Unsupervised Feature Learning via Non-Parametric Instance Discrimination

Zhirong Wu[⋆†] Yuanjun Xiong[†‡] Stella X. Yu[⋆] Dahua Lin[†]
[⋆]UC Berkeley / ICSI [†]Chinese University of Hong Kong [‡]Amazon Rekognition

Abstract

Neural net classifiers trained on data with annotated class labels can also capture apparent visual similarity among categories without being directed to do so. We study whether this observation can be extended beyond the conventional domain of supervised learning: Can we learn a good feature representation that captures apparent similarity among instances, instead of classes, by merely asking the feature to be discriminative of individual instances?

We formulate this intuition as a non-parametric classification problem at the instance-level, and use noise-contrastive estimation to tackle the computational challenges imposed by the large number of instance classes.

Our experimental results demonstrate that, under unsupervised learning settings, our method surpasses the state-of-the-art on ImageNet classification by a large margin. Our method is also remarkable for consistently improving test performance with more training data and better network architectures. By fine-tuning the learned feature, we further obtain competitive results for semi-supervised learning and object detection tasks. Our non-parametric model is highly compact: With 128 *features per image, our method requires only* 600*MB storage for a million images, enabling fast nearest neighbour retrieval at the run time.*

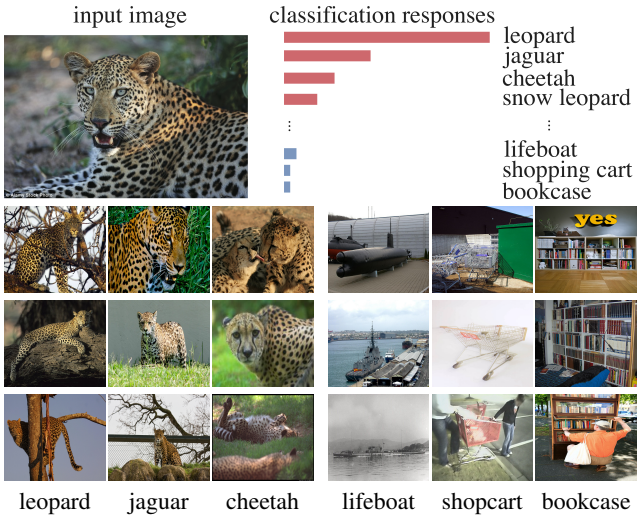

Figure 1: Supervised learning results that motivate our unsupervised approach. For an image from class *leopard*, the classes that get highest responses from a trained neural net classifier are all visually correlated, e.g., *jaguar* and *cheetah*. It is not the semantic labeling, but the apparent similarity in the data themselves that brings some classes closer than others. Our unsupervised approach takes the class-wise supervision to the extreme and learns a feature representation that discriminates among individual instances.

1. Introduction

The rise of deep neural networks, especially convolutional neural networks (CNN), has led to several breakthroughs in computer vision benchmarks. Most successful models are trained via supervised learning, which requires large datasets that are completely annotated for a specific task. However, obtaining annotated data is often very costly or even infeasible in certain cases. In recent years, unsupervised learning has received increasing attention from the community [5, 2].

Our novel approach to unsupervised learning stems from a few observations on the results of supervised learning for object recognition. On ImageNet, the top-5 classification error is significantly lower than the top-1 error [18], and the second highest responding class in the softmax output to an image is more likely to be visually correlated. Fig. 1 shows that an image from class *leopard* is rated much higher by class *jaguar* rather than by class *bookcase* [11]. Such observations reveal that a typical discriminative learning method can automatically discover apparent similarity among semantic categories, without being explicitly guided to do so. In other words, apparent similarity is learned not from semantic annotations, but from the visual data themselves.

We take the class-wise supervision to the extreme of instance-wise supervision, and ask: Can we learn a meaningful metric that reflects apparent similarity among *instances* via pure discriminative learning? An image is distinctive in its own right, and each could differ significantly from other images in the same semantic category [23]. If we learn to discriminate between individual instances, without any notion of semantic categories, we may end up with a

representation that captures apparent similarity *among instances*, just like how class-wise supervised learning still retains apparent similarity *among classes*. This formulation of unsupervised learning as an instance-level discrimination is also technically appealing, as it could benefit from latest advances in discriminative supervised learning, *e.g.* on new network architectures.

However, we also face a major challenge, now that the number of *"classes"* is the size of the entire training set. For ImageNet, it would be 1.2-million instead of 1,000 classes. Simply extending softmax to many more classes becomes infeasible. We tackle this challenge by approximating the full softmax distribution with noise-contrastive estimation (NCE) [9], and by resorting to a proximal regularization method [29] to stabilize the learning process.

To evaluate the effectiveness of unsupervised learning, past works such as [2, 31] have relied on a linear classifier, *e.g.* Support Vector Machine (SVM), to connect the learned feature to categories for classification at the test time. However, it is unclear why features learned via a training task could be *linearly* separable for an unknown testing task.

We advocate a non-parametric approach for both training and testing. We formulate instance-level discrimination as a metric learning problem, where distances (similarity) between instances are calculated directly from the features in a non-parametric way. That is, the features for each instance are stored in a discrete memory bank, rather than weights in a network. At the test time, we perform classification using *k*-nearest neighbors (kNN) based on the learned metric. Our training and testing are thus consistent, since both learning and evaluation of our model are concerned with the same metric space between images. We report and compare experimental results with both SVM and kNN accuracies.

Our experimental results demonstrate that, under unsupervised learning settings, our method surpasses the state-of-the-art on image classification by a large margin, with top-1 accuracy 42.5% on ImageNet 1K [1] and 38.7% for Places 205 [49]. Our method is also remarkable for consistently improving test performance with more training data and better network architectures. By fine-tuning the learned feature, we further obtain competitive results for semi-supervised learning and object detection tasks. Finally, our non-parametric model is highly compact: With 128 features per image, our method requires only 600MB storage for a million images, enabling fast nearest neighbour retrieval at the run time.

