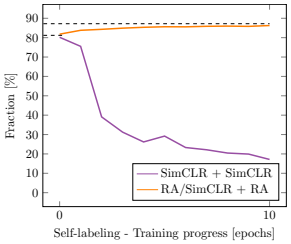


Fig. 5: Self-labeling with SimCLR or RandAugment augmentations.

in order for self-labeling to be successfully applied, a shift in augmentations is required (see Table 1 or Figure 5). We hypothesize that this is required to prevent the network from overfitting on already well-classified examples. Finally, Figure 6 shows that self-labeling procedure is not sensitive to the threshold's value.

Pretext task. We study the effect of using different pretext tasks to mine the nearest neighbors. In particular we consider two different implementations of the instance discrimination task from before [51,7], and RotNet [16]. The latter trains the network to predict image rotations. As a consequence, the distance between an image X_i and its augmentations $T[X_i]$ is not minimized in the embedding space of a model pretrained through RotNet (see Equation 1). Differently, the instance discrimintation task satisfies the invariance criterion w.r.t. the used augmentations. Table 2 shows the results on CIFAR10.

First, we observe that the proposed method is not tied to a specific pretext task. All cases report high accuracy ($> 70\%$). Second, pretext tasks that satisfy the invariance criterion are better suited to mine the nearest neighbors, i.e. 83.5% and 87.6% for inst. discr. versus 74.3% for RotNet. This confirms our hypothesis from Section 2.1, i.e. it is beneficial to choose a pretext task which imposes invariance between an image and its augmentations.

Number of neighbors. Figure 7 shows the influence of using a different number of nearest neighbors K during the clustering step. The results are not very

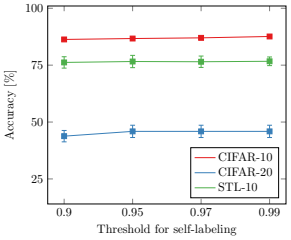


Fig. 6: Ablation threshold during self-labeling step.

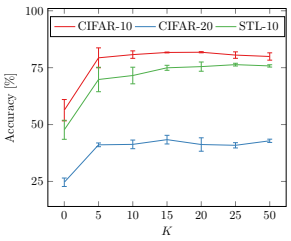


Fig. 7: Influence of the used number of neighbors K.

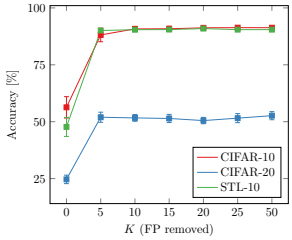


Fig. 8: Results without false positives in the nearest neighbors.

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 very new dataset. In fact, both robustness and accuracy improve when increasing the value of K upto a certain value. We also consider the corner case $K = 0$, when only enforcing consistent predictions for images and their augmentations. the performance decreases on all three datasets compared to $K = 5$, 56.3% vs 79.3% on CIFAR10, 24.6% vs 41.1% on CIFAR100-20 and 47.70% vs 69.8% on STL10. This confirms that better representations can be learned by also 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 fully-supervised performance 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), 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. +26.6% on CIFAR10, +25.0% on CIFAR100-20 and +21.3% on STL10 in terms of accuracy. We also compare with the state-of-the-art in representation learning [7] (Pretext + K-means). As shown in Section 3.2, our method outperforms the application of K-means on the pretext features. Finally, we also include results when tackling the problem in a fully-supervised manner. Our

Table 3: State-of-the-art comparison: We report the averaged results for 10 different runs after the clustering (∗) and self-labeling steps (†), and the best model. Opposed to prior work, we train and evaluate using the train and val split respectively, instead of using the full dataset for both training and testing.

