Contrastive Multiview Coding

Yonglong Tian
MIT CSAIL
yonglong@mit.edu

Dilip Krishnan
Google Research
dilipkay@google.com

Phillip Isola
MIT CSAIL
phillipi@mit.edu

Abstract

Humans view the world through many sensory channels, e.g., the long-wavelength light channel, viewed by the left eye, or the high-frequency vibrations channel, viewed by the right ear. Each view is noisy and incomplete, but important factors, such as physics, geometry, and semantics, tend to be shared between all views (e.g., a "dog" can be seen, heard, and felt). We hypothesize that a powerful representation is one that models view-invariant factors. Based on this hypothesis, we investigate a contrastive coding scheme, in which a representation is learned that aims to maximize mutual information between different views but is otherwise compact. Our approach scales to any number of views, and is view-agnostic. The resulting learned representations perform above the state of the art for downstream tasks such as object classification, compared to formulations based on predictive learning or single view reconstruction, and improve as more views are added. Code and reference implementations are released on our project page: http://github.com/HobbitLong/CMC/.

1. Introduction

A foundational idea in coding theory is to learn compressed representations that nonetheless can be used to reconstruct the raw data. This idea shows up in contemporary representation learning in the form of autoencoders [50] and generative models [30, 20], which try to represent a data point or distribution as losslessly as possible. Yet lossless representation might not be what we really want, and indeed is trivial to achieve – the raw data itself is a lossless representation. What we might instead prefer is to keep the "good" information (signal) and throw away the rest (noise). How can we identify what information is signal and what is noise?

To an autoencoder, or a maximum likelihood generative model, a bit is a bit (since the goal is to maximize the log likelihood of the data). No one bit is better than any other. Our conjecture in this paper is that some bits *are* in fact better than others. Some bits code important properties like semantics, physics, and geometry, while others code attributes that we might consider less important, like incidental lighting conditions or thermal noise in a camera's sensor.

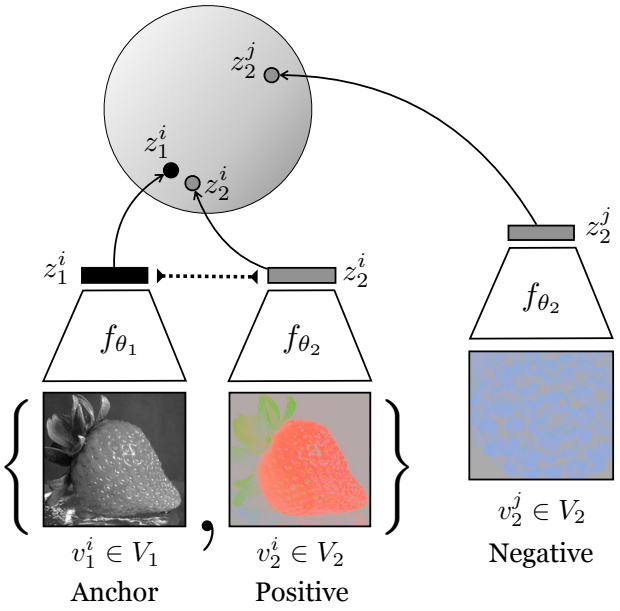


Figure 1: Our general framework for learning from multiple views of a dataset using a contrastive objective. For each set of views a deep representation is learnt by bringing views of the *same* scene together in embedding space, while pushing views of *different* scenes apart. Here we show an example of learning from two views: the luminance channel (L) of an image and the ab-color channel. The strawberry's L and ab channels embed to nearby points whereas the ab channel of a different image (a photo of blueberries) embeds to a far away point. This objective can be seen as maximizing a lower-bound to the mutual information between the representations of each view. The learnt representations transfer well to downstream tasks such as image classification.

We hypothesize that the good bits are the ones that are shared between multiple *views* of the world, for example between multiple sensory modalities like vision, sound, and touch. Under this perspective "presence of dog" is good information, since dogs can be seen, heard, and felt, but "camera pose" is bad information, since a camera's pose has little or no effect on the acoustic and tactile properties of the imaged scene. There is significant evidence in the

cognitive science and neuroscience literature that such cross-view representations are encoded across different regions of the brain e.g. [12, 25].