2. Related Works

There has been growing interest in unsupervised learning without human-provided labels. Previous works mainly fall into two categories: 1) generative models and 2) self-supervised approaches.

Generative Models. The primary objective of generative models is to reconstruct the distribution of data as faithfully as possible. Classical generative models include Restricted Bolztmann Machines (RBMs) [12, 39, 21], and Auto-encoders [40, 20]. The latent features produced by generative models could also help object recognition. Recent approaches such as generative adversarial networks [8, 4] and variational auto-encoder [14] improve both generative qualities and feature learning.

Self-supervised Learning. Self-supervised learning exploits internal structures of data and formulates *predictive* tasks to train a model. Specifically, the model needs to predict either an omitted aspect or component of an instance given the rest. To learn a representation of images, the tasks could be: predicting the context [2], counting the objects [28], filling in missing parts of an image [31], recovering colors from grayscale images [47], or even solving a jigsaw puzzle [27]. For videos, self-supervision strategies include: leveraging temporal continuity via tracking [44, 45], predicting future [42], or preserving the equivariance of egomotion [13, 50, 30]. Recent work [3] attempts to combine several self-supervised tasks to obtain better visual representations. Whereas self-supervised learning may capture relations among parts or aspects of an instance, it is unclear why a particular self supervision task should help semantic recognition and which task would be optimal.

Metric Learning. Every feature representation F induces a metric between instances x and y: $d_F(x, y) = \|F(x) - F(y)\|$. Feature learning can thus also be viewed as a certain form of metric learning. There have been extensive studies on metric learning [15, 33]. Successful application of metric learning can often result in competitive performance, *e.g.* on face recognition [35] and person re-identification [46]. In these tasks, the classes at the test time are disjoint from those at the training time. Once a network is trained, one can only infer from its feature representation, not from the subsequent linear classifier. Metric learning has been shown to be effective for few-shot learning [38, 41, 37]. An important technical point on metric learning for face recognition is normalization [35, 22, 43], which we also utilize in this work. Note that all the methods mentioned here require supervision in certain ways. Our work is drastically different: It learns the feature and thus the induced metric in an *unsupervised* fashion, without any human annotations.

Exemplar CNN. *Exemplar CNN* [5] appears similar to our work. The fundamental difference is that it adopts a parametric paradigm during both training and testing, while our method is non-parametric in nature. We study this essential difference experimentally in Sec 4.1. *Exemplar CNN* is computationally demanding for large-scale datasets such as ImageNet.

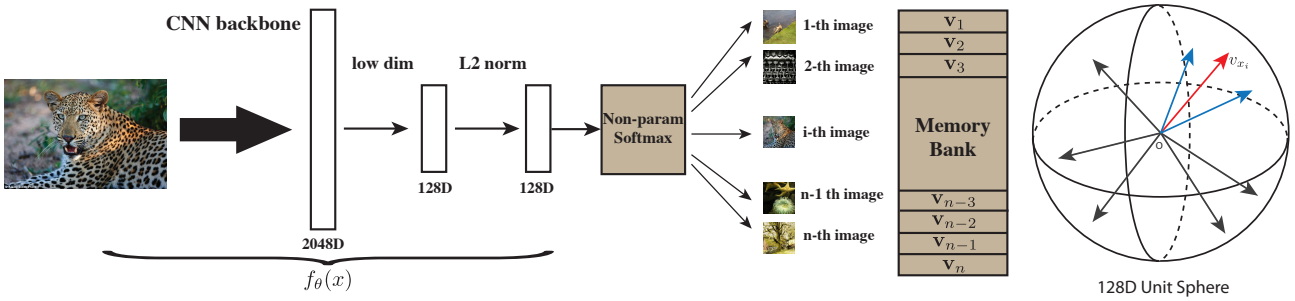

Figure 2: The pipeline of our unsupervised feature learning approach. We use a backbone CNN to encode each image as a feature vector, which is projected to a 128-dimensional space and L2 normalized. The optimal feature embedding is learned via instance-level discrimination, which tries to maximally scatter the features of training samples over the 128-dimensional unit sphere.

3. Approach

Our goal is to learn an embedding function $\mathbf{v} = f_{\boldsymbol{\theta}}(x)$ without supervision. $f_{\boldsymbol{\theta}}$ is a deep neural network with parameters $\boldsymbol{\theta}$, mapping image x to feature $\mathbf{v}$. This embedding would induces a metric over the image space, as $d_{\boldsymbol{\theta}}(x, y) = \|f_{\boldsymbol{\theta}}(x) - f_{\boldsymbol{\theta}}(y)\|$ for instances x and y. A good embedding should map visually similar images closer to each other.

Our novel unsupervised feature learning approach is *instance-level discrimination*. We treat *each image instance as a distinct class of its own* and train a classifier to distinguish between individual instance classes (Fig.2).

3.1. Non-Parametric Softmax Classifier

Parametric Classifier. We formulate the instance-level classification objective using the softmax criterion. Suppose we have n images $x_1, \ldots, x_n$ in n classes and their features $\mathbf{v}_1, \ldots, \mathbf{v}_n$ with $\mathbf{v}_i = f_{\boldsymbol{\theta}}(x_i)$. Under the conventional parametric softmax formulation, for image x with feature $\mathbf{v} = f_{\boldsymbol{\theta}}(x)$, the probability of it being recognized as i-th example is

$$P(i|\mathbf{v}) = \frac{\exp\left(\mathbf{w}_i^T \mathbf{v}\right)}{\sum_{j=1}^{n} \exp\left(\mathbf{w}_j^T \mathbf{v}\right)}. \tag{1}$$

where $\mathbf{w}_j$ is a weight vector for class j, and $\mathbf{w}_j^T \mathbf{v}$ measures how well $\mathbf{v}$ matches the j-th class *i.e.*, instance.

