Data-Efficient Image Recognition with Contrastive Predictive Coding

Olivier J. Hénaff, Aravind Srinivas, Jeffrey De Fauw, Ali Razavi,
Carl Doersch, S. M. Ali Eslami, Aaron van den Oord
DeepMind
London, UK

Abstract

Human observers can learn to recognize new categories of images from a handful of examples, yet doing so with machine perception remains an open challenge. We hypothesize that data-efficient recognition is enabled by representations which make the variability in natural signals more predictable. We therefore revisit and improve Contrastive Predictive Coding, an unsupervised objective for learning such representations. This new implementation produces features which support state-of-the-art linear classification accuracy on the ImageNet dataset. When used as input for non-linear classification with deep neural networks, this representation allows us to use 2–5× less labels than classifiers trained directly on image pixels. Finally, this unsupervised representation substantially improves transfer learning to object detection on PASCAL VOC-2007, surpassing fully supervised pre-trained ImageNet classifiers.

1 Introduction

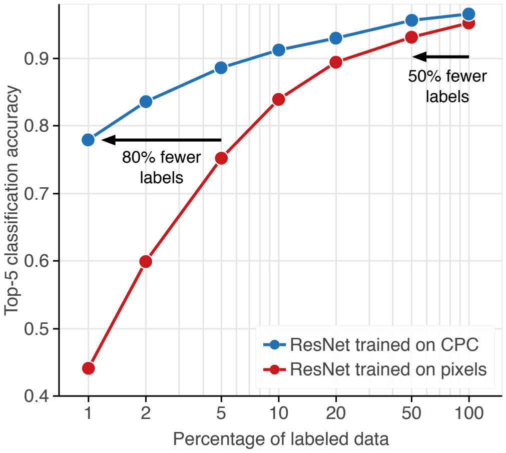

Figure 1: Data-efficient image recognition with Contrastive Predictive Coding. With decreasing amounts of labeled data, supervised networks trained on pixels fail to generalize (red). When trained on unsupervised representations learned with CPC, these networks retain a much higher accuracy in this low-data regime (blue). Equivalently, the accuracy of supervised networks can be matched with significantly fewer labels.

Deep neural networks excel at perceptual tasks when labeled data are abundant, yet their performance degrades substantially when provided with limited supervision (Fig. 1, red). In contrast, humans and animals can quickly learn about new classes of images from few examples [34, 40]. What accounts for this monumental difference in data-efficiency between biological and machine vision? While highly-structured representations (e.g. as proposed by [33]) may improve data-efficiency, it remains unclear how to program explicit structures that capture the enormous complexity of real-world visual scenes, such as those captured in the ImageNet dataset [52]. An alternative hypothesis has therefore proposed that intelligent systems need not be structured *a priori*, but can instead learn about the structure of the world in an unsupervised manner [6, 26, 36]. Choosing an appropriate training objective is an open problem, but a promising guiding principle has emerged recently: good representations should make the spatio-temporal variability in natural signals more predictable. Indeed, human perceptual representations have been shown to linearize (or 'straighten') the temporal transformations found in natural videos, a property lacking from current supervised image recognition models [27], and theories of both spatial and temporal predictability have succeeded in describing properties of early visual areas [50, 46]. In this work, we hypothesize that spatially predictable representations may allow artificial systems to benefit from human-like data-efficiency.

Contrastive Predictive Coding (CPC, [58]) is an unsupervised objective which learns predictable representations. CPC is a general technique that only requires in its definition that observations be ordered along e.g. temporal or spatial dimensions, and as such has been applied to a variety of different modalities including speech, natural language and images. This generality, combined with the strong performance of its representations in downstream linear classification tasks, makes CPC a promising candidate for investigating the efficacy of predictable representations for data-efficient image recognition.

Our work makes the following contributions:

- We revisit CPC in terms of its architecture and training methodology, and arrive at a new implementation with a dramatically-improved ability to linearly separate image classes (from 48.7% to 71.5% Top-1 ImageNet classification accuracy, a 23% absolute improvement), setting a new state-of-the-art.
- We then train deep neural networks on top of the resulting CPC representations using very few labeled images (e.g. 1% of the ImageNet dataset), and demonstrate test-time classification accuracy far above networks trained on raw pixels (78% Top-5 accuracy, a 34% absolute improvement), outperforming all other semi-supervised learning methods (+20% Top-5 accuracy over the previous state-of-the-art [63]). This gain in accuracy allows our classifier to surpass supervised ones trained with 5× more labels.
- Surprisingly, this representation also surpasses supervised methods when given the entire ImageNet dataset (+3.2% Top-1 accuracy). Alternatively our classifier matches fully-supervised ones while only using half of the labels.
- We isolate the contributions of different components of the final model to such downstream tasks. Interestingly, we find that linear classification accuracy is not always predictive of low-data classification accuracy, emphasizing the importance of this metric as a stand-alone benchmark for unsupervised learning.
- Finally, we assess the generality of CPC representations by transferring them to a new task and dataset: object detection on PASCAL-VOC 2007. Consistent with the results from the previous sections, we find CPC to give state-of-the-art performance in this setting (76.6% mAP), surpassing the performance of supervised transfer learning (+2% absolute improvement).

2 Experimental Setup

We first review the CPC architecture and learning objective in section 2.1, before detailing how we use its resulting representations for image recognition tasks in section 2.2.

2.1 Contrastive Predictive Coding

Contrastive Predictive Coding as formulated in [58] learns representations by training neural networks to predict the representations of future observations from those of past ones. When applied to images, the original formulation of CPC operates by predicting the representations of patches below a certain position from those above it (Fig. 2, left). These predictions are evaluated using a contrastive loss [9, 22], in which the network must correctly classify the 'future' representation amongst a set of unrelated 'negative' representations. This avoids trivial solutions such as representing all patches with a constant vector, as would be the case with a mean squared error loss.