Dataset	**CIFAR10**			**CIFAR100-20**			**STL10**		
Metric	ACC	NMI	ARI	ACC	NMI	ARI	ACC	NMI	ARI
K-means [50]	22.9	8.7	4.9	13.0	8.4	2.8	19.2	12.5	6.1
SC [55]	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 [54]	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 [49]	29.7	25.1	16.3	15.1	11.1	4.6	30.2	22.4	15.2
SWWAE [59]	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 [52]	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
Supervised	93.8	86.2	87.0	80.0	68.0	63.2	80.6	65.9	63.1
Pretext [7] + K-means	65.9 ± 5.7	59.8 ± 2.0	50.9 ± 3.7	39.5 ± 1.9	40.2 ± 1.1	23.9 ± 1.1	65.8 ± 5.1	60.4 ± 2.5	50.6 ± 4.1
SCAN* (Avg ± Std)	81.8 ± 0.3	71.2 ± 0.4	66.5 ± 0.4	42.2 ± 3.0	44.1 ± 1.0	26.7 ± 1.3	75.5 ± 2.0	65.4 ± 1.2	59.0 ± 1.6
SCAN† (Avg ± Std)	87.6 ± 0.4	78.7 ± 0.5	75.8 ± 0.7	45.9 ± 2.7	46.8 ± 1.3	30.1 ± 2.1	76.7 ± 1.9	68.0 ± 1.2	61.6 ± 1.8
SCAN† (Best)	**88.3**	**79.7**	**77.2**	**50.7**	**48.6**	**33.3**	**80.9**	**69.8**	**64.6**
SCAN† (Overcluster)	86.2 ± 0.8	77.1 ± 0.1	73.8 ± 1.4	55.1 ± 1.6	50.0 ± 1.1	35.7 ± 1.7	76.8 ± 1.1	65.6 ± 0.8	58.6 ± 1.6

model obtains close to supervised performance on CIFAR-10 and STL-10. The performance gap is larger on CIFAR100-20, due to the use of superclasses.

Other advantages. In contrast to prior work [6,24,21], we did not have to perform any dataset specific fine-tuning. Furthermore, the results on CIFAR10 can be obtained within 6 hours on a single GPU. As a comparison, training the model from [24] requires at least a day of training time.

3.4 Overclustering

So far we assumed to have knowledge about the number of ground-truth classes. The method 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 (87.6% to 86.2%) and STL10 (76.7% to 76.8%), and improves for CIFAR100-20 (45.9% to 55.1%)[3]. We conclude that the approach does not require knowledge of the exact number of clusters. We hypothesize that the increased performance on CIFAR100-20 is related to the higher intra-class variance. More specifically, CIFAR100-20 groups multiple object categories together in superclasses. In this case, an overclustering is better suited to explain the intra-class variance.

[3] 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.

Table 4: Validation set results for 50, 100 and 200 randomly selected classes from ImageNet. The results with K-means were obtained using the pretext features from MoCo [8]. We provide the results obtained by our method after the clustering step ($*$), and after the self-labeling step ($\dagger$).

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
K-means	65.9	-	77.5	57.9	59.7	-	76.1	50.8	52.5	-	75.5	43.2
SCAN*	75.1	91.9	80.5	63.5	66.2	88.1	78.7	54.4	56.3	80.3	75.7	44.1
SCAN†	76.8	91.4	82.2	66.1	68.9	86.1	80.8	57.6	58.1	80.6	77.2	47.0

3.5 ImageNet

Setup. We consider the problem of unsupervised image classification on the large-scale ImageNet dataset [11]. 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. Additional details of the training setup can be found in the supplementary materials.

Quantitative evaluation. Table 4 compares our results against applying K-means on the pretext features from MoCo [8]. Surprisingly, the application of K-means already performs well on this challenging task. We conclude that the pretext features are well-suited for the down-stream task of semantic clustering. Training the model with the SCAN-loss again outperforms the application of K-means. Also, the results are further improved when fine-tuning the model through self-labeling. We do not include numbers for the prior state-of-the-art [24], since we could not obtain convincing results on ImageNet when running the publicly available code. We refer the reader to the supplementary materials for additional qualitative results on ImageNet-50.