Non-Parametric Classifier. The problem with the parametric softmax formulation in Eq. (1) is that the weight vector $\mathbf{w}$ serves as a class prototype, preventing explicit comparisons between instances.

We propose a *non-parametric* variant of Eq. (1) that replaces $\mathbf{w}_j^T \mathbf{v}$ with $\mathbf{v}_j^T \mathbf{v}$, and we enforce $\|\mathbf{v}\| = 1$ via a L2-normalization layer. Then the probability $P(i|\mathbf{v})$ becomes:

$$P(i|\mathbf{v}) = \frac{\exp\left(\mathbf{v}_i^T \mathbf{v}/\tau\right)}{\sum_{j=1}^{n} \exp\left(\mathbf{v}_j^T \mathbf{v}/\tau\right)}, \tag{2}$$

where τ is a temperature parameter that controls the concentration level of the distribution [11]. τ is important for supervised feature learning [43], and also necessary for tuning the concentration of $\mathbf{v}$ on our unit sphere.

The learning objective is then to maximize the joint probability $\prod_{i=1}^{n} P_{\boldsymbol{\theta}}(i|f_{\boldsymbol{\theta}}(x_i))$, or equivalently to minimize the negative log-likelihood over the training set, as

$$J(\boldsymbol{\theta}) = -\sum_{i=1}^{n} \log P(i|f_{\boldsymbol{\theta}}(x_i)). \tag{3}$$

Learning with A Memory Bank. To compute the probability $P(i|\mathbf{v})$ in Eq. (2), $\{\mathbf{v}_j\}$ for all the images are needed. Instead of exhaustively computing these representations every time, we maintain a feature memory bank V for storing them [46]. In the following, we introduce separate notations for the memory bank and features forwarded from the network. Let $V = \{\mathbf{v}_j\}$ be the memory bank and $\mathbf{f}_i = f_{\boldsymbol{\theta}}(x_i)$ be the feature of x_i. During each learning iteration, the representation $\mathbf{f}_i$ as well as the network parameters θ are optimized via stochastic gradient descend. Then $\mathbf{f}_i$ is updated to V at the corresponding instance entry $\mathbf{f}_i \rightarrow \mathbf{v}_i$. We initialize all the representations in the memory bank V as unit random vectors.

Discussions. The conceptual change from class weight vector $\mathbf{w}_j$ to feature representation $\mathbf{v}_j$ directly is significant. The weight vectors $\{\mathbf{w}_j\}$ in the original softmax formulation are only valid for training classes. Consequently, they are not generalized to new classes, or in our setting, new instances. When we get rid of these weight vectors, our learning objective focuses entirely on the feature representation and its induced *metric*, which can be applied everywhere in the space and to any new instances at the test time.

Computationally, our non-parametric formulation eliminates the need for computing and storing the gradients for $\{\mathbf{w}_j\}$, making it more scalable for big data applications.

3.2. Noise-Contrastive Estimation

Computing the non-parametric softmax in Eq.(2) is cost prohibitive when the number of classes n is very large, e.g. at the scale of millions. Similar problems have been well addressed in the literature for learning word embeddings [25, 24], where the number of words can also scale to millions. Popular techniques to reduce computation include hierarchical softmax [26], noise-contrastive estimation (NCE) [9], and negative sampling [24]. We use NCE [9] to approximate the full softmax.

We adapt NCE to our problem, in order to tackle the difficulty of computing the similarity to all the instances in the training set. The basic idea is to cast the multi-class classification problem into a set of binary classification problems, where the binary classification task is to discriminate between *data samples* and *noise samples*. Specifically, the probability that feature representation $\mathbf{v}$ in the memory bank corresponds to the i-th example under our model is,

$$P(i|\mathbf{v}) = \frac{\exp(\mathbf{v}^T \mathbf{f}_i/\tau)}{Z_i} \tag{4}$$

$$Z_i = \sum_{j=1}^{n} \exp\left(\mathbf{v}_j^T \mathbf{f}_i/\tau\right) \tag{5}$$

where Z_i is the normalizing constant. We formalize the *noise distribution* as a uniform distribution: $P_n = 1/n$. Following prior work, we assume that noise samples are m times more frequent than data samples. Then the posterior probability of sample i with feature $\mathbf{v}$ being from the data distribution (denoted by $D = 1$) is:

$$h(i, \mathbf{v}) := P(D = 1|i, \mathbf{v}) = \frac{P(i|\mathbf{v})}{P(i|\mathbf{v}) + mP_n(i)}. \tag{6}$$

Our approximated training objective is to minimize the negative log-posterior distribution of data and noise samples,

$$\begin{aligned} J_{NCE}(\boldsymbol{\theta}) = &-E_{P_d}\left[\log h(i, \mathbf{v})\right] \\ &-m{\cdot}E_{P_n}\left[\log(1 - h(i, \mathbf{v}'))\right]. \end{aligned} \tag{7}$$

Here, P_d denotes the actual data distribution. For P_d, $\mathbf{v}$ is the feature corresponding to x_i; whereas for P_n, $\mathbf{v}'$ is the feature from another image, randomly sampled according to noise distribution P_n. In our model, both $\mathbf{v}$ and $\mathbf{v}'$ are sampled from the non-parametric memory bank V.

Computing normalizing constant Z_i according to Eq. (4) is expensive. We follow [25], treating it as a constant and estimating its value via Monte Carlo approximation:

$$Z \simeq Z_i \simeq nE_j\left[\exp(\mathbf{v}_j^T \mathbf{f}_i/\tau)\right] = \frac{n}{m}\sum_{k=1}^{m} \exp(\mathbf{v}_{j_k}^T \mathbf{f}_i/\tau), \tag{8}$$

where $\{j_k\}$ is a random subset of indices. Empirically, we find the approximation derived from initial batches sufficient to work well in practice.