In the CPC architecture, each input image is first divided into a set of overlapping patches $\boldsymbol{x}_{i,j}$, each of which is encoded with a neural network f_θ into a single vector $\boldsymbol{z}_{i,j} = f_\theta(\boldsymbol{x}_{i,j})$. To make predictions, a masked convolutional network g_ϕ is then applied to the grid of feature vectors. The masks are such that the receptive field of each resulting *context vector* $c_{i,j}$ only includes feature vectors that lie above it in the image (i.e. $\{\boldsymbol{z}_{u,v}\}_{u \leq i,v}$). The prediction task then consists of predicting 'future' feature vectors $z_{i+k,j}$ from current context vectors $c_{i,j}$, where $k > 0$. The predictions are

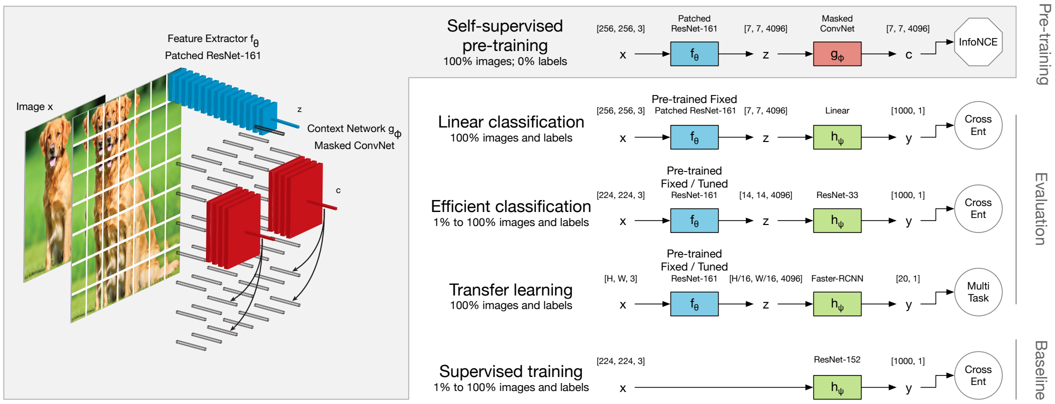

Figure 2: Overview of the framework for semi-supervised learning with Contrastive Predictive Coding. Left: unsupervised pre-training with the spatial prediction task (See Section 2.1). First, an image is divided into a grid of overlapping patches. Each patch is encoded independently from the rest with a feature extractor (blue) which terminates with a mean-pooling operation, yielding a single feature vector for that patch. Doing so for all patches yields a field of such feature vectors (wireframe vectors). Feature vectors above a certain level (in this case, the center of the image) are then aggregated with a context network (red), yielding a row of context vectors which are used to linearly predict features vectors below. Right: using the CPC representation for a classification task. Having trained the encoder network, the context network (red) is discarded and replaced by a classifier network (green) which can be trained in a supervised manner. In some experiments, we also fine-tune the encoder network (blue) for the classification task. When applying the encoder to cropped patches (as opposed to the full image) we refer to it as a *patched* ResNet in the figure.

made linearly: given a context vector $c_{i,j}$, a prediction length $k > 0$, and a prediction matrix W_k, the predicted feature vector is $\hat{\boldsymbol{z}}_{i+k,j} = \boldsymbol{W}_k \boldsymbol{c}_{i,j}$.

The quality of this prediction is then evaluated using a contrastive loss. Specifically, the goal is to correctly recognize the target $z_{i+k,j}$ among a set of randomly sampled feature vectors $\{z_l\}$ from the dataset. We compute the probability assigned to the target using a softmax, and evaluate this probability using the usual cross-entropy loss. Summing this loss over locations and prediction offsets, we arrive at the CPC objective as defined in [58]:

$$\mathcal{L}_{\text{CPC}} = -\sum_{i,j,k} \log p(\boldsymbol{z}_{i+k,j} | \hat{\boldsymbol{z}}_{i+k,j}, \{\boldsymbol{z}_l\}) = -\sum_{i,j,k} \log \frac{\exp(\hat{\boldsymbol{z}}_{i+k,j}^T \boldsymbol{z}_{i+k,j})}{\exp(\hat{\boldsymbol{z}}_{i+k,j}^T \boldsymbol{z}_{i+k,j}) + \sum_l \exp(\hat{\boldsymbol{z}}_{i+k,j}^T \boldsymbol{z}_l)}$$

The *negative samples* $\{\boldsymbol{z}_l\}$ are taken from other locations in the image and other images in the mini-batch. This loss is called InfoNCE [58] as it is inspired by Noise-Contrastive Estimation [21, 44] and has been shown to maximize the mutual information between $\boldsymbol{c}_{i,j}$ and $\boldsymbol{z}_{i+k,j}$ [58].

2.2 Evaluation protocol

Having trained an encoder network f_θ, a context network g_ϕ, and a set of linear predictors $\{\boldsymbol{W}_k\}$ using the CPC objective, we use the latents $\boldsymbol{z} = f_\theta(\boldsymbol{x})$ as a *representation* of new observations $\boldsymbol{x}$ for downstream tasks, and discard the rest. We then train a model h_ψ to classify these representations given a dataset of labeled images. More formally, given a dataset of N unlabeled images $\mathbb{D}_u = \{x_n\}$, and a (potentially much smaller) dataset of M labeled images $\mathbb{D}_l = \{x_m, y_m\}$:

$$\theta^* = \arg\min_\theta \frac{1}{N} \sum_{n=1}^{N} \mathcal{L}_{\text{CPC}}[f_\theta(x_n)] \qquad \psi^* = \arg\min_\psi \frac{1}{M} \sum_{m=1}^{M} \mathcal{L}_{\text{Sup}}[h_\psi \circ f_{\theta^*}(x_m), y_m]$$

In all cases, the dataset of unlabeled images $\mathbb{D}_u$ we pre-train on is the full ImageNet ILSVRC 2012 training set [52]. We consider three labeled datasets $\mathbb{D}_l$ for evaluation, each with an associated classifier h_ψ and supervised loss $\mathcal{L}_{\text{Sup}}$ (see Fig. 2, right). This protocol is sufficiently generic to allow us to later compare the CPC representation to other methods which have their own means of learning a feature extractor f_θ.

