Unsupervised Learning by Predicting Noise

Piotr Bojanowski[1] **Armand Joulin**[1]

Abstract

Convolutional neural networks provide visual features that perform well in many computer vision applications. However, training these networks requires large amounts of supervision; this paper introduces a generic framework to train such networks, end-to-end, with no supervision. We propose to fix a set of target representations, called *Noise As Targets* (NAT), and to constrain the deep features to align to them. This domain agnostic approach avoids the standard unsupervised learning issues of trivial solutions and collapsing of features. Thanks to a stochastic batch reassignment strategy and a separable square loss function, it scales to millions of images. The proposed approach produces representations that perform on par with state-of-the-art unsupervised methods on ImageNet and PASCAL VOC.

1. Introduction

In recent years, convolutional neural networks, or *convnets* (Fukushima, 1980; LeCun et al., 1989) have pushed the limits of computer vision (Krizhevsky et al., 2012; He et al., 2016), leading to important progress in a variety of tasks, like object detection (Girshick, 2015) or image segmentation (Pinheiro et al., 2015). Key to this success is their ability to produce features that easily transfer to new domains when trained on massive databases of labeled images (Razavian et al., 2014; Oquab et al., 2014) or weakly-supervised data (Joulin et al., 2016). However, human annotations may introduce unforeseen bias that could limit the potential of learned features to capture subtle information hidden in a vast collection of images.

Several strategies exist to learn deep convolutional features with no annotation (Donahue et al., 2016). They either try to capture a signal from the source as a form of *self-supervision* (Doersch et al., 2015; Wang & Gupta, 2015) or learn the underlying distribution of images (Vincent et al., 2010; Goodfellow et al., 2014). While some of these approaches obtain promising performance in transfer learning (Donahue et al., 2016; Wang & Gupta, 2015), they do not explicitly aim to learn discriminative features. Some attempts were made with retrieval based approaches (Dosovitskiy et al., 2014) and clustering (Yang et al., 2016; Liao et al., 2016), but they are hard to scale and have only been tested on small datasets. Unfortunately, as in the supervised case, a lot of data is required to learn good representations.

In this work, we propose a discriminative framework designed to learn deep architectures on large datasets. Our approach is general, but we focus on convnets since they require millions of images to produce good features. Similar to self-organizing maps (Kohonen, 1982; Martinetz & Schulten, 1991), we map deep features to a set of predefined representations in a low dimensional space. As opposed to these approaches, we aim to learn the features in a end-to-end fashion, which traditionally suffers from a feature collapsing problem. Our approach deals with this issue by fixing the target representations and aligning them to our features. These representations are sampled from a uninformative distribution and we use this *Noise As Targets* (NAT). Our approach also shares some similarities with standard clustering approches like k-means (Lloyd, 1982) or discriminative clustering (Bach & Harchaoui, 2007).

In addition, we propose an online algorithm able to scale to massive image databases like ImageNet (Deng et al., 2009). Importantly, our approach is barely less efficient to train than standard supervised approaches and can reuse any optimization procedure designed for them. This is achieved by using a quadratic loss as in (Tygert et al., 2017) and a fast approximation of the Hungarian algorithm. We show the potential of our approach by training end-to-end on ImageNet a standard architecture, namely AlexNet (Krizhevsky et al., 2012) with no supervision.

We test the quality of our features on several image classification problems, following the setting of Donahue et al. (2016). We are on par with state-of-the-art unsupervised and self-supervised learning approaches while being much simpler to train and to scale.

The paper is organized as follows: after a brief review of the related work in Section 2, we present our approach in

[1] Facebook AI Research. Correspondence to: Piotr Bojanowski <bojanowski@fb.com>.

Proceedings of the 34^{th} International Conference on Machine Learning, Sydney, Australia, PMLR 70, 2017. Copyright 2017 by the author(s).

Section 3. We then validate our solution with several experiments and comparisons with standard unsupervised and self-supervised approaches in Section 4.

2. Related work

Several approaches have been recently proposed to tackle the problem of deep unsupervised learning (Coates & Ng, 2012; Mairal et al., 2014; Dosovitskiy et al., 2014). Some of them are based on a clustering loss (Xie et al., 2016; Yang et al., 2016; Liao et al., 2016), but they are not tested at a scale comparable to that of supervised convnet training. Coates & Ng (2012) uses k-means to pre-train convnets, by learning each layer sequentially in a bottom-up fashion. In our work, we train the convnet end-to-end with a loss that shares similarities with k-means. Closer to our work, Dosovitskiy et al. (2014) proposes to train convnets by solving a retrieval problem. They assign a class per image and its transformation. In contrast to our work, this approach can hardly scale to more than a few hundred of thousands of images, and requires a custom-tailored architecture while we use a standard AlexNet.

Another traditional approach for learning visual representations in an unsupervised manner is to define a parametrized mapping between a predefined random variable and a set of images. Traditional examples of this approach are variational autoencoders (Kingma & Welling, 2013), generative adversarial networks (Goodfellow et al., 2014), and to a lesser extent, noisy autoencoders (Vincent et al., 2010). In our work, we are doing the opposite; that is, we map images to a predefined random variable. This allows us to re-use standard convolutional networks and greatly simplifies the training.