NCE reduces the computational complexity from $O(n)$ to $O(1)$ per sample. With such drastic reduction, our experiments still yield competitive performance.

3.3. Proximal Regularization

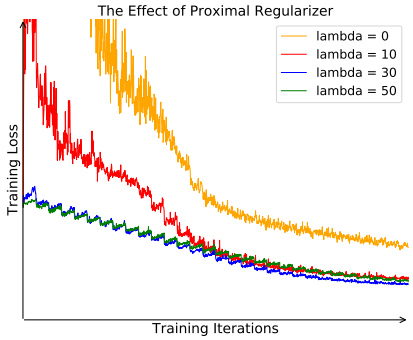

Figure 3: The effect of our proximal regularization. The original objective value oscillates a lot and converges very slowly, whereas the regularized objective has smoother learning dynamics.

Unlike typical classification settings where each class has many instances, we only have one instance per class. During each training epoch, each class is only visited once. Therefore, the learning process oscillates a lot from random sampling fluctuation. We employ the proximal optimization method [29] and introduce an additional term to encourage the smoothness of the training dynamics. At current iteration t, the feature representation for data x_i is computed from the network $\mathbf{v}_i^{(t)} = f_{\boldsymbol{\theta}}(x_i)$. The memory bank of all the representation are stored at previous iteration $V = \{\mathbf{v}^{(t-1)}\}$. The loss function for a positive sample from P_d is:

$$-\log h(i, \mathbf{v}_i^{(t-1)}) + \lambda\|\mathbf{v}_i^{(t)} - \mathbf{v}_i^{(t-1)}\|_2^2. \tag{9}$$

As learning converges, the difference between iterations, *i.e.* $\mathbf{v}_i^{(t)} - \mathbf{v}_i^{(t-1)}$, gradually vanishes, and the augmented loss is reduced to the original one. With proximal regularization, our final objective becomes:

$$\begin{aligned} J_{NCE}(\boldsymbol{\theta}) = &-E_{P_d}\left[\log h(i, \mathbf{v}_i^{(t-1)}) - \lambda\|\mathbf{v}_i^{(t)} - \mathbf{v}_i^{(t-1)}\|_2^2\right] \\ &-m{\cdot}E_{P_n}\left[\log(1 - h(i, \mathbf{v}'^{(t-1)}))\right]. \end{aligned} \tag{10}$$

Fig. 3 shows that, empirically, proximal regularization helps stabilize training, speed up convergence, and improve the learned representation, with negligible extra cost.

3.4. Weighted k-Nearest Neighbor Classifier

To classify test image $\hat{x}$, we first compute its feature $\hat{\mathbf{f}} = f_{\boldsymbol{\theta}}(\hat{x})$, and then compare it against the embeddings of all the images in the memory bank, using the cosine similarity

Training / Testing	Linear SVM	Nearest Neighbor
Param Softmax	60.3	63.0
Non-Param Softmax	75.4	**80.8**
NCE $m = 1$	44.3	42.5
NCE $m = 10$	60.2	63.4
NCE $m = 512$	64.3	78.4
NCE $m = 4096$	70.2	**80.4**

Table 1: Top-1 accuracies on CIFAR10, by applying linear SVM or kNN classifiers on the learned features. Our non-parametric softmax outperforms parametric softmax, and NCE provides close approximation as m increases.

$s_i = \cos(\mathbf{v}_i, \hat{\mathbf{f}})$. The top k nearest neighbors, denoted by $\mathcal{N}_k$, would then be used to make the prediction via weighted voting. Specifically, the class c would get a total weight $w_c = \sum_{i \in \mathcal{N}_k} \alpha_i \cdot 1(c_i = c)$. Here, α_i is the contributing weight of neighbor x_i, which depends on the similarity as $\alpha_i = \exp(s_i/\tau)$. We choose $\tau = 0.07$ as in training and we set $k = 200$.

4. Experiments

We conduct 4 sets of experiments to evaluate our approach. The first set is on CIFAR-10 to compare our non-parametric softmax with parametric softmax. The second set is on ImageNet to compare our method with other unsupervised learning methods. The last two sets of experiments investigate two different tasks, semi-supervised learning and object detection, to show the generalization ability of our learned feature representation.

4.1. Parametric vs. Non-parametric Softmax

A key novelty of our approach is the non-parametric softmax function. Compared to the conventional parametric softmax, our softmax allows a non-parametric metric to transfer to supervised tasks.

We compare both parametric and non-parametric formulations on CIFAR-10 [17], a dataset with $50,000$ training instances in 10 classes. This size allows us to compute the non-parametric softmax in Eq.(2) without any approximation. We use ResNet18 as the backbone network and its output features mapped into 128-dimensional vectors.

We evaluate the classification effectiveness based on the learned feature representation. A common practice [48, 2, 31] is to train an SVM on the learned feature over the training set, and to then classify test instances based on the feature extracted from the trained network. In addition, we also use nearest neighbor classifiers to assess the learned feature. The latter directly relies on the feature metric and may better reflect the quality of the representation.

Table 1 shows top-1 classification accuracies on CIFAR10. On the features learned with parametric softmax, we obtain accuracies of 60.3% and 63.0% with linear SVM and kNN classifiers respectively. On the features learned with non-parametric softmax, the accuracy rises to 75.4% and 80.8% for the linear and nearest neighbour classifiers, a remarkable 18% boost for the latter.

We also study the quality of NCE approximating non-parametric softmax (Sec. 3.2). The approximation is controlled by m, the number of negatives drawn for each instance. With $m = 1$, the accuracy with kNN drops significantly to 42.5%. As m increases, the performance improves steadily. When $m = 4,096$, the accuracy approaches that at $m = 49,999$ – full form evaluation without any approximation. This result provides assurance that NCE is an efficient approximation.

4.2. Image Classification

We learn a feature representation on ImageNet ILSVRC [34], and compare our method with representative unsupervised learning methods.