Linear classification is the standard benchmark for evaluating the quality of unsupervised image representations. In this regime, the classification network h_ψ is restricted to mean pooling followed by a single linear layer, and the parameters of f_θ are kept fixed. The labeled dataset $\mathbb{D}_l$ is the entire ImageNet dataset, and the supervised loss $\mathcal{L}_{\text{Sup}}$ is standard cross-entropy. We use the same data-augmentation as in the unsupervised learning phase for training, and none at test time and evaluate with a single crop.

Efficient classification directly tests whether the CPC representation enables generalization from few labels. For this task, the classifier h_ψ is an arbitrary deep neural network (we use an 11-block ResNet architecture with 4096-dimensional feature maps and 1024-dimensional bottleneck layers). The labeled dataset $\mathbb{D}_l$ is a subset of the ImageNet dataset: we investigated using 1%, 2%, 5%, 10%, 20%, 50% and 100% of the ImageNet dataset. The supervised loss $\mathcal{L}_{\text{Sup}}$ is again cross-entropy. We use the same data-augmentation as during unsupervised pre-training, none at test-time and evaluate with a single crop.

Transfer learning tests the generality of the representation by applying it to a new task and dataset. For this we chose image detection on the PASCAL-2007 dataset, a standard benchmark in computer vision [18]. As such $\mathbb{D}_l$ is the entire PASCAL-2007 dataset (comprised of 5011 labeled images); h_ψ and $\mathcal{L}_{\text{Sup}}$ are the Faster-RCNN architecture and loss [51]. In addition to color-dropping, we use scale-augmentation [14] for training.

For **linear classification**, we keep the feature extractor f_θ *fixed* to assess the representation in absolute terms. For **efficient classification** and **transfer learning**, we additionally explore *fine-tuning* the feature extractor for the supervised objective. In this regime, we initialize the feature extractor and classifier with the solutions θ^*, ψ^* found in the previous learning phase, and train them both for the supervised objective. To ensure that the feature extractor does not deviate too much from the solution dictated by the CPC objective, we use a smaller learning rate and early-stopping.

3 Related Work

Data-efficient learning has typically been approached by two complementary methods, both of which seek to make use of more plentiful unlabeled data: representation learning and label propagation. The former formulates an objective to learn a feature extractor f_θ in an unsupervised manner, whereas the latter directly constrains the classifier h_ψ using the unlabeled data.

Representation learning saw early success using generative modeling [31], but likelihood-based models have yet to generalize to more complex stimulus classes. Generative adversarial models have also been harnessed for representation learning [16], and large-scale implementations have recently achieved corresponding gains in linear classification accuracy [15].

In contrast to generative models which require the reconstruction of observations, self-supervised techniques directly formulate tasks involving the learned representation. For example, simply asking a network to recognize the spatial layout of an image led to representations that transferred to popular vision tasks such as classification and detection [14, 45]. Other works showed that prediction of color [64, 35] and image orientation [19], and invariance to data augmentation [17] can provide useful self-supervised tasks. Beyond single images, works have leveraged video cues such as object tracking [59], frame ordering [42], and object boundary cues [38, 47]. Non-visual information can be equally powerful: information about camera motion [1, 29], scene geometry [62], or sound [2, 3] can all serve as natural sources of supervision.

While many of these tasks require predicting fixed quantities computed from the data, another class of *contrastive* methods [9, 22] formulate their objectives in the learned representations themselves. CPC is a contrastive representation learning method that maximizes the mutual information between spatially removed latent representations with InfoNCE [58], a loss function based on Noise-Contrastive Estimation [21, 44]. Two other methods have recently been proposed using the same loss function, but with different associated prediction tasks. Contrastive Multiview Coding [57] maximizes the mutual information between representations of different views of the same observation. Augmented Multiscale Deep InfoMax (AMDIM, [5]) is most similar to CPC in that it makes predictions across space, but differs in that it also predicts representations across layers in the model.

A common alternative approach for improving data efficiency is **label-propagation** [65], where a classifier is trained on a subset of labeled data, then used to label parts of the unlabeled dataset. This label-propagation can either be discrete (as in pseudo-labeling [37]) or continuous (as in entropy minimization [20]). The predictions of this classifier are often constrained to be smooth with respect to certain deformations, such as data-augmentation [61] or adversarial perturbation [43]. Representation learning and label propagation have been shown to be complementary and can be combined to great effect [63], hence we focus solely on representation learning in this paper.

4 Results

When asking whether CPC enables data-efficient learning, we wish to use the best representative of this model class. Unfortunately, purely unsupervised metrics tell us little about downstream performance, and implementation details have been shown to matter enormously [13, 32]. Since most representation learning methods have previously been evaluated using linear classification, we use this benchmark to guide a series of modifications to the training protocol and architecture (section 4.1) and compare to published results. In section 4.2 we assess whether this model enables efficient classification, and whether the first, more common metric (linear classification accuracy) is predictive of efficient classification. Finally, in section 4.3 we investigate the generality of our results through transfer learning to PASCAL-2007.

4.1 From CPC v1 to CPC v2

The overarching principle behind our new model design is to increase the scale and efficiency of the encoder architecture while also maximizing the supervisory signal we obtain from each image. At the same time, it is important to control the types of predictions that can be made across image patches, by removing low-level cues which might lead to degenerate solutions. To this end, we augment individual patches independently using stochastic data-processing techniques from supervised and self-supervised learning.

We identify four axes for model capacity and task setup that could impact the model's performance. The first axis increases model capacity by increasing depth and width, while the second improves training efficiency by introducing layer normalization. The third axis increases task complexity by making predictions in all four directions, and the fourth does so by performing more extensive patch-based augmentation.