Generative adversarial networks. Among those approaches, generative adversarial networks (GANs) (Goodfellow et al., 2014; Denton et al., 2015; Donahue et al., 2016) share another similarity with our approach, namely they are explicitly minimizing a discriminative loss to learn their features. While these models cannot learn an inverse mapping, Donahue et al. (2016) recently proposed to add an encoder to extract visual features from GANs. Like ours, their encoder can be any standard convolutional network. However, their loss aims at differentiating real and generated images, while we are aiming directly at differentiating between images. This makes our approach much simpler and faster to train, since we do not need to learn the generator nor the discriminator.

Self-supervision. Recently, a lot of work has explored *self-supervison*: leveraging supervision contained in the input signal (Doersch et al., 2015; Noroozi & Favaro, 2016; Pathak et al., 2016). In the same vein as word2vec (Mikolov et al., 2013), Doersch et al. (2015) show that spatial context is a strong signal to learn visual features. Noroozi & Favaro (2016) have further extended this work. Others have shown that temporal coherence in videos also provides a signal that can be used to learn powerful visual features (Agrawal et al., 2015; Jayaraman & Grauman, 2015; Wang & Gupta, 2015). In particular, Wang & Gupta (2015) show that such features provide promising performance on ImageNet. In contrast to our work, these approaches are domain dependent since they require explicit derivation of weak supervision directly from the input.

Autoencoders. Many have also used autoencoders with a reconstruction loss (Bengio et al., 2007; Ranzato et al., 2007; Masci et al., 2011). The idea is to encode and decode an image, while minimizing the loss between the decoded and original images. Once trained, the encoder produces image features and the decoder can be used to generate images from codes. The decoder is often a fully connected network (Ranzato et al., 2007) or a deconvolutional network (Masci et al., 2011; Zhao et al., 2016) but can be more sophisticated, like a PixelCNN network (van den Oord et al., 2016).

Self-organizing map. This family of unsupervised methods aims at learning a low dimensional representation of the data that preserves certain topological properties (Kohonen, 1982; Vesanto & Alhoniemi, 2000). In particular, Neural Gas (Martinetz & Schulten, 1991) aligns feature vectors to the input data. Each input datum is then assigned to one of these vectors in a winner-takes-all manner. These feature vectors are in spirit similar to our target representations and we use a similar assignment strategy. In contrast to our work, the target vectors are not fixed and aligned to the input vectors. Since we primarly aim at learning the input features, we do the opposite.

Discriminative clustering. Many methods have been proposed to use discriminative losses for clustering (Xu et al., 2004; Bach & Harchaoui, 2007; Krause et al., 2010; Joulin & Bach, 2012). In particular, Bach & Harchaoui (2007) shows that the ridge regression loss could be use to learn discriminative clusters. It has been successfully applied to several computer vision applications, like object discovery (Joulin et al., 2010; Tang et al., 2014) or video/text alignment (Bojanowski et al., 2013; 2014; Ramanathan et al., 2014). In this work, we show that a similar framework can be designed for neural networks. As opposed to Xu et al. (2004), we address the empty assignment problems by restricting the set of possible reassignments to permutations rather than using global linear constrains the assignments. Our assignments can be updated online, allowing our approach to scale to very large datasets.

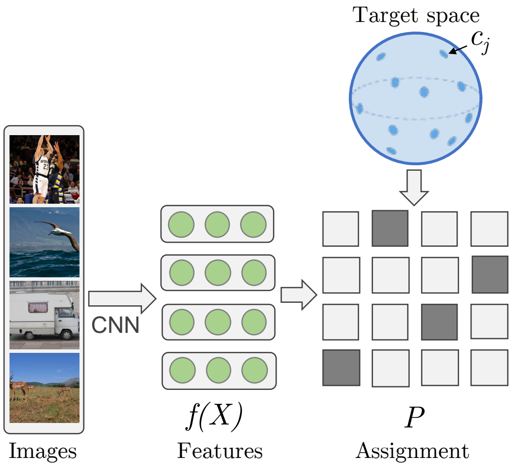

Figure 1. Our approach takes a set of images, computes their deep features with a convolutional network and matches them to a set of predefined targets from a low dimensional space. The parameters of the network are learned by aligning the features to the targets.

3. Method

In this section, we present our model and discuss its relations with several clustering approaches including k-means. Figure 1 shows an overview of our approach. We also show that it can be trained on massive datasets using an online procedure. Finally, we provide all the implementation details.

3.1. Unsupervised learning

We are interested in learning visual features with no supervision. These features are produced by applying a parametrized mapping f_θ to the images. In the presence of supervision, the parameters θ are learned by minimizing a loss function between the features produced by this mapping and some given targets, e.g., labels. In absence of supervision, there is no clear target representations and we thus need to learn them as well. More precisely, given a set of n images x_i, we jointly learn the parameters θ of the mapping f_θ, and some target vectors y_i:

$$\min_\theta \frac{1}{n} \sum_{i=1}^{n} \min_{y_i \in \mathbb{R}^d} \ell(f_\theta(x_i), y_i), \tag{1}$$

where d is the dimension of target vectors. In the rest of the paper, we use matrix notations, *i.e.*, we denote by Y the matrix whose rows are the target representations y_i, and by X the matrix whose rows are the images x_i. With a slight abuse of notation, we denote by $f_\theta(X)$ the $n \times d$ matrix of features whose rows are obtained by applying the function f_θ to each image independently.