Experimental Settings. We choose design parameters via empirical validation. In particular, we set temperature $\tau = 0.07$ and use NCE with $m = 4,096$ to balance performance and computing cost. The model is trained for 200 epochs using SGD with momentum. The batch size is 256. The learning rate is initialized to 0.03, scaled down with coefficient 0.1 every 40 epochs after the first 120 epochs. Our code is available at: `http://github.com/zhirongw/lemniscate.pytorch`.

Comparisons. We compare our method with a randomly initialized network (as a lower bound) and various unsupervised learning methods, including self-supervised learning [2, 47, 27, 48], adversarial learning [4], and Exemplar CNN [3]. The split-brain autoencoder [48] serves a strong baseline that represents the state of the art. The results of these methods are reported with AlexNet architecture [18] in their original papers, except for exemplar CNN [5], whose results are reported with ResNet-101 [3]. As the network architecture has a big impact on the performance, we consider a few typical architectures: AlexNet [18], VGG16 [36], ResNet-18, and ResNet-50 [10].

We evaluate the performance with two different protocols: (1) Perform linear SVM on the intermediate features from `conv1` to `conv5`. Note that there are also corresponding layers in VGG16 and ResNet [36, 10]. (2) Perform kNN on the output features. Table 2 shows that:

1. With AlexNet and linear classification on intermediate features, our method achieves an accuracy of 35.6%, outperforming all baselines, including the state-of-the-art. Our method can readily scale up to deeper networks. As we move from AlexNet to ResNet-50, our accuracy is raised to 42.5%, whereas the accuracy with exemplar CNN [3] is only 31.5% even with ResNet-101.

Image Classification Accuracy on ImageNet							
method	conv1	conv2	conv3	conv4	conv5	kNN	#dim
Random	11.6	17.1	16.9	16.3	14.1	3.5	10K
Data-Init [16]	17.5	23.0	24.5	23.2	20.6	-	10K
Context [2]	16.2	23.3	30.2	31.7	29.6	-	10K
Adversarial [4]	17.7	24.5	31.0	29.9	28.0	-	10K
Color [47]	13.1	24.8	31.0	32.6	31.8	-	10K
Jigsaw [27]	19.2	30.1	34.7	33.9	28.3	-	10K
Count [28]	18.0	30.6	34.3	32.5	25.7	-	10K
SplitBrain [48]	17.7	29.3	35.4	35.2	32.8	11.8	10K
Exemplar[3]			31.5			-	4.5K
Ours Alexnet	16.8	26.5	31.8	34.1	**35.6**	31.3	128
Ours VGG16	16.5	21.4	27.6	33.1	**37.2**	33.9	128
Ours Resnet18	16.0	19.9	26.3	35.7	**42.1**	**40.5**	128
Ours Resnet50	15.3	18.8	24.4	35.3	**43.9**	**42.5**	128

Table 2: Top-1 classification accuracies on ImageNet.

Image Classification Accuracy on Places							
method	conv1	conv2	conv3	conv4	conv5	kNN	#dim
Random	15.7	20.3	19.8	19.1	17.5	3.9	10K
Data-Init [16]	21.4	26.2	27.1	26.1	24.0	-	10K
Context [2]	19.7	26.7	31.9	32.7	30.9	-	10K
Adversarial [4]	17.7	24.5	31.0	29.9	28.0	-	10K
Video [44]	20.1	28.5	29.9	29.7	27.9	-	10K
Color [47]	22.0	28.7	31.8	31.3	29.7	-	10K
Jigsaw [27]	23.0	32.1	35.5	34.8	31.3	-	10K
SplitBrain [48]	21.3	30.7	34.0	34.1	32.5	10.8	10K
Ours Alexnet	18.8	24.3	31.9	**34.5**	33.6	30.1	128
Ours VGG16	17.6	23.1	29.5	33.8	**36.3**	32.8	128
Ours Resnet18	17.8	23.0	30.3	34.2	**41.3**	**36.7**	128
Ours Resnet50	18.1	22.3	29.7	34.1	**42.1**	**38.7**	128

Table 3: Top-1 classification accuracies on Places, based directly on features learned on ImageNet, without any fine-tuning.

2. Using nearest neighbor classification on the final 128 dimensional features, our method achieves 31.3%, 33.9%, 40.5% and 42.5% accuracies with AlexNet, VGG16, ResNet-18 and ResNet-50, not much lower than the linear classification results, demonstrating that our learned feature induces a reasonably good metric. As a comparison, for Split-brain, the accuracy drops to 8.9% with nearest neighbor classification on `conv3` features, and to 11.8% after projecting the features to 128 dimensions.
3. With our method, the performance gradually increases as we examine the learned feature representation from earlier to later layers, which is generally desirable. With all other methods, the performance decreases beyond `conv3` or `conv4`.
4. It is important to note that the features from intermediate convolutional layers can be over $10,000$ dimensions. Hence, for other methods, using the features from the *best-performing* layers can incur significant storage and computation costs. Our method produces a 128-dimensional representation at the last layer, which is very efficient to work with. The encoded features of all $1.28M$ images in ImageNet only take about 600 MB of storage. Exhaustive nearest neighbor search over this dataset only takes 20 ms per image on a Titan X GPU.

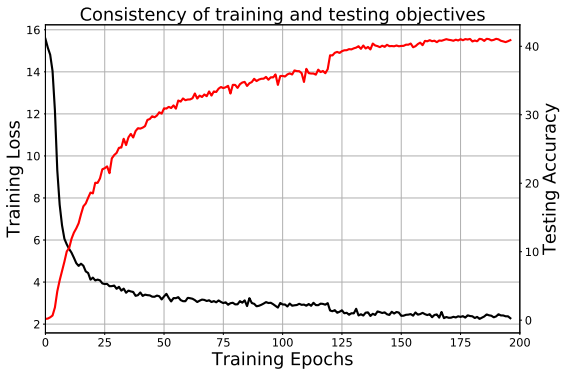

Figure 4: Our kNN testing accuracy on ImageNet continues to improve as the training loss decreases, demonstrating that our unsupervised learning objective captures apparent similarity which aligns well with the semantic annotation of the data.