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 top-10 most confident samples in every cluster. The results are shown together with the name of the matched ground-truth classes in Fig. 9. Importantly, we observe that the found samples align well with the classes of the dataset, except for 'oboe' and 'guacamole' (red). Furthermore, the discriminative features of each object class are clearly present in the images. Therefore, we regard the obtained samples as "prototypes" of the various clusters. Notice that the performed experiment aligns well with prototypical networks [45].

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

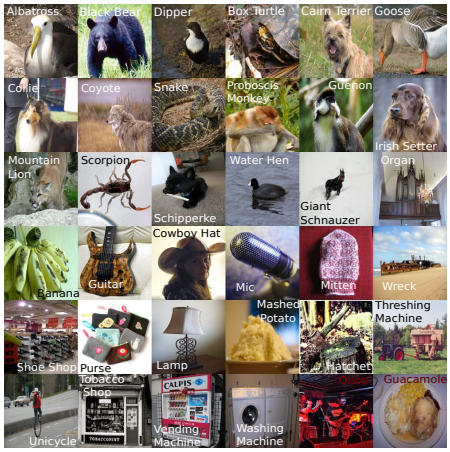


Fig. 9: Prototypes obtained by sampling the confident samples.

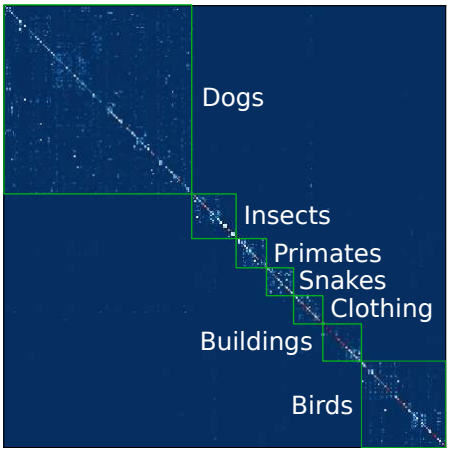


Fig. 10: Zoom on seven superclasses in the confusion matrix on ImageNet.

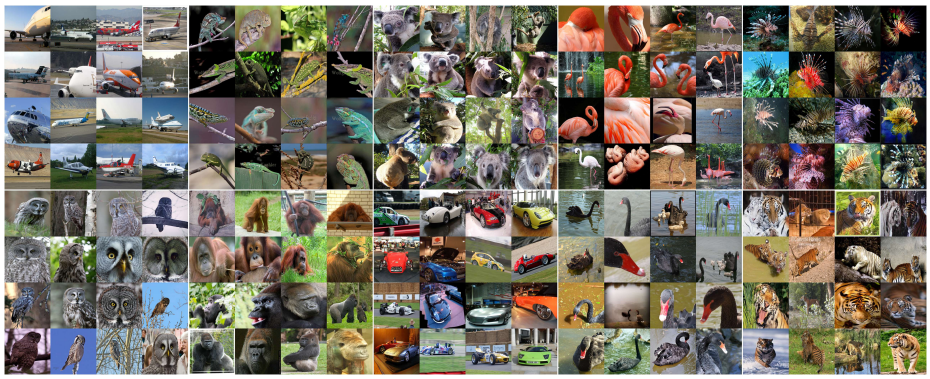


Fig. 11: Clusters extracted by our model on ImageNet (more in supplementary).

meaningful, e.g. planes, cars and primates. 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 ground-truth 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.

Based on the ImageNet hierarchy we select class instances of the following superclasses: dogs, insects, primates, snake, clothing, buildings and birds. Fig-

ure 10 shows a confusion matrix of the selected classes. The confusion matrix has a block diagonal structure. The results 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.

Finally, Table 5 compares our method against recent semi-supervised learning approaches when using 1% of the images as labelled data. We obtain the following quantitative results on ImageNet: Top-1: 39.9%, Top-5: 60.0%, NMI: 72.0%, ARI: 27.5%. Our method outperforms several semi-supervised learning approaches, without using labels. This further demonstrates the strength of our approach.

Table 5: Comparison with supervised, and semi-supervised learning methods using 1% of the labelled data on ImageNet.