Choosing the loss function. In the supervised setting, a popular choice for the loss ℓ is the softmax function. However, computing this loss is linear in the number of targets, making it impractical for large output spaces (Goodman, 2001). While there are workarounds to scale these losses to large output spaces, Tygert et al. (2017) has recently shown that using a squared ℓ_2 distance works well in many supervised settings, as long as the final activations are unit normalized. This loss only requires access to a single target per sample, making its computation independent of the number of targets. This leads to the following problem:

$$\min_\theta \min_{Y \in \mathbb{R}^{n \times d}} \frac{1}{2n} \| f_\theta(X) - Y \|_F^2, \tag{2}$$

where we still denote by $f_\theta(X)$ the unit normalized features.

Using fixed target representations. Directly solving the problem defined in Eq. (2) would lead to a representation collapsing problem: all the images would be assigned to the same representation (Xu et al., 2004). We avoid this issue by fixing a set of k predefined target representations and matching them to the visual features. More precisely, the matrix Y is defined as the product of a matrix C containing these k representations and an assignment matrix P in $\{0,1\}^{n \times k}$, *i.e.*,

$$Y = PC. \tag{3}$$

Note that we can assume that k is greater than n with no loss of generality (by duplicating representations otherwise). Each image is assigned to a different target and each target can only be assigned once. This leads to a set $\mathcal{P}$ of constraints for the assignment matrices:

$$\mathcal{P} = \{ P \in \{0,1\}^{n \times k} \mid P 1_k \leq 1_n, \ P^\top 1_n = 1_k \}. \tag{4}$$

This formulation forces the visual features to be diversified, avoiding the collapsing issue at the cost of fixing the target representations. Predefining these targets is an issue if their number k is small, which is why we are interested in the case where k is at least as large as the number n of images.

Choosing the target representations. Until now, we have not discussed the set of target representations stored in C. A simple choice for the targets would be to take k elements of the canonical basis of $\mathbb{R}^d$. If d is larger than n, this formulation would be similar to the framework of Dosovitskiy et al. (2014), and is impractical for large n. On the other hand, if d is smaller than n, this formulation is equivalent to the discriminative clustering approach of Bach & Harchaoui (2007). Choosing such targets makes very strong assumptions on the nature of the underlying problem. Indeed, it assumes that each image belongs to a unique class and that all classes are orthogonal. While this assumption might be true for some classification datasets, it does not generalize to large image collections nor capture subtle similarities between images belonging to different classes.

Since our features are unit normalized, another natural choice is to uniformly sample target vectors on the ℓ_2 unit sphere. Note that the dimension d will then directly influence the level of correlation between representations, *i.e.*, the correlation is inversely proportional to the square root of d. Using this *Noise As Targets* (NAT), Eq. (2) is now equivalent to:

$$\max_{\theta} \max_{P \in \mathcal{P}} \operatorname{Tr}\left(PCf_{\theta}(X)^{\top}\right). \tag{5}$$

This problem can be interpreted as mapping deep features to a uniform distribution over a manifold, namely the d-dimension ℓ_2 sphere. Using k predefined representations is a discrete approximation of this manifold that justifies the restriction of the mapping matrices to the set $\mathcal{P}$ of 1-to-1 assignment matrices. In some sense, we are optimizing a crude approximation of the earth mover's distance between the distribution of deep features and a given target distribution (Rubner et al., 1998).

Relation to clustering approaches. Using the same notations as in Eq. (5), several clustering approaches share similarities with our method. In the linear case, spherical k-means minimizes the same loss function w.r.t. P and C, *i.e.*,

$$\max_{C} \max_{P \in \mathcal{Q}} \operatorname{tr}\left(PCX^{T}\right).$$

The main difference is the set $\mathcal{Q}$ of assignment matrices:

$$\mathcal{Q} = \{P \in \{0,1\}^{n \times k} \mid P 1_k = 1_n\}.$$

This set only guarantees that each data point is assigned to a single target representation. Once we jointly learn the features and the assignment, this set does not prevent the collapsing of the data points to a single target representation.

Another similar clustering approach is Diffrac (Bach & Harchaoui, 2007). Their loss is equivalent to ours in the case of unit normalized features. Their set $\mathcal{R}$ of assignment matrices, however, is different:

$$\mathcal{R} = \{P \in \{0,1\}^{n \times k} \mid P^{\top} 1_n \geq c 1_k\},$$

where $c > 0$ is some fixed parameter. While restricting the assignment matrices to this set prevents the collapsing issue, it introduces global constraints that are not suited for online optimization. This makes their approach hard to scale to large datasets.

3.2. Optimization

In this section, we describe how to efficiently optimize the cost function described in Eq. (5). In particular, we explore approximated updates of the assignment matrix that are compatible with online optimization schemes, like stochastic gradient descent (SGD).

Algorithm 1 Stochastic optimization of Eq. (5).

```
Require: $T$ batches of images, $\lambda_0 > 0$
  for $t = \{1, \dots, T\}$ do
    Obtain batch $b$ and representations $r$
    Compute $f_\theta(X_b)$
    Compute $P^*$ by minimizing Eq. (2) w.r.t. $P$
    Compute $\nabla_\theta L(\theta)$ from Eq. (2) with $P^*$
    Update $\theta \leftarrow \theta - \lambda_t \nabla_\theta L(\theta)$
  end for
```

Updating the assignment matrix P. Directly solving for the optimal assignment requires to evaluate the distances between all the n features and the k representations. In order to efficiently solve this problem, we first reduce the number k of representations to n. This limits the set $\mathcal{P}$ to the set of permutation matrices, *i.e.*,

$$\mathcal{P} = \{P \in \{0,1\}^{n \times n} \mid P 1_n = 1_n,\ P^{\top} 1_n = 1_n\}. \tag{6}$$

Restricting the problem defined in Eq. (5) to this set, the linear assignment problem in P can be solved exactly with the Hungarian algorithm (Kuhn, 1955), but at the prohibitive cost of $O(n^3)$.