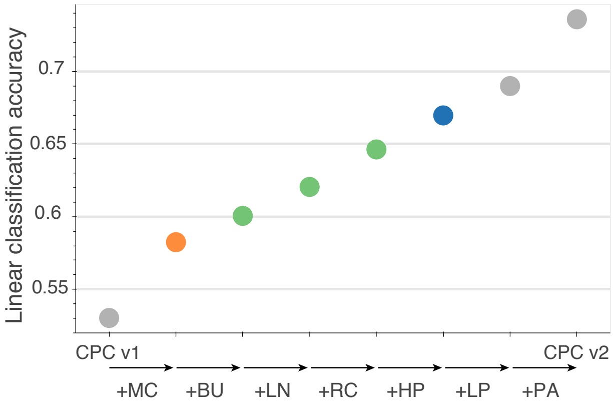

Figure 3: Linear classification performance of new variants of CPC, which incrementally add a series of modifications. MC: model capacity. BU: bottom-up spatial predictions. LN: layer normalization. RC: random color-dropping. HP: horizontal spatial predictions. LP: larger patches. PA: further patch-based augmentation. We use color to indicate the number of spatial predictions used (orange, green, blue for 1, 2 and 4 directions). Note that these accuracies are evaluated on a custom validation set and are therefore not directly comparable to the results we report and compare to.

Model capacity. Recent work has shown that larger networks and more effective training improves self-supervised learning [13, 32], but the original CPC model used only the first 3 stacks of a ResNet-101 [23] architecture. Therefore, we converted the third residual stack of ResNet-101 (originally containing 23 blocks, 1024-dimensional feature maps, and 256-dimensional bottleneck layers) to use 46 blocks with 4096-dimensional feature maps and 512-dimensional bottleneck layers. We call the resulting network ResNet-161. Consistent with prior results, this new architecture delivers better performance without any further modifications (Fig. 3, **+5%** Top-1 accuracy). We also increase the model's expressivity by increasing the size of its receptive field with larger patches (**+2%** Top-1 accuracy).

Table 1: Linear classification accuracy, and comparison to other self-supervised methods. In all cases the feature extractor is optimized in an unsupervised manner (using one of the methods listed below), and a linear classifier is trained on top using all labels in the ImageNet dataset.

Method	**Architecture**	**Parameters (M)**	**Top-1**	**Top-5**
Methods using ResNet-50:				
Local Aggregation [66]	ResNet-50	24	60.2	-
Momentum Contrast [25]	ResNet-50	24	60.6	-
CPC v2	ResNet-50	24	**63.8**	**85.3**
Methods using different architectures:				
Multi-task [13]	ResNet-101	28	-	69.3
Rotation [32]	RevNet-50 ×4	86	55.4	-
CPC v1 [58]	ResNet-101	28	48.7	73.6
BigBiGAN [15]	RevNet-50 ×4	86	61.3	81.9
AMDIM [5]	Custom-103	626	68.1	-
CMC [57]	ResNet-50 ×2	188	68.4	88.2
Momentum Contrast [25]	ResNet-50 ×4	375	68.6	-
CPC v2	ResNet-161	305	**71.5**	**90.1**

Layer normalization. Large architectures are more difficult to train efficiently. Early works on context prediction with patches used batch normalization [28, 14] to speed up training. However, with CPC we find that batch normalization actually harms downstream performance of large models. We hypothesize that batch normalization allows these models to find a trivial solution to CPC: it introduces a dependency between patches (through the batch statistics) that can be exploited to bypass the constraints on the receptive field. Nevertheless we find that we can reclaim much of batch normalization's training efficiency using layer normalization (**+2%** accuracy [4]).

Prediction lengths and directions. Larger architectures also run a greater risk of overfitting. We address this by asking more from the network: specifically, whereas van den Oord et al. [58] predicted each patch using only context from above, we repeatedly predict the patch using context from below, the right and the left, resulting in up to four times as many prediction tasks. Additional predictions tasks incrementally increased accuracy (adding bottom-up predictions: **+2%** accuracy, using all four spatial directions: **+2.5%** accuracy).

Patch-based augmentation. If the network can solve CPC using low-level patterns (e.g. straight lines continuing between patches or chromatic aberration), it need not learn semantically meaningful content. Augmenting the low-level variability across patches can remove such cues. The original CPC model spatially jitters individual patches independently. We further this logic by adopting the 'color dropping' method of [14], which randomly drops two of the three color channels in each patch, and find it to deliver systematic gains (**+3%** accuracy). We therefore continued by adding a fixed, generic augmentation scheme using the primitives from Cubuk et al. [10] (e.g. shearing, rotation, etc), as well as random elastic deformations and color transforms [11] (**+4.5%** accuracy). Note that these augmentations introduce some inductive bias about content-preserving transformations in images, but we do not optimize them for downstream performance (as in [10, 39]).

Comparison to previous art. Cumulatively, these fairly straightforward implementation changes lead to a substantial improvement to the original CPC model, setting a new state-of-the-art in unsupervised linear classification of **71.5%**Top-1 accuracy (see table 1).

How does this implementation methodology transfer to different architectures? In order to directly compare to other published results which use smaller models, we also ran these combined changes with a **ResNet-50**, arriving at **63.8%** classification accuracy. This model outperforms many recent approaches which at times use substantially larger models [13, 58, 32, 66, 15].

4.2 Efficient image classification

We now turn to our original question of whether CPC can enable data-efficient image recognition. We start by evaluating the performance of purely-supervised networks as the size of the labeled dataset $\mathbb{D}_l$ varies from 1% to 100% of ImageNet, training separate classifiers on each subset. We compared a range of different architectures (ResNet-50, -101, -152, and -200) and found a ResNet-200 to work best across all data-regimes (see Appendix). Despite our efforts to tune the supervised model for low-data classification (varying network depth, regularization, and optimization parameters) and extensive use of data-augmentation (including the transformations used for CPC pre-training), the accuracy of the best model only reaches 44.1% Top-5 accuracy when trained on 1% of the dataset (compared to 95.2% when trained on the entire dataset, see Table 2 and Fig. 1, red).