Our goal is therefore to learn representations that capture information shared between multiple sensory views but that are otherwise compact (i.e. throw away the bad information). To do so, we employ contrastive learning, where we learn a feature embedding such that views of the same scene map to nearby points while views of different scenes map to far apart points. In particular, we adapt the recently proposed method of Contrastive Predictive Coding (CPC) [46], except we simplify it – removing the recurrent network – and generalize it – show how to apply it to arbitrary collections of views, rather than just to temporal predictions. In reference to CPC, we term our method *Contrastive Multiview Coding* (CMC). The contrastive objective in our formulation, as in CPC, is based on Noise Contrastive Estimation (NCE) [21]. This objective can be understood as attempting to maximize the mutual information between the representations of each view.

We intentionally leave "good bits" only loosely defined and treat its definition as an empirical question. Ultimately, the proof is in the pudding: we consider a representation to be good if it makes subsequent problem solving easy, on tasks of human interest. For example, a useful representation of images might be a feature space in which it is easy to learn to recognize objects. We therefore evaluate our method by testing if the learned representations transfer well to standard semantic recognition tasks. On several benchmark tasks, our method achieves state of the art results, compared to other methods for unsupervised representation learning. We additionally find that the quality of the representation improves as a function of the number of views used for training. Finally, we compare the contrastive formulation of multiview learning to the recently popular approach of cross-view prediction, and find that in head-to-head comparisons, the contrastive approach learns stronger representations.

The core ideas that we build on: contrastive learning, mutual information maximization, and deep representation learning, are not new and have been explored in the literature on representation and multiview learning [34, 59, 2]. Our main contribution is to set up a framework to extend these ideas to any number of views, and we show the resulting significant benefits to the learned representation, in terms of tasks such as object recognition. Fig. 4 gives a pictorial overview of our framework for the different learning tasks we consider in this paper, to learn representations across datasets with different sets of views.

Our main contributions are:

- We apply contrastive learning to the multiview setting, where we learn representations that attempt to maximizes mutual information between different views of the same scene (e.g., between different image channels, or different modalities).
- Our approach yields representations that outperform the state-of-the-art in self-supervised learning in head-to-head comparisons.
- We compare the contrastive objective to cross-view prediction, finding an advantage to the contrastive approach.
- We extend the framework to learn from *more than two* views, and show that as the number of views increases, the quality of the learned representation improves.

2. Background

Unsupervised representation learning is about learning transformations of the data that make subsequent problem solving easier [5]. This field has a long history, starting with classical methods with well established algorithms, such as principal components analysis (PCA [29]) and independent components analysis (ICA [26]). These methods tend to learn representations that focus on low-level variations in the data, which are not very useful from the perspective of downstream tasks such as object recognition.

Representations better suited to such tasks have been learnt using deep neural networks, starting with seminal techniques such as Boltzmann machines [54, 50], autoencoders [23], variational autoencoders [30], generative adversarial networks [20] and autoregressive models [45]. Numerous other works exist, for a review see [5]. A powerful family of models for unsupervised representations are collected under the umbrella of "self-supervised" learning [62, 61, 28, 57, 48]. In these models, an input X to the model is transformed into an output $\hat{X}$, which is supposed to be close to another signal Y, which itself is related to X in some meaningful way. Examples of such X/Y pairs are: luminance and chrominance color channels of an image [62], patches from a single image [46], modalities such as vision and sound [47] or the frames of a video [57]. Clearly, such examples are numerous in the world, and provides us with nearly infinite amounts of training data: this is one of the appeals of this paradigm.

Closely related to self-supervised learning is the idea of multi-view learning, which is a general term involving many different approaches such as co-training [6], multi-kernel learning [11] and metric learning [4]; for comprehensive surveys please see [59, 34]. Nearly all existing works have dealt with one or two views such as video or image/sound. However, in many situations, many more views are available to provide training signals for any representation.