Instead, we perform stochastic updates of the matrix. Given a batch of samples, we optimize the assignment matrix P on its restriction to this batch. Given a subset $\mathcal{B}$ of b distinct images, we only update the $b \times b$ square sub matrix $P_{\mathcal{B}}$ obtained by restricting P to these b images and their corresponding targets. In other words, each image can only be re-assigned to a target that was previously assigned to another image in the batch. This procedure has a complexity of $O(b^3)$ per batch, leading to an overall complexity of $O(nb^2)$, which is linear in the number of data points. We perform this update before updating the parameters θ of our features, in an on-line manner. Note that this simple procedure would not have been possible if $k > n$; we would have had to also consider the $k - n$ unassigned representations.

Stochastic gradient descent. Apart from the update of the assignment matrix P, we use the same optimization scheme as standard supervised approaches, *i.e.*, SGD with batch normalization (Ioffe & Szegedy, 2015). As noted by Tygert et al. (2017), batch normalization plays a crucial role when optimizing the l_2 loss, as it avoids exploding gradients. For each batch b of images, we first perform a forward pass to compute the distance between the images and the corresponding subset of target representations r. The Hungarian algorithm is then used on these distances to obtain the optimal reassignments within the batch. Once

	Softmax	Square loss
ImageNet	59.2	58.4

Table 1. Comparison between the softmax and the square loss for supervised object classification on ImageNet. The architecture is an AlexNet. The features are unit normalized for the square loss (Tygert et al., 2017). We report the accuracy on the validation set.

the assignments are updated, we use the chain rule in order to compute the gradients of all our parameters. Our optimization algorithm is summarized in Algorithm 1.

3.3. Implementation details

Our experiments solely focus on learning visual features with convnets. All details required to train these architectures with our approach are described below. Most of them are standard tricks used in the supervised setting.

Deep features. To ensure a fair empirical comparison with previous work, we follow Wang & Gupta (2015) and use an AlexNet architecture. We train it end to end using our unsupervised loss function. We subsequently test the quality of the learned visual feature by re-training a classifier on top. During transfer learning, we consider the output of the last convolutional layer as our features as in Razavian et al. (2014). We use the same multi-layer perceptron (MLP) as in Krizhevsky et al. (2012) for the classifier.

Pre-processing. We observe in practice that pre-processing the images greatly helps the quality of our learned features. As in Ranzato et al. (2007), we use image gradients instead of the images to avoid trivial solutions like clustering according to colors. Using this preprocessing is not surprising since most hand-made features like SIFT or HoG are based on image gradients (Lowe, 1999; Dalal & Triggs, 2005). In addition to this preprocessing, we also perform all the standard image transformations that are commonly applied in the supervised setting (Krizhevsky et al., 2012), such as random cropping and flipping of images.

Optimization details. We project the output of the network on the ℓ_2 sphere as in Tygert et al. (2017). The network is trained with SGD with a batch size of 256. During the first t_0 batches, we use a constant step size. After t_0 batches, we use a linear decay of the step size, *i.e.*, $l_t = \frac{l_0}{1+\gamma[t-t_0]_+}$. Unless mentioned otherwise, we permute the assignments within batches every 3 epochs. For the transfer learning experiments, we follow the guideline described in Donahue et al. (2016).

4. Experiments

We perform several experiments to validate different design choices in NAT. We then evaluate the quality of our features by comparing them to state-of-the-art unsupervised approaches on several auxiliary supervised tasks, namely object classification on ImageNet and object classification and detection of Pascal VOC 2007 (Everingham et al., 2010).

Transfering the features. In order to measure the quality of our features, we measure their performance on transfer learning. We freeze the parameters of all the convolutional layers and overwrite the parameters of the MLP classifier with random Gaussian weights. We precisely follow the training and testing procedure that is specific to each of the datasets following Donahue et al. (2016).

Datasets and baselines. We use the training set of ImageNet to learn our convolutional network (Deng et al., 2009). This dataset is composed of $1,281,167$ images that belong to $1,000$ object categories. For the transfer learning experiments, we also consider Pascal VOC 2007. In addition to fully supervised approaches (Krizhevsky et al., 2012), we compare our method to several unsupervised approaches, *i.e.*, autoencoder, GAN and BiGAN as reported in Donahue et al. (2016). We also compare to self-supervised approaches, *i.e.*, Agrawal et al. (2015); Doersch et al. (2015); Pathak et al. (2016); Wang & Gupta (2015) and Zhang et al. (2016). Finally we compare to state-of-the-art hand-made features, *i.e.*, SIFT with Fisher Vectors (SIFT+FV) (Sánchez et al., 2013). They reduce the Fisher Vectors to a $4,096$ dimensional vector with PCA, and apply an $8,192$ unit 3-layer MLP on top.