Feature generalization. We also study how the learned feature representations can generalize to other datasets. With the same settings, we conduct another large-scale experiment on *Places* [49], a large dataset for scene classification, which contains $2.45M$ training images in 205 categories. In this experiment, we directly use the feature extraction networks trained on ImageNet *without finetuning*. Table 3 compares the results obtained with different methods and under different evaluation policies. Again, with linear classifier on `conv5` features, our method achieves competitive performance of top-1 accuracy 34.5% with AlexNet, and 42.1% with ResNet-50. With nearest neighbors on the last layer which is much smaller than intermediate layers, we achieve an accuracy of 38.7% with ResNet-50. These results show remarkable generalization ability of the representations learned using our method.

Consistency of training and testing objectives. Unsupervised feature learning is difficult because the training objective is agnostic about the testing objective. A good training objective should be reflected in consistent improvement in the testing performance. We investigate the relation between the training loss and the testing accuracy across iterations. Fig. 4 shows that our testing accuracy continues to improve as training proceeds, with no sign of overfitting. It also suggests that better optimization of the training objective may further improve our testing accuracy.

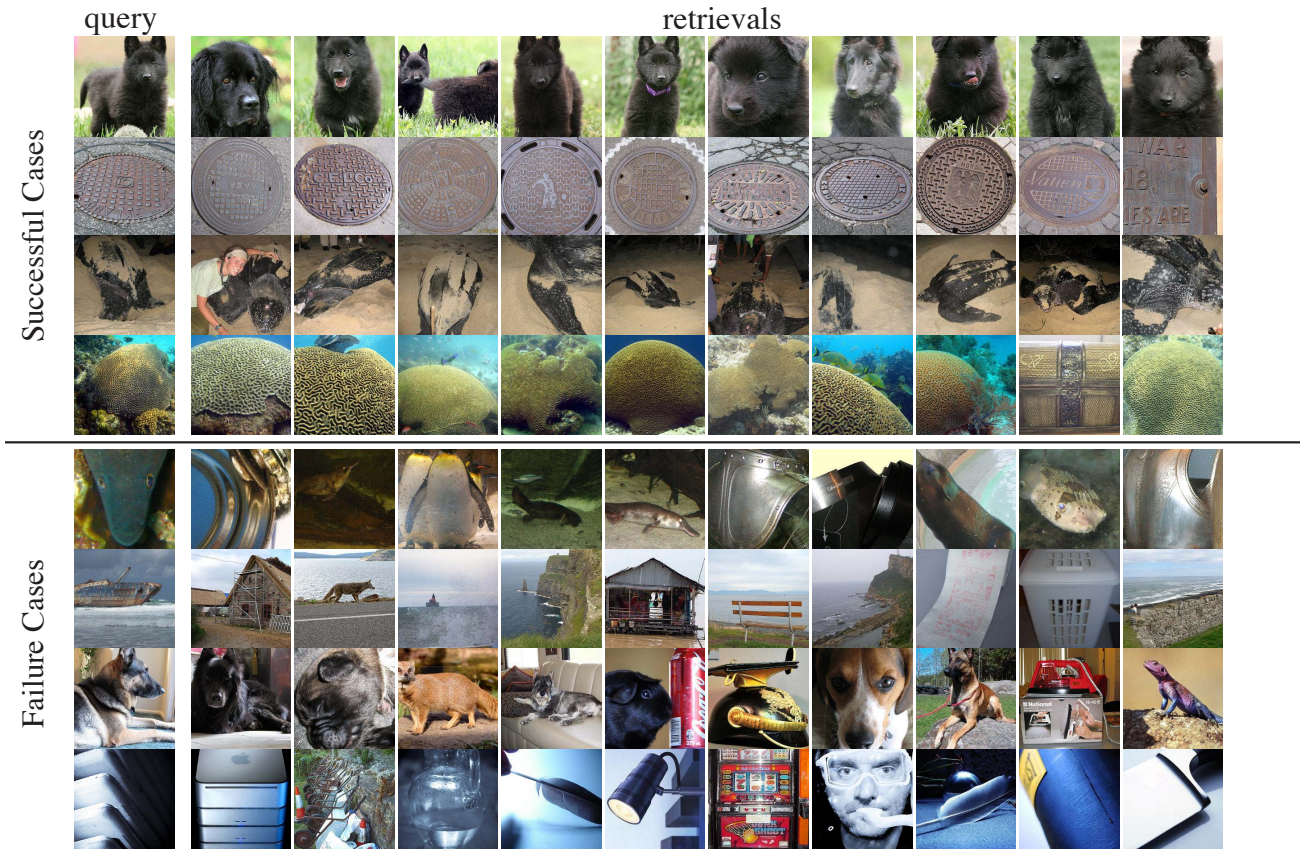

Figure 5: Retrieval results for example queries. The left column are queries from the validation set, while the right columns show the 10 closest instances from the training set. The upper half shows the best cases. The lower half shows the worst cases.

embedding size	32	64	128	256
top-1 accuracy	34.0	38.8	40.5	40.1

Table 4: Classification performance on ImageNet with ResNet18 for different embedding feature sizes.

training set size	0.1%	1%	10%	30%	100%
accuracy	3.9	10.7	23.1	31.7	40.5

Table 5: Classification performances trained on different amount of training set with ResNet-18.

The embedding feature size. We study how the performance changes as we vary the embedding size from 32 to 256. Table 4 shows that the performance increases from 32, plateaus at 128, and appears to saturate towards 256.

Training set size. To study how our method scales with the data size, we train different representations with various proportions of ImageNet data, and evaluate the classification performance on the full labeled set using nearest neighbors. Table 5 shows that our feature learning method benefits from larger training sets, and the testing accuracy improves as the training set grows. This property is crucial for successful unsupervised learning, as there is no shortage of unlabeled data in the wild.