The objective functions used to train deep learning based representations in many of the above methods are either reconstruction-based loss functions such as Euclidean losses in different norms e.g. [27], adversarial loss functions [20]

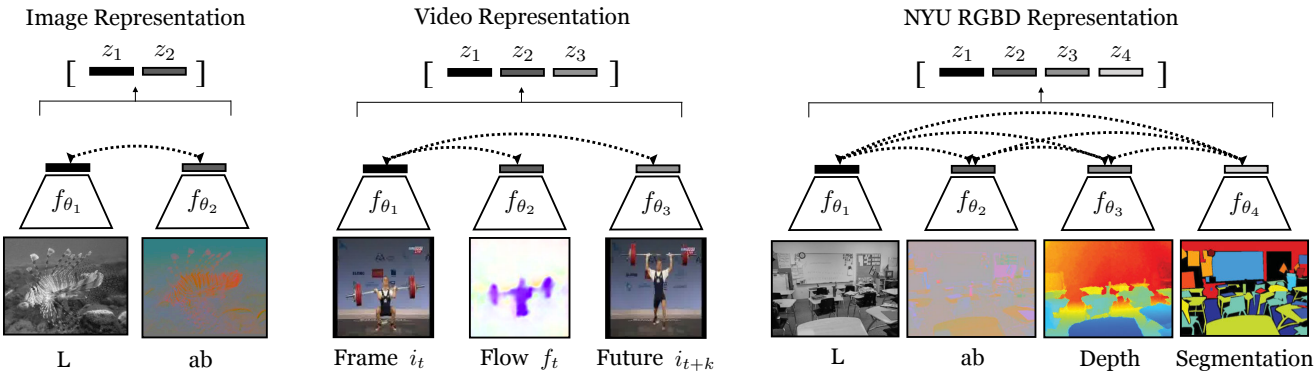


Figure 2: Examples of multiview datasets and the representations learned from them. Dotted lines represent the contrastive objective that encourages congruent views to be brought together in representation-space. The encodings for each view are concatenated to form the full representation for each scene. The video representation diagram depicts an example of our "Core View" loss while the NYU RGBD diagram depicts an example of our "Full Graph" loss (see Section 3.3).

that learn the loss in addition to the representation, or contrastive losses e.g. [21, 24, 46, 2] that take advantage of the co-occurence of multiple views. Another recently introduced novel objective function is instance discrimination [58]. In this work, we compare the two most commonly used objectives: predictive and contrastive.

The prior works most similar to our own (and inspirational to us) are Contrastive Predictive Coding (CPC) [46] and Deep InfoMax [22]. These two methods, like ours, learn representations by contrasting between congruent and incongruent representations of a scene, and are motivated as forms of infomax learning. CPC learns from two views – the past and future – and is applicable to sequential data. Deep Infomax [24] considers the two views to be the input to a neural network and its output. These two methods share the same mathematical objective, but differ in the definition of the views. Our technical method is also highly related, but differs in the following ways: we extend the objective to the case of *more than two* views; and we use a loss function which more closely follows the original method of noise contrastive estimation [21] (See details in Section 3.5). Although CPC, Deep InfoMax, and the present paper are all very similar at the mathematical level, they each explore a different set of view definitions, architectures, and application settings, and each contributes its own unique empirical investigation of this paradigm of representation learning.

Concurrent Work. Concurrently with our work two new papers have appeared on arXiv in the last few weeks: Deep Infomax++[3] and CPC++[22]. The former extends Deep InfoMax to the "multiview" setting where the two views are different data augmentations of a single data point. The latter expands on CPC [46] by using bi-directional prediction and larger scale networks. Both papers achieve impressive performance gains primarily by using larger architectures. In the present paper we have compared only to the original CPC [46] and Deep InfoMax [24] papers, finding that CMC outperforms these methods when all are tested with the same architecture and dataset (in particular, with AlexNet [32], and STL-10 [10]; see Table 1). Given the new results in [22] and [3], we expect that CMC, and perhaps most prior representation learning methods, would perform much more strongly if we scale the architecture and data. A preliminary exploration of this is given in Section 4.1.2, where we achieve 60.1% top-1 accuracy on ImageNet with CMC using Resnet-101, compared to 61.0% with CPC++ using Resnet-170 and 60.2% with Deep Infomax++ using a heavily-customized Resnet. To our best knowledge, CMC, CPC++, and Deep Infomax++ form the first batch of unsupervised/self-supervised learning algorithms which surpass the supervised AlexNet for ImageNet classification.