4.1. Detailed analysis

In this section, we validate some of our design choices, like the loss function, representations and the influences of some parameters on the quality of our features. All the experiments are run on ImageNet.

Softmax *versus* square loss. Table 1 compares the performance of an AlexNet trained with a softmax and a square loss. We report the accuracy on the validation set. The square loss requires the features to be unit normalized to avoid exploding gradients. As previously observed by Tygert et al. (2017), the performances are similar, hence validating our choice of loss function.

Effect of image preprocessing. In supervised classification, image pre-processing is not frequently used, and transformations that remove information are usually avoided. In the unsupervised case, however, we observe that is it is preferable to work with simpler inputs as

	clean	high-pass	sobel
acc@1	59.7	58.5	57.4

Table 2. Performance of supervised models with various image pre-processings applied. We train an AlexNet on ImageNet, and report classification accuracy.

it avoids learning trivial features. In particular, we observe that using grayscale image gradients greatly helps our method, as mentioned in Sec. 3. In order to verify that this preprocessing does not destroy crucial information, we propose to evaluate its effect on supervised classification. We also compare with high-pass filtering. Table 2 shows the impact of this preprocessing methods on the accuracy of an AlexNet on the validation set of ImageNet. None of these pre-processings degrade the perform significantly, meaning that the information related to gradients are sufficient for object classification. This experiment confirms that such pre-processing does not lead to a significant drop in the upper bound performance for our model.

Continuous *versus* discrete representations. We compare our choice for target vectors to those commonly used for clustering, *i.e.*, elements of the canonical basis of a k dimensional space. Such a representation makes a strong assumption on the structure of the problem, that it can be linearly separated in k different classes. This holds for ImageNet, giving a fair advantage to this discrete representation. We test this representation with k in $\{10^3, 10^4, 10^5\}$, which is a range well-suited for the $1,000$ classes of ImageNet. The matrix C contains n/k replications of k elements of the canonical basis. This assumes that the clusters are balanced, which is verified on ImageNet.

We compare these cluster-like representations to our continuous target vectors on the transfer task on ImageNet. Using discrete targets achieves an accuracy of 22%, which is significantly worse that our best performance, *i.e.*, 36.0%. A possible explanation is that binary vectors induce sharp discontinuous distances between representations. Such distances are hard to optimize over and may result in early convergence to poorer local minima.

Evolution of the features. In this experiment, we are interested in understanding how the quality of our features evolves with the optimization of our cost function. During the unsupervised training, we freeze the network every 20 epochs and learn a MLP classifier on top. We report the accuracy on the validation set of ImageNet. Figure 2 shows the evolution of the performance on this transfer task as we optimize for our unsupervised approach. The training performance improves monotonically with the epochs of the unsupervised training. This suggests that optimizing our objective function correlates with learning transferable features, *i.e.*, our features do not destroy useful class-level information. On the other hand, the test accuracy seems to saturate after a hundred epochs. This suggests that the MLP is overfitting rapidly on pre-trained features.

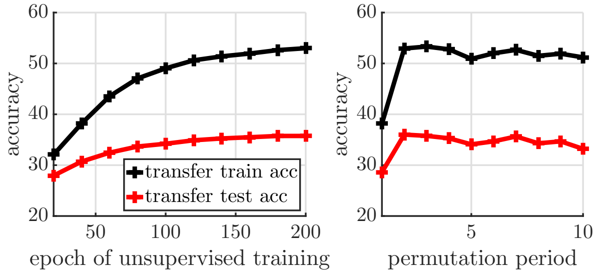

Figure 2. On the left, we measure the accuracy on ImageNet after training the features with our unsupervised approach as a function of the number of epochs. The performance improves with longer unsupervised training. On the right, we measure the accuracy on ImageNet after training the features with different permutation rates There is a clear trade-off with an optimum at permutations performed every 3 epochs.

Effect of permutations. Assigning images to their target representations is a the main feature of our approach. In this experiment, we want to understand how frequently we should update this assignment. Updating the assignment, even partially, is costly and may not be required to achieve good performance. Figure 2 shows the transfer accuracies on ImageNet as a function of the frequency of these updates. The model is quite robust to choice of frequency, with a test accuracy always above 30%. Interestingly, the accuracy actually degrades slightly with high frequency. A possible explanation is that the network overfits rapidly to its own output, leading to relatively worse features. In practice, we observe that updating the assignment matrix every 3 epochs offers a good trade-off between performance and accuracy. When training the network and keeping P fixed, we obtain a baseline test accuracy of 26.4%.

Visualizing the filters. Figure 4 shows a comparison between the first convolutional layer of an AlexNet trained with and without supervision. Both take grayscale gradient images as input. The visualization are obtained by composing the Sobel filtering with the filters of the first layer of the AlexNet. Unsupervised filters are slightly less sharp than their supervised counterpart, but still maintain edge and orientation information.