Contrastive Predictive Coding. We now address our central question of whether CPC enables data-efficient learning. We follow the same paradigm as for the supervised baseline (training and evaluating a separate classifier for each subset), stacking a neural network classifier on top of the CPC v2 latents $\boldsymbol{z} = f_\theta(\boldsymbol{x})$ rather than the raw image pixels $\boldsymbol{x}$ (see section 2.2). This leads to a substantial increase in accuracy (Table 2 and Fig. 1, blue curve), yielding 77.9% Top-5 accuracy with only 1% of the labels, a 34% absolute improvement (77% relative) over purely-supervised methods. Surprisingly, when given the entire dataset, this classifier reaches 83.4%/96.5% Top1/Top5 accuracy, surpassing our supervised baseline (ResNet-200: 80.2%/95.2% accuracy) and published results (original ResNet-200 v2: 79.9%/95.2% [24], with AutoAugment: 80.0%/95.0% [10]). Using this representation also leads to gains in data-efficiency. With only 50% of the labels our classifier surpasses the supervised baseline given the entire dataset. Similarly, with only 1% of the labels, our classifier surpasses the supervised baseline given 5%, demonstrating a 2× to 5× gain in data-efficiency across the range we tested (see Fig. 1 and highlighted numbers in table 2).

How important are the model specifications described in Section 4.1 for low-data classification? We hypothesized that predictable representations might enable data-efficient classification, and therefore expect that increasing the amount of 'predictability' in the representation should also increase its ability to learn from small amounts of data. Fig. 4 shows evidence for this by ablating model parameters and comparing linear classification performance against low-data classification. Consistent with our hypothesis, increasing the number of spatial directions in the CPC prediction task (i.e., from only top-down predictions to both vertical directions, to all four spatial directions) systematically increases low-data classification performance (Fig. 4, different color groups). As a control, we asked if all modifications that improve linear classification also improve low-data classification. We did not find evidence in favor of this: improvements in linear classification as a result of changing other model parameters seem uncorrelated to performance in other tasks (Fig. 4, within green group: $R^2 = 0.17, p = 0.36$).

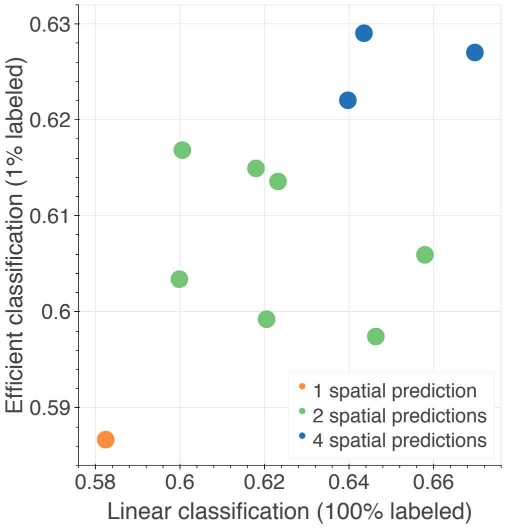

Figure 4: Relationship between linear classification accuracy and low-data classification, for different training protocols. Orange, green, and blue dots correspond to CPC models making predictions in 1, 2, and 4 spatial directions respectively. Within a color group, different models correspond to other implementation details.

Other unsupervised representations. How well does the CPC representation compare to other representations that have been learned in an unsupervised manner? Table 2 compares our best model with other works on efficient recognition. We consider three objectives from different model classes: self-supervised learning with rotation prediction [63], large-scale adversarial feature learning (BigBiGAN [15]), and another contrastive prediction objective (AMDIM [5]). Zhai et al. [63] evaluate the low-data classification performance of representations learned with rotation prediction using a similar paradigm and architecture (ResNet-152), hence we report their results directly. Given 1% of

Table 2: Comparison to other methods for semi-supervised learning. *Representation learning* methods use a classifier to discriminate an unsupervised representation, and optimize it solely with respect to labeled data. *Label-propagation* methods on the other hand further constrain the classifier with smoothness and entropy criteria on unlabeled data, making the additional assumption that all training images fit into a single (unknown) testing category. *Fixed* and *fine-tuned* denote whether the feature extractor is allowed to accommodate the supervised objective. The $^{\star\#}$ markers highlight comparisons showing gains in data-efficiency relative to supervised learning.

Method	**Architecture**			**Top-5 accuracy**		
Labeled data		1%	5%	10%	50%	100%
†Supervised baseline	ResNet-200	44.1	75.2*	83.9	93.1	95.2$^{\#}$
Methods using label-propagation:						
Pseudolabeling [63]	ResNet-50	51.6	-	82.4	-	-
VAT + Entropy Minimization [63]	ResNet-50	47.0	-	83.4	-	-
Unsup. Data Augmentation [61]	ResNet-50	-	-	88.5	-	-
Rotation + VAT + Ent. Min. [63]	ResNet-50 ×4	-	-	**91.2**	-	95.0
Methods using representation learning only:						
Instance Discrimination [60]	ResNet-50	39.2	-	77.4	-	-
Rotation [63]	ResNet-152 ×2	57.5	-	86.4	-	-
ResNet on BigBiGAN (fixed)	RevNet-50 ×4	55.2	73.7	78.8	85.5	87.0
ResNet on AMDIM (fixed)	Custom-103	67.4	81.8	85.8	91.0	92.2
ResNet on CPC v2 (fixed)	ResNet-161	77.1	87.5	90.5	95.0	96.2
ResNet on CPC v2 (fine-tuned)	ResNet-161	**77.9***	**88.6**	**91.2**	**95.6**$^{\#}$	**96.5**

ImageNet, their method achieves 57.5% Top-5 accuracy. The authors of BigBiGAN and AMDIM do not report results on efficient classification, hence we evaluated these representations using the same paradigm we used for evaluating CPC. Specifically, since fine-tuned representations yield only marginal gains over fixed ones, we stack a ResNet classifier on top of these representations while keeping them fixed. Given 1% of ImageNet, classifiers trained on top of AMDIM and BigBiGAN achieve only 55.2% and 67.4% Top-5 accuracy, respectively.

Finally, Table 2 (top) also includes results for label-propagation algorithms. Note that the comparison is imperfect. These methods have an advantage in assuming that all unlabeled images can be assigned to a single category. At the same time, prior works (except for Zhai et al. [63] which use a ResNet-50 x4) report results with smaller networks, which may degrade performance relative to ours. Overall, we find that our results are on par with even the best such results [63], even though this work combines a variety of techniques (entropy minimization, virtual adversarial training, and pseudo-labeling) with self-supervised learning.