Qualitative case study. To illustrate the learned features, Figure 5 shows the results of image retrieval using the learned features. The upper four rows show the best cases where all top 10 results are in the same categories as the queries. The lower four rows show the worst cases where none of the top 10 are in the same categories. However, even for the failure cases, the retrieved images are still visually similar to the queries, a testament to the power of our unsupervised learning objective.

4.3. Semi-supervised Learning

We now study how the learned feature extraction network can benefit other tasks, and whether it can provide a good basis for transfer learning to other tasks. A common scenario that can benefit from unsupervised learning is when we have a large amount of data of which only a small fraction are labeled. A natural semi-supervised learning approach is to first learn from the big unlabeled data and then fine-tune the model on the small labeled data.

We randomly choose a subset of ImageNet as labeled

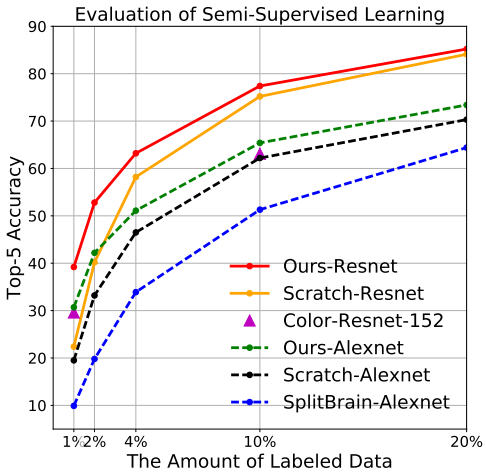

Figure 6: Semi-supervised learning results on ImageNet with an increasing fraction of labeled data (x axis). Ours are consistently and significantly better. Note that the results for colorization-based pretraining are from a deeper ResNet-152 network [19].

and treat others as unlabeled. We perform the above semi-supervised learning and measure the classification accuracy on the validation set. In order to compare with [19], we report the top-5 accuracy here.

We compare our method with three baselines: (1) *Scratch*, *i.e*. fully supervised training on the small labeled subsets, (2) *Split-brain* [48] for pre-training, and (3) *Colorization* [19] for pre-training. Finetuning on the labeled subset takes 70 epochs with initial learning rate 0.01 and a decay rate of 10 every 30 epochs. We vary the proportion of labeled subset from 1% to 20% of the entire dataset.

Fig. 6 shows that our method significantly outperforms all other approaches, and ours is the only one outperforming supervised learning from limited labeled data. When only 1% of data is labeled, we outperform by a large 10% margin, demonstrating that our feature learned from unlabeled data is effective for task adaptation.

4.4. Object Detection

To further assess the generalization capacity of the learned features, we transfer the learned networks to the new task of object detection on PASCAL VOC 2007 [6]. Training object detection model from scratch is often difficult, and a prevalent practice is to pretrain the underlying CNN on ImageNet and fine-tune it for the detection task.

We experiment with Fast R-CNN [7] with AlexNet and VGG16 architectures, and Faster R-CNN [32] with ResNet-50. When fine-tuning Fast R-CNN, the learning rate is initialized to 0.001 and scaled down by 10 times after every $50K$ iterations. When fine-tuning AlexNet and VGG16, we follow the standard practice, fixing the conv1 model weights. When fine-tuning Faster R-CNN, we fix the model weights below the 3rd type of residual blocks, only updating the layers above and freezing all batch normalization layers. We follow the standard pipeline for finetuning and do not use the rescaling method proposed in [2]. We use the standard trainval set in VOC 2007 for training and testing.

Method	mAP	Method	mAP
AlexNet Labels†	56.8	VGG Labels†	67.3
Gaussian	43.4	Gaussian	39.7
Data-Init [16]	45.6	Video [44]	60.2
Context [2]	**51.1**	Context [2]	61.5
Adversarial [4]	46.9	Transitivity [45]	**63.2**
Color [47]	46.9	Ours VGG	60.5
Video [44]	47.4	ResNet Labels†	76.2
Ours Alexnet	48.1	Ours ResNet	**65.4**

Table 6: Object detection performance on PASCAL VOC 2007 test, in terms of mean average precision (mAP), for supervised pretraining methods (marked by †), existing unsupervised methods, and our method.

We compare three settings: 1) directly training from scratch (lower bound), 2) pretraining on ImageNet in a supervised way (upper bound), and 3) pretraining on ImageNet or other data using various unsupervised methods.

Table 6 lists detection performance in terms of mean average precision (mAP). With AlexNet and VGG16, our method achieves an mAP of 48.1% and 60.5%, on par with the state-of-the-art unsupervised methods. With Resnet-50, our method achieves an mAP of 65.4%, surpassing all existing unsupervised learning approaches. It also shows that our method scales well as the network gets deeper. There remains a significant gap of 11% to be narrowed towards mAP 76.2% from supervised pretraining.

5. Summary

We present an unsupervised feature learning approach by maximizing distinction between instances via a novel non-parametric softmax formulation. It is motivated by the observation that supervised learning results in apparent image similarity. Our experimental results show that our method outperforms the state-of-the-art on image classification on ImageNet and Places, with a compact 128-dimensional representation that scales well with more data and deeper networks. It also delivers competitive generalization results on semi-supervised learning and object detection tasks.

Acknowledgements. This work was supported in part by Berkeley Deep Drive, Big Data Collaboration Research grant from SenseTime Group (CUHK Agreement No. TS1610626), and the General Research Fund (GRF) of Hong Kong (No. 14236516).