Nearest neighbor queries. Our loss optimizes a distance between features and fixed vectors. This means that looking at the distance between features should provide some information about the type of structure that our model cap-

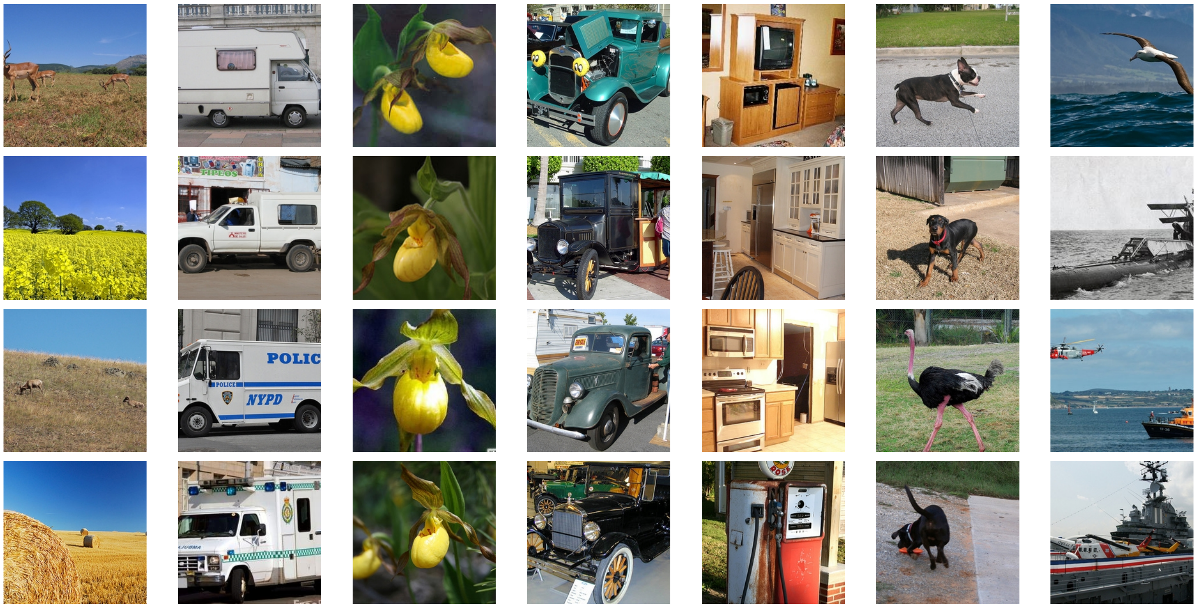

Figure 3. Images and their 3 nearest neighbors in ImageNet according to our model using an ℓ_2 distance. The query images are shown on the top row, and the nearest neighbors are sorted from the closer to the further. Our features seem to capture global distinctive structures.

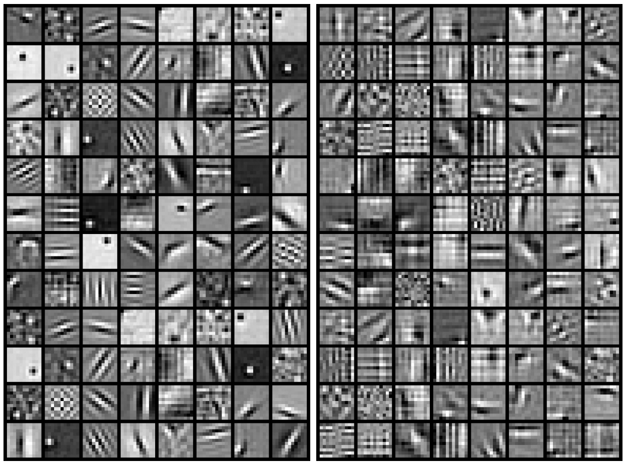

Figure 4. Filters form the first layer of an AlexNet trained on ImageNet with supervision (left) or with NAT (right). The filters are in grayscale, since we use grayscale gradient images as input. This visualization shows the composition of the gradients with the first layer.

tures. Given a query image x, we compute its feature $f_\theta(x)$ and search for its nearest neighbors according to the ℓ_2 distance. Figure 3 shows images and their nearest neighbors.

The features capture relatively complex structures in images. Objects with distinctive structures, like trunks or fruits, are well captured by our approach. However, this information is not always related to image labels. For example, the image of bird on the sea is matched to images more related to the sea or the sky rather than the bird.

4.2. Comparison with the state of the art

We report results on the transfer task both on ImageNet and PASCAL VOC 2007. The model is trained on ImageNet.

ImageNet classification. In this experiment, we evaluate the quality of our features for the object classification task of ImageNet. In this setup, we build the unsupervised features on images that correspond to predefined image categories. Even though we do not have access to labels, the data itself is biased towards these classes. In order to evaluate the features, we freeze the layers up to the last convolutional layer and train the classifier with supervision. This experimental setting follows Noroozi & Favaro (2016).

We compare our model with several self-supervised (Wang & Gupta, 2015; Doersch et al., 2015; Zhang et al., 2016) and one unsupervised approach, *i.e.*, Donahue et al. (2016). Note that self-supervised approaches use losses specifically designed for visual features. Like BiGANs (Donahue et al., 2016), NAT does not make any assumption about the domain but of the structure of its features. Table 3 compares NAT with these approaches.