Method	**Backbone**	**Labels**	**Top-1**	**Top-5**
Supervised Baseline	ResNet-50	✓	25.4	48.4
Pseudo-Label	ResNet-50	✓	-	51.6
VAT + Entropy Min. [56]	ResNet-50	✓	-	47.0
InstDisc [51]	ResNet-50	✓	-	39.2
BigBiGAN [15]	ResNet-50(4x)	✓	-	55.2
PIRL [32]	ResNet-50	✓	-	57.2
CPC v2 [20]	ResNet-161	✓	52.7	77.9
SimCLR [7]	ResNet-50	✓	48.3	75.5
SCAN (Ours)	ResNet-50	✗	39.9	60.0

4 Conclusion

We presented a novel 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. Chen, T., Kornblith, S., Norouzi, M., Hinton, G.: A simple framework for contrastive learning of visual representations. arXiv preprint arXiv:2002.05709 (2020)
8. Chen, X., Fan, H., Girshick, R., He, K.: Improved baselines with momentum contrastive learning. arXiv preprint arXiv:2003.04297 (2020)
9. Coates, A., Ng, A., Lee, H.: An analysis of single-layer networks in unsupervised feature learning. In: JMLR (2011)
10. Cubuk, E.D., Zoph, B., Shlens, J., Le, Q.V.: Randaugment: Practical automated data augmentation with a reduced search space. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops. pp. 702–703 (2020)
11. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: CVPR (2009)
12. DeVries, T., Taylor, G.W.: Improved regularization of convolutional neural networks with cutout. arXiv preprint arXiv:1708.04552 (2017)
13. Doersch, C., Gupta, A., Efros, A.A.: Unsupervised visual representation learning by context prediction. In: ICCV (2015)
14. Donahue, J., Krähenbühl, P., Darrell, T.: Adversarial feature learning. In: ICLR (2017)
15. Donahue, J., Simonyan, K.: Large scale adversarial representation learning. In: NIPS (2019)
16. Gidaris, S., Singh, P., Komodakis, N.: Unsupervised representation learning by predicting image rotations. In: ICLR (2018)
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. In: arXiv preprint arXiv:1911.05722 (2020)
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. In: CVPR (2020)
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. Sohn, K., Berthelot, D., Li, C.L., Zhang, Z., Carlini, N., Cubuk, E.D., Kurakin, A., Zhang, H., Raffel, C.: Fixmatch: Simplifying semi-supervised learning with consistency and confidence. arXiv preprint arXiv:2001.07685 (2020)
47. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: CVPR (2016)
48. Tian, Y., Krishnan, D., Isola, P.: Contrastive multiview coding. arXiv preprint arXiv:1906.05849 (2019)
49. 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)
50. 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)
51. Wu, Z., Xiong, Y., Yu, S.X., Lin, D.: Unsupervised feature learning via non-parametric instance discrimination. In: CVPR (2018)
52. Xie, J., Girshick, R., Farhadi, A.: Unsupervised deep embedding for clustering analysis. In: ICML (2016)
53. Xie, S., Girshick, R., Dollár, P., Tu, Z., He, K.: Aggregated residual transformations for deep neural networks. In: CVPR (2017)
54. Yang, J., Parikh, D., Batra, D.: Joint unsupervised learning of deep representations and image clusters. In: CVPR (2016)
55. Zelnik-Manor, L., Perona, P.: Self-tuning spectral clustering. In: NIPS (2005)
56. Zhai, X., Oliver, A., Kolesnikov, A., Beyer, L.: S4l: Self-supervised semi-supervised learning. In: Proceedings of the IEEE international conference on computer vision. pp. 1476–1485 (2019)
57. 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)
58. Zhang, R., Isola, P., Efros, A.A.: Colorful image colorization. In: ECCV (2016)
59. Zhao, J., Mathieu, M., Goroshin, R., Lecun, Y.: Stacked what-where auto-encoders. arXiv preprint arXiv:1506.02351 (2015)

Supplementary Material

A Smaller datasets

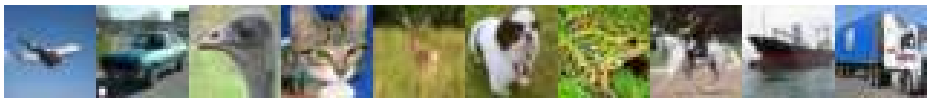
We include additional qualitative results on the smaller datasets, i.e. CIFAR10 [27], CIFAR100-20 [27] and STL10 [9]. We used the models from the state-of-the-art comparison.