4.3 Transfer learning: image detection on PASCAL VOC 2007

We next investigate transfer learning performance on object detection on the PASCAL-2007 dataset, which reflects the practical scenario where a representation must be trained on a dataset with different statistics than the dataset of interest. This dataset also tests the efficiency of the representation as it only contains 5011 labeled images to train from. In this setting, we used a Faster-RCNN [51] image detection architecture, keeping the CPC representation trained on ImageNet as a feature extractor. As for efficient classification, we first trained the Faster-RCNN model while keeping the feature extractor fixed, then fine-tuned the entire model end-to-end. Table 3 displays our results compared to other methods. Most competing methods, which optimize a single unsupervised objective on ImageNet before fine-tuning on PASCAL detection, attain around 65% mean average precision. Leveraging larger unlabeled datasets increases their performance up to 67.8% [8]. Combining multiple forms of self-supervision enables them to reach 70.5% [13]. The proposed method, which learns only from ImageNet data using a single unsupervised objective, reaches 76.6%. Importantly,

Table 3: Comparison of PASCAL 2007 image detection accuracy to other transfer methods. The supervised baseline learns from the entire labeled ImageNet dataset and fine-tunes for PASCAL detection. The second class of methods learns from the same *unlabeled* images before transferring. All of these methods pre-train on the ImageNet dataset, except for DeeperCluster which learns from the larger, but uncurated, YFCC100M dataset [56]. All results are reported in terms of mean average precision (mAP).

Method	**Architecture**	**mAP**
Transfer from labeled data:		
Supervised baseline	ResNet-152	74.7
Transfer from unlabeled data:		
Exemplar [17] by [13]	ResNet-101	60.9
Motion Segmentation [47] by [13]	ResNet-101	61.1
Colorization [64] by [13]	ResNet-101	65.5
Relative Position [14] by [13]	ResNet-101	66.8
Multi-task [13]	ResNet-101	70.5
Instance Discrimination [60]	ResNet-50	65.4
Deep Cluster [7]	VGG-16	65.9
Deeper Cluster [8]	VGG-16	67.8
Local Aggregation [66]	ResNet-50	69.1
Momentum Contrast [25]	ResNet-50	74.9
Faster-RCNN trained on CPC v2	ResNet-161	**76.6**

this accuracy surpasses that attained by a representation that has been trained in a supervised manner with *all* ImageNet labels (74.7%), a longstanding challenge in computer vision. Concurrently with our results, He et al. [25] achieve 74.9% in the same transfer setting.

5 Discussion

We asked whether CPC could enable data-efficient image recognition, and found that it indeed greatly improves the accuracy of classifiers and object detectors when given small amounts of labeled data. Surprisingly, CPC even improves results given ImageNet-scale labels. Our results show that there is still room for improvement using relatively straightforward changes such as augmentation, optimization, and network architecture. Furthermore, we found that the standard method for evaluating unsupervised representations—linear classification—is only partially predictive of efficient recognition performance, suggesting that further research should focus on efficient recognition as a standalone benchmark. Overall, these results open the door toward research on problems where data is naturally limited, e.g. medical imaging or robotics.

Furthermore, images are far from the only domain where unsupervised representation learning is important: for example, unsupervised learning is already a critical step in language [41, 12], and shows promise in domains like audio [58, 3, 2], video [30, 42], and robotic manipulation [48, 49, 53]. Currently much self-supervised work builds upon tasks tailored for a specific domain (often images), which may not be easily adapted to other domains. Contrastive prediction methods, including the techniques suggested in this paper, are task agnostic and could therefore serve as a unifying framework for integrating these tasks and modalities. This generality is particularly useful given that many real-world environments are inherently multimodal, e.g. robotic environments which can have vision, audio, touch, proprioception, action, and more over long temporal sequences. Given the importance of increasing the amounts of self-supervision (via additional prediction tasks), integrating these modalities and tasks could lead to unsupervised representations which rival the efficiency and effectiveness of biological ones.

REFERENCES

[1] Pulkit Agrawal, Joao Carreira, and Jitendra Malik. Learning to see by moving. In *ICCV*, 2015.

[2] Relja Arandjelovic and Andrew Zisserman. Look, listen and learn. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 609–617, 2017.

[3] Relja Arandjelovic and Andrew Zisserman. Objects that sound. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 435–451, 2018.

[4] Lei Jimmy Ba, Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. *CoRR*, abs/1607.06450, 2016.

[5] Philip Bachman, R Devon Hjelm, and William Buchwalter. Learning representations by maximizing mutual information across views. *arXiv preprint arXiv:1906.00910*, 2019.

[6] H.B. Barlow. Unsupervised learning. *Neural Computation*, 1(3):295–311, 1989. doi: 10.1162/neco.1989.1.3.295.

[7] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *The European Conference on Computer Vision (ECCV)*, September 2018.

[8] Mathilde Caron, Piotr Bojanowski, Julien Mairal, and Armand Joulin. Leveraging large-scale uncurated data for unsupervised pre-training of visual features. 2019.

[9] Sumit Chopra, Raia Hadsell, and Yann LeCun. Learning a similarity metric discriminatively, with application to face verification. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2005*, pp. 539–546, 2005.

[10] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation policies from data. *arXiv preprint arXiv:1805.09501*, 2018.

[11] Jeffrey De Fauw, Joseph R Ledsam, Bernardino Romera-Paredes, Stanislav Nikolov, Nenad Tomasev, Sam Blackwell, Harry Askham, Xavier Glorot, Brendan O'Donoghue, Daniel Visentin, et al. Clinically applicable deep learning for diagnosis and referral in retinal disease. *Nature medicine*, 24(9):1342, 2018.

[12] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018.

[13] Carl Doersch and Andrew Zisserman. Multi-task self-supervised visual learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2051–2060, 2017.

[14] Carl Doersch, Abhinav Gupta, and Alexei A Efros. Unsupervised visual representation learning by context prediction. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1422–1430, 2015.