Among unsupervised approaches, NAT compares favorably to BiGAN (Donahue et al., 2016). Interestingly, the performance of NAT are slightly better than *self-supervised* methods, even though we do not explicitly use domain-specific clues in images or videos to guide the learning. While all the models provide performance in the $30-36\%$ range, it is not clear if they all learn the same features. Finally, all the unsupervised deep features are outperformed

Method	Acc@1
Random (Noroozi & Favaro, 2016)	12.0
SIFT+FV (Sánchez et al., 2013)	55.6
Wang & Gupta (2015)	29.8
Doersch et al. (2015)	30.4
Zhang et al. (2016)	35.2
[1]Noroozi & Favaro (2016)	38.1
BiGAN (Donahue et al., 2016)	32.2
NAT	36.0

Table 3. Comparison of the proposed approach to state-of-the-art unsupervised feature learning on ImageNet. A full multi-layer perceptron is retrained on top of the features. We compare to several self-supervised approaches and an unsupervised approach, *i.e.*, BiGAN (Donahue et al., 2016). [1]Noroozi & Favaro (2016) uses a significantly larger amount of features than the original AlexNet. We report classification accuracy.

by hand-made features, in particular Fisher Vectors with SIFT descriptors. This baseline uses a slightly bigger MLP for the classifier and its performance can be improved by 2.2% by bagging 8 such models. This difference of 20% in accuracy shows that unsupervised deep features are still quite far from the state-of-the-arts among *all* unsupervised features.

Transferring to PASCAL VOC 2007. We carry out a second transfer experiment on the PASCAL VOC dataset, on the classification and detection tasks. The model is trained on ImageNet. Depending on the task, we *finetune* all layers in the network, or solely the classifier, following Donahue et al. (2016). In all experiments, the parameters of the convolutional layers are initialized with the ones obtained with our unsupervised approach. The parameters of the classification layers are initialized with gaussian weights. We get rid of batch normalization layers and use a data-dependent rescaling of the parameters (Krähenbühl et al., 2015). Table 4 shows the comparison between our model and other unsupervised approaches. The results for other methods are taken from Donahue et al. (2016) except for Zhang et al. (2016).

As with the ImageNet classification task, our performance is on par with self-supervised approaches, for both detection and classification. Among purely unsupervised approaches, we outperform standard approaches like autoencoders or GANs by a large margin. Our model also performs slightly better than the best performing BiGAN model (Donahue et al., 2016). These experiments confirm our findings from the ImageNet experiments. Despite its simplicity, NAT learns feature that are as good as those obtained with more sophisticated and data-specific models.

	Classification		Detection
Trained layers	fc6-8	all	all
ImageNet labels	78.9	79.9	56.8
Agrawal et al. (2015)	31.0	54.2	43.9
Pathak et al. (2016)	34.6	56.5	44.5
Wang & Gupta (2015)	55.6	63.1	47.4
Doersch et al. (2015)	55.1	65.3	51.1
Zhang et al. (2016)	61.5	65.6	46.9
Autoencoder	16.0	53.8	41.9
GAN	40.5	56.4	-
BiGAN (Donahue et al., 2016)	52.3	60.1	46.9
NAT	56.7	65.3	49.4

Table 4. Comparison of the proposed approach to state-of-the-art unsupervised feature learning on VOC 2007 Classification and detection. We either fix the features after conv5 or we fine-tune the whole model. We compare to several self-supervised and an unsupervised approaches. The GAN and autoencoder baselines are from Donahue et al. (2016). We report mean average prevision as customary on PASCAL VOC.

5. Conclusion

This paper presents a simple unsupervised framework to learn discriminative features. By aligning the output of a neural network to low-dimensional noise, we obtain features on par with state-of-the-art unsupervised learning approaches. Our approach explicitly aims at learning discriminative features, while most unsupervised approaches target surrogate problems, like image denoising or image generation. As opposed to self-supervised approaches, we make very few assumptions about the input space. This makes our appproach very simple and fast to train. Interestingly, it also shares some similarities with traditional clustering approaches as well as retrieval methods. While we show the potential of our approach on visual data, it will be interesting to try other domains. Finally, this work only considers simple noise distributions and alignment methods. A possible direction of research is to explore target distributions and alignments that are more informative. This also would strengthen the relation between NAT and methods based on distribution matching like the earth mover distance.

Acknowledgement. We greatly thank Hervé Jégou for his help throughout the development of this project. We also thank Allan Jabri, Edouard Grave, Iasonas Kokkinos, Léon Bottou, Matthijs Douze and the rest of FAIR for their support and helpful discussion. Finally, we thank Richard Zhang, Jeff Donahue and Florent Perronnin for their help.

References

Agrawal, P., Carreira, J., and Malik, J. Learning to see by moving. In *ICCV*, 2015.

Bach, F. and Harchaoui, Z. Diffrac: a discriminative and flexible framework for clustering. In *NIPS*, 2007.

Bengio, Y., Lamblin, P., Popovici, D., and Larochelle, H. Greedy layer-wise training of deep networks. In *NIPS*, 2007.

Bojanowski, P., Bach, F., Laptev, I., Ponce, J., Schmid, C., and Sivic, J. Finding actors and actions in movies. In *ICCV*, 2013.

Bojanowski, P., Lajugie, R., Bach, F., Laptev, I., Ponce, J., Schmid, C., and Sivic, J. Weakly supervised action labeling in videos under ordering constraints. In *ECCV*, 2014.

Coates, A. and Ng, A. Learning feature representations with k-means. In *Neural Networks: Tricks of the Trade*. Springer, 2012.

Dalal, N. and Triggs, B. Histograms of oriented gradients for human detection. In *CVPR*, 2005.

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

Denton, E. L., Chintala, S., and Fergus, R. Deep generative image models using a laplacian pyramid of adversarial networks. In *NIPS*, 2015.

Doersch, C., Gupta, A., and Efros, A. Unsupervised visual representation learning by context prediction. In *CVPR*, 2015.