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. *bus* and *bicycle* belong to the *vehicles 1* ground-truth class. This behavior can be easily understood since CIFAR-20 makes use of superclasses. As a consequence, it is difficult to explain the intra-class variance from visual appearance alone. Interestingly, we can reduce this mismatch through overclustering (see Sec 3.4.).

Fig. S1: Prototype images on the smaller datasets.

(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

Fig. S2: Low confidence predictions.

(a) CIFAR10

(b) STL10

(c) CIFAR100-20

examples include objects which are: only partially visible, occluded, under bad lighting conditions, etc.

B ImageNet

B.1 Training setup

We summarize the training setup for ImageNet below.

Pretext Task Similar to our setup on the smaller datasets, we select instance discrimination as our pretext task. In particular, we use the implementation from MoCo [8]. We use a ResNet-50 model as backbone.

Clustering Step We freeze the backbone weights during the clustering step, and only train the final linear layer using the SCAN-loss. More specifically, we train ten separate linear heads in parallel. When initiating the self-labeling step, we select the head with the lowest loss to continue training. Every image is augmented using augmentations from SimCLR [7]. We reuse the entropy weight from before (5.0), and train with batches of size 512, 1024 and 1024 on the subsets of 50, 100 and 200 classes respectively. We use an SGD optimizer with momentum 0.9 and initial learning rate 5.0. The model is trained for 100 epochs. On the full ImageNet dataset, we increase the batch size and learning rate to 4096 and 30.0 respectively, and decrease the number of neighbors to 20.

Self-Labeling Step We use the strong augmentations from RandAugment to finetune the weights through self-labeling. The model weights are updated for 25 epochs using SGD with momentum 0.9. The initial learning rate is set to 0.03 and kept constant. Batches of size 512 are used. Importantly, the model weights are updated through an exponential moving average with $\alpha = 0.999$. We did not find it necessary to apply class balancing in the cross-entropy loss.

B.2 ImageNet - Subsets

Confusion matrix Figure S3 shows a confusion matrix on the ImageNet-50 dataset. 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, *'Guacamole'* and *'Mashed Potato'* are both food, etc.

Prototype examples Figure S4 shows a prototype image for every cluster on the ImageNet-50 subset. This figure extends Figure 9 from the main paper. Remarkably, the vast majority of prototype images can be matched with one of the ground-truth classes.

Low confidence examples Figure S5 shows examples for which the model produces low confidence predictions on 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.

B.3 ImageNet - Full

We include additional qualitative results on the full ImageNet dataset. In particular, Figures S6, S7 and S8 show images from the validation set that were assigned to the same cluster. These can be viewed together with Figure 11 in the main paper. Additionally, we show some mistakes in Figure S9. The failure cases occur when the model focuses too much on the background, or when the network cannot easily discriminate between pairs of similarly looking images. However, in most cases, we can still attach some semantic meaning to the clusters, e.g. animals in cages, white fences.

C Experimental setup

C.1 Datasets

Different from prior work [24,6,52,54], we do not train and evaluate on the full datasets. Differently, we use the standard train-val splits to study the generalization properties of our models. Additionally, we report the mean and standard deviation on the smaller datasets. We would like to encourage future works to