References

[1] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*. IEEE, 2009. 2

[2] C. Doersch, A. Gupta, and A. A. Efros. Unsupervised visual representation learning by context prediction. In *ICCV*, 2015. 1, 2, 5, 6, 8

[3] C. Doersch and A. Zisserman. Multi-task self-supervised visual learning. *arXiv preprint arXiv:1708.07860*, 2017. 2, 5, 6

[4] J. Donahue, P. Krähenbühl, and T. Darrell. Adversarial feature learning. *arXiv preprint arXiv:1605.09782*, 2016. 2, 5, 6, 8

[5] A. Dosovitskiy, J. T. Springenberg, M. Riedmiller, and T. Brox. Discriminative unsupervised feature learning with convolutional neural networks. In *NIPS*, 2014. 1, 2, 5

[6] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *IJCV*, 2010. 8

[7] R. Girshick. Fast r-cnn. In *ICCV*, 2015. 8

[8] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *NIPS*, 2014. 2

[9] M. Gutmann and A. Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *AISTATS*, 2010. 2, 4

[10] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*, 2015. 5

[11] G. Hinton, O. Vinyals, and J. Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 1, 3

[12] G. E. Hinton, S. Osindero, and Y.-W. Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 2006. 2

[13] D. Jayaraman and K. Grauman. Learning image representations tied to egomotion from unlabeled video. *IJCV*, 2017. 2

[14] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 2

[15] M. Koestinger, M. Hirzer, P. Wohlhart, P. M. Roth, and H. Bischof. Large scale metric learning from equivalence constraints. In *CVPR*. IEEE, 2012. 2

[16] P. Krähenbühl, C. Doersch, J. Donahue, and T. Darrell. Data-dependent initializations of convolutional neural networks. *arXiv preprint arXiv:1511.06856*, 2015. 6, 8

[17] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. 2009. 5

[18] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012. 1, 5

[19] G. Larsson, M. Maire, and G. Shakhnarovich. Colorization as a proxy task for visual understanding. CVPR, 2017. 8

[20] Q. V. Le. Building high-level features using large scale unsupervised learning. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE, 2013. 2

[21] H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *Proceedings of the 26th annual international conference on machine learning*. ACM, 2009. 2

[22] W. Liu, Y. Wen, Z. Yu, M. Li, B. Raj, and L. Song. Sphereface: Deep hypersphere embedding for face recognition. In *CVPR*, 2017. 2

[23] T. Malisiewicz, A. Gupta, and A. A. Efros. Ensemble of exemplar-svms for object detection and beyond. In *ICCV*. IEEE, 2011. 1

[24] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *NIPS*, 2013. 4

[25] A. Mnih and K. Kavukcuoglu. Learning word embeddings efficiently with noise-contrastive estimation. In *NIPS*, 2013. 4

[26] F. Morin and Y. Bengio. Hierarchical probabilistic neural network language model. In *Aistats*, volume 5. Citeseer, 2005. 4

[27] M. Noroozi and P. Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *ECCV*. Springer, 2016. 2, 5, 6

[28] M. Noroozi, H. Pirsiavash, and P. Favaro. Representation learning by learning to count. *arXiv preprint arXiv:1708.06734*, 2017. 2, 6

[29] N. Parikh, S. Boyd, et al. Proximal algorithms. *Foundations and Trends® in Optimization*, 2014. 2, 4

[30] D. Pathak, R. Girshick, P. Dollár, T. Darrell, and B. Hariharan. Learning features by watching objects move. *arXiv preprint arXiv:1612.06370*, 2016. 2

[31] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros. Context encoders: Feature learning by inpainting. In *CVPR*, 2016. 2, 5

[32] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NIPS*, 2015. 8

[33] S. Roweis, G. Hinton, and R. Salakhutdinov. Neighbourhood component analysis. *Adv. Neural Inf. Process. Syst.(NIPS)*, 17, 2004. 2

[34] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge. *IJCV*, 2015. 5

[35] F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. In *CVPR*, 2015. 2

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

[37] J. Snell, K. Swersky, and R. S. Zemel. Prototypical networks for few-shot learning. *arXiv preprint arXiv:1703.05175*, 2017. 2

[38] K. Sohn. Improved deep metric learning with multi-class n-pair loss objective. In *NIPS*, 2016. 2

[39] Y. Tang, R. Salakhutdinov, and G. Hinton. Robust boltzmann machines for recognition and denoising. In *CVPR*. IEEE, 2012. 2

[40] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*. ACM, 2008. 2

[41] O. Vinyals, C. Blundell, T. Lillicrap, D. Wierstra, et al.

Matching networks for one shot learning. In *NIPS*, 2016. 2

[42] J. Walker, C. Doersch, A. Gupta, and M. Hebert. An uncertain future: Forecasting from static images using variational autoencoders. In *ECCV*. Springer, 2016. 2

[43] F. Wang, X. Xiang, J. Cheng, and A. L. Yuille. Normface: l_2 hypersphere embedding for face verification. *arXiv preprint arXiv:1704.06369*, 2017. 2, 3

[44] X. Wang and A. Gupta. Unsupervised learning of visual representations using videos. In *ICCV*, 2015. 2, 6, 8

[45] X. Wang, K. He, and A. Gupta. Transitive invariance for self-supervised visual representation learning. *arXiv preprint arXiv:1708.02901*, 2017. 2, 8

[46] T. Xiao, S. Li, B. Wang, L. Lin, and X. Wang. Joint detection and identification feature learning for person search. CVPR, 2017. 2, 3

[47] R. Zhang, P. Isola, and A. A. Efros. Colorful image colorization. *ECCV*, 2016. 2, 5, 6, 8

[48] R. Zhang, P. Isola, and A. A. Efros. Split-brain autoencoders: Unsupervised learning by cross-channel prediction. *CVPR*, 2017. 5, 6, 8

[49] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva. Learning deep features for scene recognition using places database. In *NIPS*, 2014. 2, 6

[50] T. Zhou, M. Brown, N. Snavely, and D. G. Lowe. Unsupervised learning of depth and ego-motion from video. *arXiv preprint arXiv:1704.07813*, 2017. 2