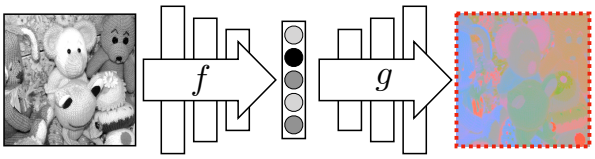
3. Contrastive Multiview Coding

Our goal is to learn representations that capture information shared between multiple sensory views without human supervision. We start by reviewing previous predictive learning (or reconstruction-based learning) methods, and then elaborate on contrastive learning within two views. We show connections to mutual information maximization and extend it to scenarios including more than two views.

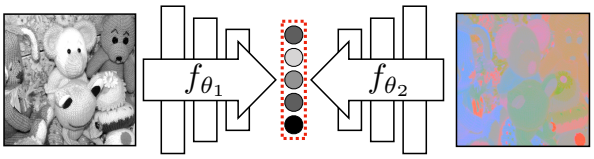
We consider a collection of M views of the data, denoted as $V_1, \ldots, V_M$. For each view V_i, we denote v_i as a random variable representing samples following $v_i \sim \mathcal{P}(V_i)$.

3.1. Predictive Learning

Let V_1 and V_2 represent two views of a dataset. For instance, V_1 might be the luminance of a particular image and V_2 the chrominance. We define the *predictive learning* setup as a deep nonlinear transformation from v_1 to v_2 through latent variables z, as shown in Fig. 3. Formally, $z = f(v_1)$



(a) Predictive learning



(b) Contrastive learning

Figure 3: Predictive Learning vs Contrastive Learning. The predictive learning framework (**Top**) learns representations by predicting one view from another view, with loss measured in the *output* space. Common prediction losses, such as the $\mathcal{L}_1$ and $\mathcal{L}_2$ norms, are *unstructured*, in the sense that they penalize each output dimension independently, perhaps leading to representations that do not capture all the shared information between the views. In contrastive learning (**Bottom**), representations are learnt by contrasting congruent and incongruent views, with loss measured in *representation* space. The red dotted outlines show where the loss function is applied.

and $\hat{v_2} = g(z)$, where f and g represent the encoder and decoder respectively and $\hat{v_2}$ is the prediction of v_2 given v_1. The parameters of the encoder and decoder models are then trained using an objective function that tries to bring $\hat{v_2}$ "close to" v_2. Simple examples of such an objective include the $\mathcal{L}_1$ or $\mathcal{L}_2$ loss functions. Note that these objectives assume independence between each pixel or element of v_1 and v_2, i.e., $p(v_2|v_1) \approx \Pi_i p(v_{2i}|v_1)$, thereby reducing their ability to model correlations or complex structure. The predictive approach has been extensively used in representation learning, for example, colorization [61, 62] and predicting sound from vision [47].

3.2. Contrasting Two Views

The idea behind contrastive learning is learning the parameters of a model by discriminating or comparing between samples from different distributions. Given a dataset of V_1 and V_2 that consists of a collection of samples $\{v_1^i, v_2^i\}_{i=1}^N$, we consider contrasting congruent and incongruent pairs. Formally, we refer to samples from the joint distribution as positives, i.e., $x \sim p(v_1, v_2)$ or $x = \{v_1^i, v_2^i\}$, and samples from the product of marginals as negatives, i.e., $y \sim p(v_1)p(v_2)$ or $y = \{v_1^i, v_2^j\}$.

We learn a score function $h_\theta(\cdot)$ that is higher for positive samples x but lower for negative samples y. Similar to recent setups for contrastive learning [46, 21, 40], we train this function to correctly select a single positive sample x out of a set $S = \{x, y_1, y_2, ..., y_k\}$ which contains k other negatives:

$$\mathcal{L}_{contrast} = -\mathop{\mathbb{E}}_{S}\left[\log \frac{h_\theta(x)}{h_\theta(x) + \sum_{i=1}^k h_\theta(y_i)}\right] \tag{1}$$