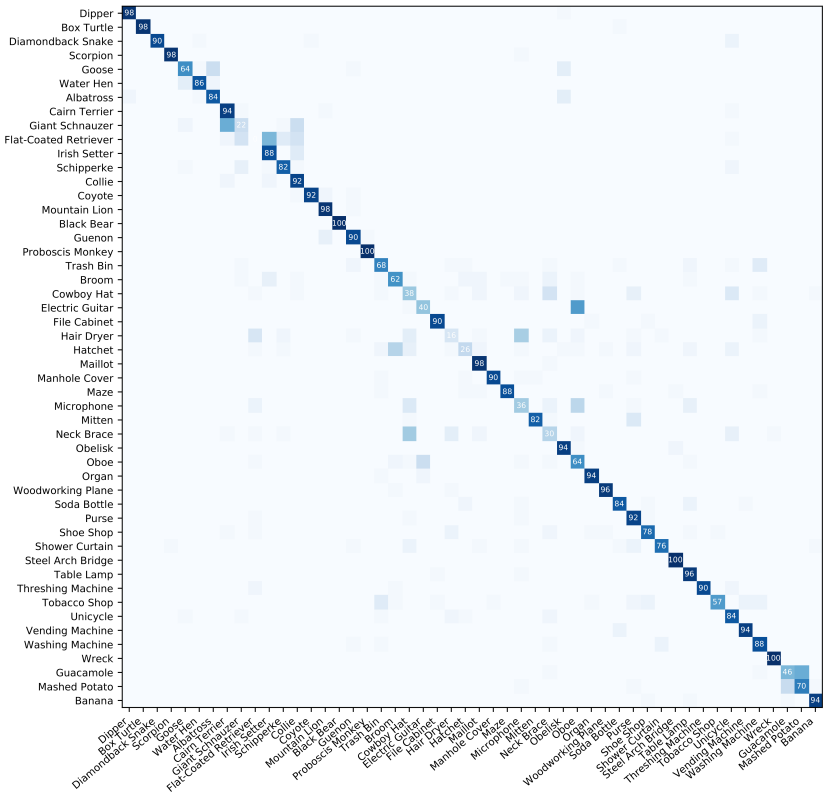


Fig. S3: Confusion matrix on ImageNet-50.

adopt this procedure as well. Table S1 provides an overview of the number of classes, the number of images and the aspect ratio of the used datasets. The selected classes on ImageNet-50, ImageNet-100 and ImageNet-200 can be found in our git repository.

Table S1: Datasets overview

Dataset	Classes	Train images	Val images	Aspect ratio
CIFAR10	10	50,000	10,000	32 x 32
CIFAR100-20	20	50,000	10,000	32 x 32
STL10	10	5,000	8,000	96 x 96
ImageNet-50	50	64,274	2,500	224 x 224
ImageNet-100	100	128,545	5,000	224 x 224
ImageNet-200	200	256,558	10,000	224 x 224
ImageNet	1000	1,281,167	50,000	224 x 224

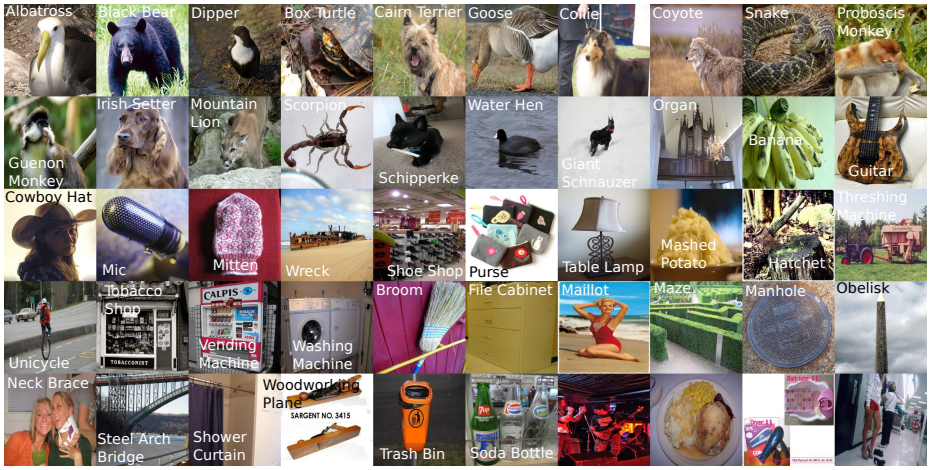


Fig. S4: Prototype images on ImageNet-50.