[15] Jeff Donahue and Karen Simonyan. Large scale adversarial representation learning. *arXiv preprint arXiv:1907.02544*, 2019.

[16] Jeff Donahue, Philipp Krähenbühl, and Trevor Darrell. Adversarial feature learning. *arXiv preprint arXiv:1605.09782*, 2016.

[17] Alexey Dosovitskiy, Jost Tobias Springenberg, Martin Riedmiller, and Thomas Brox. Discriminative unsupervised feature learning with convolutional neural networks. In *Advances in neural information processing systems*, pp. 766–774, 2014.

[18] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes challenge 2007 (voc2007) results. 2007.

[19] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. *arXiv preprint arXiv:1803.07728*, 2018.

[20] Yves Grandvalet and Yoshua Bengio. Semi-supervised learning by entropy minimization. In *Advances in neural information processing systems*, pp. 529–536, 2005.

[21] Michael Gutmann and Aapo Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pp. 297–304, 2010.

[22] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pp. 1735–1742. IEEE, 2006.

[23] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[24] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European conference on computer vision*, pp. 630–645. Springer, 2016.

[25] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. *arXiv preprint arXiv:1911.05722*, 2019.

[26] G.E. Hinton, T.J. Sejnowski, H.H.M.I.C.N.L.T.J. Sejnowski, and T.A. Poggio. *Unsupervised Learning: Foundations of Neural Computation*. A Bradford Book. MIT Press, 1999. ISBN 9780262581684.

[27] Olivier J. Hénaff, Robbe L. T. Goris, and Eero P. Simoncelli. Perceptual straightening of natural videos. *Nature Neuroscience*, Apr 2019.

[28] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[29] Dinesh Jayaraman and Kristen Grauman. Learning image representations tied to ego-motion. In *ICCV*, 2015.

[30] Longlong Jing and Yingli Tian. Self-supervised spatiotemporal feature learning by video geometric transformations. *arXiv preprint arXiv:1811.11387*, 2018.

[31] Durk P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. Semi-supervised learning with deep generative models. In *Advances in neural information processing systems*, pp. 3581–3589, 2014.

[32] Alexander Kolesnikov, Xiaohua Zhai, and Lucas Beyer. Revisiting self-supervised visual representation learning. *CoRR*, abs/1901.09005, 2019. URL `http://arxiv.org/abs/1901.09005`.

[33] Brenden M Lake, Ruslan Salakhutdinov, and Joshua B Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338, 2015.

[34] Barbara Landau, Linda B Smith, and Susan S Jones. The importance of shape in early lexical learning. *Cognitive development*, 3(3):299–321, 1988.

[35] Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. Colorization as a proxy task for visual understanding. In *CVPR*, pp. 6874–6883, 2017.

[36] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436, 2015.

[37] Dong-Hyun Lee. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on Challenges in Representation Learning, ICML*, volume 3, pp. 2, 2013.

[38] Yin Li, Manohar Paluri, James M Rehg, and Piotr Dollár. Unsupervised learning of edges. In *CVPR*, 2016.

[39] Sungbin Lim, Ildoo Kim, Taesup Kim, Chiheon Kim, and Sungwoong Kim. Fast autoaugment. *arXiv preprint arXiv:1905.00397*, 2019.

[40] Ellen M Markman. *Categorization and naming in children: Problems of induction*. mit Press, 1989.

[41] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger (eds.), *Advances in Neural Information Processing Systems 26*, pp. 3111–3119. Curran Associates, Inc., 2013.

[42] Ishan Misra, C Lawrence Zitnick, and Martial Hebert. Shuffle and learn: unsupervised learning using temporal order verification. In *ECCV*, 2016.

[43] Takeru Miyato, Shin-ichi Maeda, Shin Ishii, and Masanori Koyama. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *IEEE transactions on pattern analysis and machine intelligence*, 2018.

[44] Andriy Mnih and Koray Kavukcuoglu. Learning word embeddings efficiently with noise-contrastive estimation. In *Advances in neural information processing systems*, pp. 2265–2273, 2013.

[45] Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *European Conference on Computer Vision*, pp. 69–84. Springer, 2016.

[46] Stephanie E Palmer, Olivier Marre, Michael J Berry, and William Bialek. Predictive information in a sensory population. *Proceedings of the National Academy of Sciences*, 112(22): 6908–6913, 2015.

[47] Deepak Pathak, Ross Girshick, Piotr Dollár, Trevor Darrell, and Bharath Hariharan. Learning features by watching objects move. *arXiv preprint arXiv:1612.06370*, 2016.

[48] Lerrel Pinto and Abhinav Gupta. Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours. In *ICRA*, 2016.

[49] Lerrel Pinto, James Davidson, and Abhinav Gupta. Supervision via competition: Robot adversaries for learning tasks. *arXiv preprint arXiv:1610.01685*, 2016.

[50] Rajesh PN Rao and Dana H Ballard. Predictive coding in the visual cortex: a functional interpretation of some extra-classical receptive-field effects. *Nature neuroscience*, 2(1):79, 1999.

[51] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pp. 91–99, 2015.

[52] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

[53] Pierre Sermanet, Corey Lynch, Yevgen Chebotar, Jasmine Hsu, Eric Jang, Stefan Schaal, Sergey Levine, and Google Brain. Time-contrastive networks: Self-supervised learning from video. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1134–1141. IEEE, 2018.

[54] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

[55] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. *CoRR*, abs/1409.4842, 2014. URL `http://arxiv.org/abs/1409.4842`.

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

[57] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. *arXiv preprint arXiv:1906.05849*, 2019.

[58] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

[59] Xiaolong Wang and Abhinav Gupta. Unsupervised learning of visual representations using videos. In *ICCV*, 2015.

[60] Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3733–3742, 2018.

[61] Qizhe Xie, Zihang Dai, Eduard Hovy, Minh-Thang Luong, and Quoc V. Le. Unsupervised Data Augmentation. *arXiv e-prints*, art. arXiv:1904.12848, Apr 2019.