Donahue, J., Krähenbühl, P., and Darrell, T. Adversarial feature learning. *arXiv preprint arXiv:1605.09782*, 2016.

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

Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., and Zisserman, A. The PASCAL visual object classes (VOC) challenge. *IJCV*, 2010.

Fukushima, K. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36:193–202, 1980.

Girshick, R. Fast r-cnn. In *CVPR*, 2015.

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

Goodman, J. Classes for fast maximum entropy training. In *ICASSP*, 2001.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *CVPR*, 2016.

Ioffe, S. and Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

Jayaraman, D. and Grauman, K. Learning image representations tied to ego-motion. In *ICCV*, 2015.

Joulin, A. and Bach, F. A convex relaxation for weakly supervised classifiers. In *ICML*, 2012.

Joulin, A., Bach, F., and Ponce, J. Discriminative clustering for image co-segmentation. In *CVPR*, 2010.

Joulin, A., van der Maaten, L., Jabri, A., and Vasilache, N. Learning visual features from large weakly supervised data. In *ECCV*, 2016.

Kingma, D. and Welling, M. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Kohonen, T. Self-organized formation of topologically correct feature maps. *Biological cybernetics*, 1982.

Krähenbühl, Philipp, Doersch, Carl, Donahue, Jeff, and Darrell, Trevor. Data-dependent initializations of convolutional neural networks. *arXiv preprint arXiv:1511.06856*, 2015.

Krause, A., Perona, P., and Gomes, R. G. Discriminative clustering by regularized information maximization. In *NIPS*, 2010.

Krizhevsky, A., Sutskever, I., and Hinton, G. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.

Kuhn, H. W. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955.

LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., and Jackel, L.D. Handwritten digit recognition with a back-propagation network. In *NIPS*, 1989.

Liao, R., Schwing, A., Zemel, R., and Urtasun, R. Learning deep parsimonious representations. In *NIPS*, 2016.

Lloyd, S. Least squares quantization in pcm. *Transactions on information theory*, 28(2):129–137, 1982.

Lowe, D. Object recognition from local scale-invariant features. In *ICCV*, 1999.

Mairal, J., Koniusz, P., Harchaoui, Z., and Schmid, C. Convolutional kernel networks. In *NIPS*, 2014.

Martinetz, T. and Schulten, K. A" neural-gas" network learns topologies. 1991.

Masci, J., Meier, U., Cireşan, D., and Schmidhuber, J. Stacked convolutional auto-encoders for hierarchical feature extraction. In *ICANN*, 2011.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

Noroozi, M. and Favaro, P. Unsupervised learning of visual representations by solving jigsaw puzzles. *arXiv preprint arXiv:1603.09246*, 2016.

Oquab, M., Bottou, L., Laptev, I., and Sivic, J. Learning and transferring mid-level image representations using convolutional neural networks. In *CVPR*, 2014.

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

Pinheiro, P. O., Collobert, R., and Dollar, P. Learning to segment object candidates. In *NIPS*, 2015.

Ramanathan, V., Joulin, A., Liang, P., and Fei-Fei, L. Linking people in videos with their names using coreference resolution. In *ECCV*, 2014.

Ranzato, M. A., Huang, F. J., Boureau, Y. L., and LeCun, Y. Unsupervised learning of invariant feature hierarchies with applications to object recognition. In *CVPR*, 2007.

Razavian, A. Sharif, Azizpour, H., Sullivan, J., and Carlsson, S. CNN features off-the-shelf: an astounding baseline for recognition. In *arXiv 1403.6382*, 2014.

Rubner, Y., Tomasi, C., and Guibas, L. J. A metric for distributions with applications to image databases. In *ICCV*, 1998.

Sánchez, J., Perronnin, F., Mensink, T., and Verbeek, J. Image classification with the fisher vector: Theory and practice. *IJCV*, 105(3):222–245, 2013.

Tang, K., Joulin, A., Li, L.-J., and Fei-Fei, L. Co-localization in real-world images. In *CVPR*, 2014.

Tygert, M., Chintala, S., Szlam, A., Tian, Y., and Zaremba, W. Scale-invariant learning and convolutional networks. *Applied and Computational Harmonic Analysis*, 42(1): 154–166, 2017.

van den Oord, A., Kalchbrenner, N., Espeholt, L., Vinyals, O., and Graves, A. Conditional image generation with pixelcnn decoders. In *NIPS*, 2016.

Vesanto, J. and Alhoniemi, E. Clustering of the self-organizing map. *Transactions on neural networks*, 2000.

Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., and Manzagol, P.-A. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *JMLR*, 11(Dec):3371–3408, 2010.

Wang, X. and Gupta, A. Unsupervised learning of visual representations using videos. In *ICCV*, 2015.

Xie, J., Girshick, R., and Farhadi, A. Unsupervised deep embedding for clustering analysis. In *ICML*, 2016.

Xu, L., Neufeld, J., Larson, B., and Schuurmans, D. Maximum margin clustering. In *NIPS*, 2004.

Yang, J., Parikh, D., and Batra, D. Joint unsupervised learning of deep representations and image clusters. In *CVPR*, 2016.

Zhang, R., Isola, P., and Efros, A. Colorful image colorization. In *ECCV*, 2016.

Zhao, J., Mathieu, M., Goroshin, R., and LeCun, Y. Stacked What-Where Auto-encoders. In *Workshop at ICLR*, 2016.