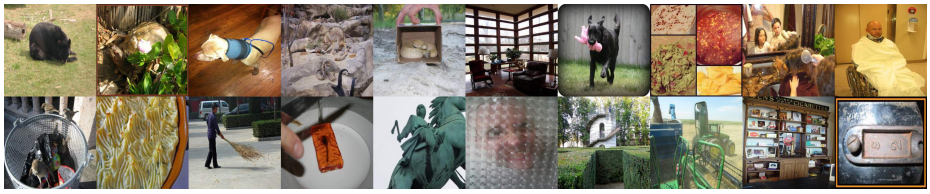


Fig. S5: Low confidence examples on ImageNet-50.

C.2 Augmentations

As shown in our experiments, it is beneficial to apply strong augmentations during training. The strong augmentations were composed of four randomly selected transformations from RandAugment [10], followed by Cutout [12]. The transformation parameters were uniformly sampled between fixed intervals. Table S2 provides a detailed overview. We applied an identical augmentation strategy across all datasets.

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

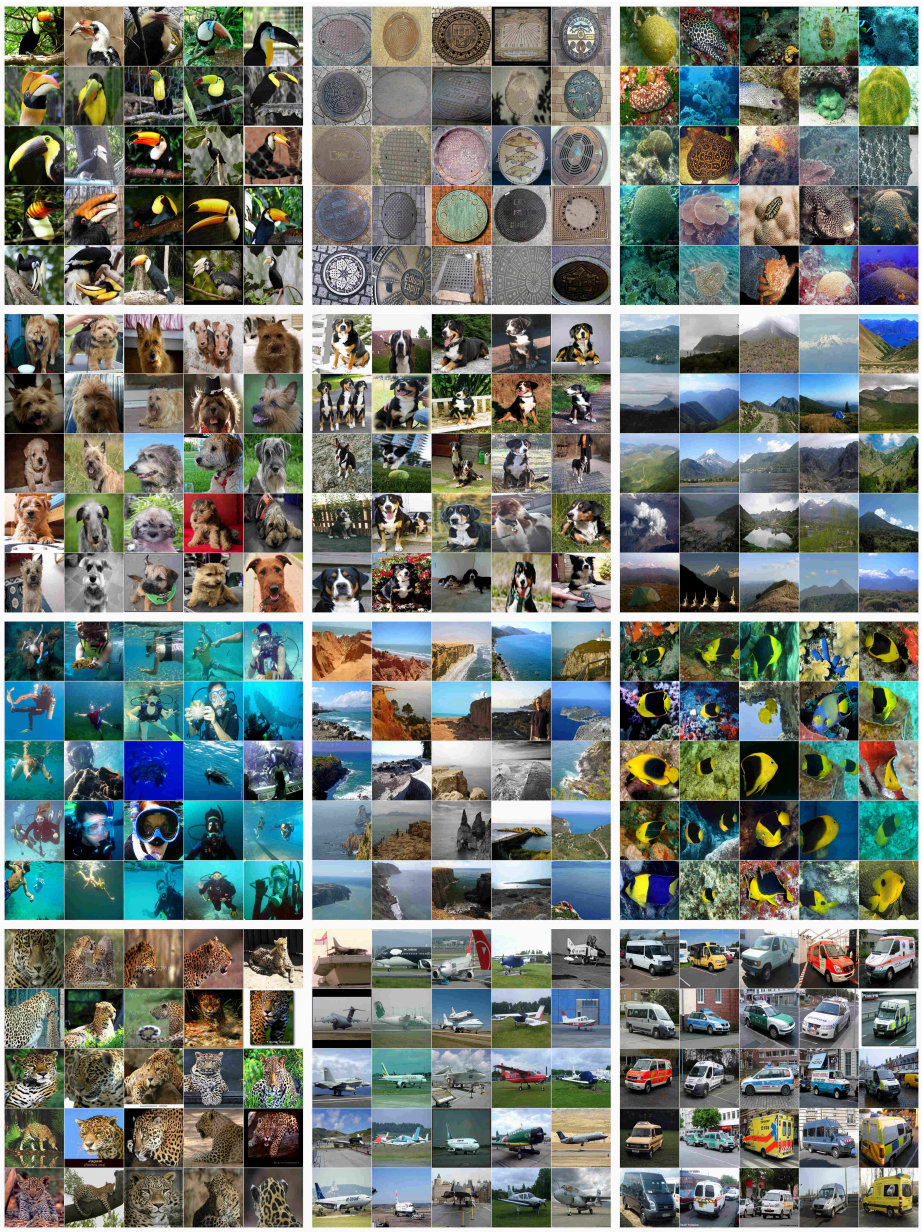


Fig. S7: Example clusters of ImageNet-1000 (2).

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

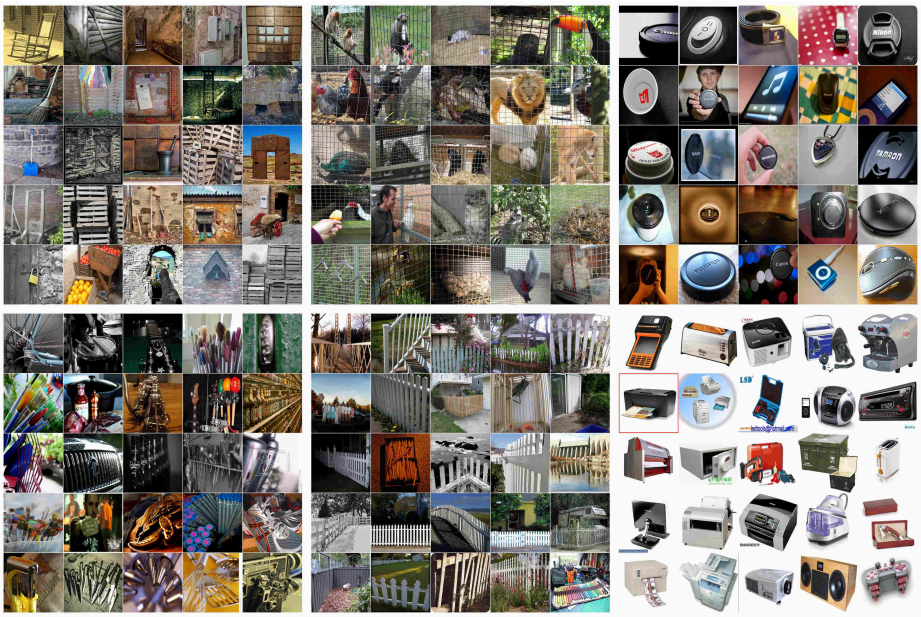


Fig. S9: Incorrect clusters of ImageNet-1000 predicted by our model.

Table S2: List of transformations. The strong transformations are composed by randomly selecting four transformations from the list, followed by Cutout.

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	$[-0.1, 0.1]$

D Change Log

The following changes were made since version 1:

- Sections 1 and 2: Minor changes were made to the text. References were added to recent related works (CMC, SimCLR, MoCo, FixMatch). Fig. 2 was updated using more recent implementations of the instance discrimination pretext task.
- Section 3.1.: The experimental setup was updated. In particular, we use the implementations from SimCLR and MoCo to perform the instance discrimination task. An identical entropy weight is now used across all datasets. We train and evaluate on the train, val split respectively, rather than using the complete dataset for both training and testing as in prior work. Doing so allows to compare the results against semi- and fully-supervised methods.
- Section 3.2.: All results were updated using the SimCLR implementation of the instance discrimination pretext task. Additional experiments were included to study the influence of applying various augmentation strategies, and to analyze the effect of the threshold value.
- Section 3.5.: We added a comparison with semi- and fully-supervised methods on ImageNet. We revised our earlier results on the smaller ImageNet subsets due to a coding mistake. Importantly, the initial conclusions remain valid, while results on the full ImageNet dataset improved. We apologize for any inconvenience this might have caused.