[62] Amir R Zamir, Tilman Wekel, Pulkit Agrawal, Colin Wei, Jitendra Malik, and Silvio Savarese. Generic 3D representation via pose estimation and matching. In *ECCV*, 2016.

[63] Xiaohua Zhai, Avital Oliver, Alexander Kolesnikov, and Lucas Beyer. S^4L: Self-supervised semi-supervised learning. *arXiv preprint arXiv:1905.03670*, 2019.

[64] Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In *European conference on computer vision*, pp. 649–666. Springer, 2016.

[65] Xiaojin Zhu and Zoubin Ghahramani. Learning from labeled and unlabeled data with label propagation. In *Technical Report CMU-CALD-02-107, Carnegie Mellon University*, 2002.

[66] Chengxu Zhuang, Alex Lin Zhai, and Daniel Yamins. Local aggregation for unsupervised learning of visual embeddings. *arXiv preprint arXiv:1903.12355*, 2019.

A Appendix

A.1 Additional Results

Table 4: Data efficient classification results with Top-1 accuracy.

Labeled data	1%	2%	5%	10%	20%	50%	100%
Method				**Top-1 accuracy**			
ResNet trained on pixels	23.1	34.8	50.6	62.5	70.3	75.9	80.2
ReseNet trained on CPC v2 (fixed)	51.2	58.6	66.2	71.4	75.9	80.0	82.3
ReseNet trained on CPC v2 (fine-tuned)	**52.7**	**60.4**	**68.1**	**73.1**	**76.7**	**81.2**	**83.4**

Table 5: Data efficient classification results with Top-5 accuracy (data in Fig. 1).

Labeled data	1%	2%	5%	10%	20%	50%	100%
Method				**Top-5 accuracy**			
ResNet trained on pixels	44.1	59.9	75.2	83.9	89.4	93.1	95.2
ReseNet trained on CPC v2 (fixed)	77.1	82.5	87.5	90.5	92.8	95.0	96.2
ReseNet trained on CPC v2 (fine-tuned)	**77.9**	**83.6**	**88.6**	**91.2**	**93.0**	**95.6**	**96.5**

A.2 InfoNCE implementation

For completeness, we provide pseudo-code for the main calculations involved in the InfoNCE objective, loosely modeled after Tensorflow operations. We suppose we have just calculated a set of latents $\boldsymbol{z}_{i,j} = f_\theta(\boldsymbol{x}_{i,j})$ for $i, j \in \{1, \ldots, 7\}$, each one being e.g. a 4096-dimensional vector. Assuming we do so for a batch of B images $\{\boldsymbol{x}\}$, the set of latents is a tensor of size $B \times 7 \times 7 \times 4096$.

```
def CPC(latents, target_dim=64, emb_scale=0.1,
        steps_to_ignore=2, steps_to_predict=3):
    # latents: [B, H, W, D]
    loss = 0.0
    context = pixelCNN(latents)
    targets = Conv2D(output_channels=target_dim,
                     kernel_shape=(1, 1))(latents)
    batch_dim, col_dim, row_dim = targets.shape[:-1]
    targets = reshape(targets, [-1, target_dim])
    for i in range(steps_to_ignore, steps_to_predict):
      col_dim_i = col_dim - i - 1
      total_elements = batch_dim * col_dim_i * row_dim

      preds_i = Conv2D(output_channels=target_dim,
                       kernel_shape=(1, 1))(context)
      preds_i = preds_i[:, :-(i+1), :, :] * emb_scale
      preds_i = reshape(preds_i, [-1, target_dim])

      logits = matmul(preds_i, targets, transpose_b=True)

      b = range(total_elements) / (col_dim_i * row_dim)
      col = range(total_elements) % (col_dim_i * row_dim)
      labels = b * col_dim * row_dim + (i+1) * row_dim + col

      loss += softmax_cross_entropy_with_logits(logits, labels)
    return loss
```

```
def pixelCNN(latents):
    # latents: [B, H, W, D]
    cres = latents
    cres_dim = cres.shape[-1]
    for _ in range(5):
      c = Conv2D(output_channels=256,
                 kernel_shape=(1, 1))(cres)
      c = ReLU(c)
      c = Conv2D(output_channels=256,
                 kernel_shape=(1, 3))(c)
      c = Pad(c, [[0, 0], [1, 0], [0, 0], [0, 0]])
      c = Conv2D(output_channels=256,
                 kernel_shape=(2, 1),
                 type='VALID')(c)
      c = ReLU(c)
      c = Conv2D(output_channels=cres_dim,
                 kernel_shape=(1, 1))(c)
      cres = cres + c
    cres = ReLU(cres)
    return cres
```

A.3 Linear classification

- Model architecture: Having extracted 80×80 patches with a stride of 32×32 from a 240×240 shaped input image, we end up with a grid of 6×6 features (each of which is obtained from our ResNet-161 architecture). This gives us a [6,6,4096] tensor for the image. We then use a Batch-Normalization layer to normalize the features (without scale parameter) followed by a 1x1 convolution mapping each feature in the grid to the 1000 logits for ImageNet classification. We then spatially-mean-pool these logits to end up with the final log probabilities for the linear classification.
- Optimization details: We use Adam Optimizer with a learning rate of 5e-4. We train the model on a batch size of 512 images with 32 images per core spread over 16 workers.

A.4 Efficient classification: purely supervised

In order to find the best model within this class, we vary the following hyperparameters:

- Model architecture: We investigate using ResNet-50, ResNet-101, ResNet-152, and ResNet-200 model architectures, all of them using the 'v2' variant [24], and find larger architecture to perform better, even when given smaller amounts of data. We insert a DropOut layer before the final linear classification layer [54].
- Optimization details: We vary the learning rate in {0.05, 0.1, 0.2}, the weight decay logarithmically from 10^{-5} to 10^{-2}, the DropOut linearly from 0 to 1, and the batch size per worker in {16, 32}.

We chose the best performing model for each training subset $\mathbb{D}_l$ of labeled ImageNet (using a separate validation set), and report its accuracy on the test set (i.e. the publicly available ILSVRC-2012 validation set).