To construct S, we simply fix one view and enumerate the other view, leading to the objective we minimize:

$$\mathcal{L}_{contrast}^{V_1,V_2} = -\mathop{\mathbb{E}}_{\{v_1^i, v_2^i\}}\left[\log \frac{h_\theta(\{v_1^i, v_2^i\})}{\sum_{j=1}^N h_\theta(\{v_1^i, v_2^j\})}\right] \tag{2}$$

where N is the number of possible negative samples v_2^j for a given samples v_1^i. In Sec. A.1, we show that the optimal score function h_θ^* for this loss is proportional to the density ratio between joint distribution $p(v_1, v_2)$ and product of marginals $p(v_1)p(v_2)$:

$$h_\theta^*(\{v_1, v_2\}) \quad \propto \quad \frac{p(v_1, v_2)}{p(v_1)p(v_2)} \propto \frac{p(v_1|v_2)}{p(v_1)} \tag{3}$$

In practice, N can be extremely large, and so directly minimizing Eq. 2 is infeasible. In Section 3.5, we show an approximation based on Noise Contrastive Estimation [21] that allows for tractable computation.

We implement the score function $h_\theta(\cdot)$ as a neural network, but other continuous and differentiable parametric functions can be used instead. To extract compact latent representations of v_1 and v_2, we employ two encoders $f_{\theta_1}(\cdot)$ and $f_{\theta_2}(\cdot)$ with parameters θ_1 and θ_2 respectively. The latent representions are extracted as $z_1 = f_{\theta_1}(v_1)$, $z_2 = f_{\theta_2}(v_2)$. On top of these features, the score is computed as the exponential of a bivariate function of z_1 and z_2, e.g., a bilinear function parameterized by W_{12}:

$$h_\theta(\{v_1, v_2\}) = e^{f_{\theta_1}(v_1)^T W_{12} f_{\theta_2}(v_2)} \tag{4}$$

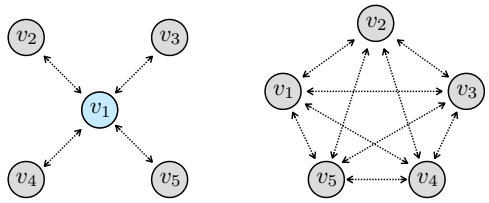
Loss $\mathcal{L}_{contrast}^{V_2,V_1}$ in Eq. 2 treats view V_1 as anchor and enumerates over V_2. Symmetrically, we can get $\mathcal{L}_{contrast}^{V_2,V_1}$ by anchoring at V_2. We add them up as our two-view loss:

$$\mathcal{L}(V_1, V_2) = \mathcal{L}_{contrast}^{V_1,V_2} + \mathcal{L}_{contrast}^{V_2,V_1} \tag{5}$$

After the contrastive learning phase, we use the representations z_1 and/or z_2, depending on our paradigm. This process is visualized in Fig. 2. To evaluate the representations, we add a linear classifier and train the classifier parameters or fine-tune the entire representation with labeled data for a specific task. See Section 4 for details on the experiments.

3.3. More than Two Views

We present more general formulations of Eq. 2 that can handle any number of views. We call them the "core view" and "full graph" paradigms, which offer different tradeoffs



(a) Core View (b) Full Graph

Figure 4: We generalize our CMC to M views following two different formulations: (a) "core view" specifies one view, and all other views are contrasted against that view; (b) "full graph" contrasts every pair and representations for all views are jointly learned. The full graph method is more expensive but learns a more expressive representation.

between efficiency and effectiveness. These formulations are visualized in Fig. 4.

Suppose we have a collection of M views $V_1, \ldots, V_M$. The "core view" formulation sets apart one view that we want to optimize over, say V_1, and builds pair-wise representations between V_1 and each other view $V_j, j > 1$, by optimizing the sum of a set of pair-wise objectives:

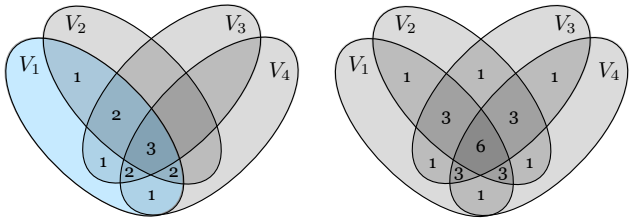
$$\mathcal{L}_C = \sum_{j=2}^{M} \mathcal{L}(V_1, V_j) \tag{6}$$

A second, even more general formulation is the "full graph" where we consider all pairs $(i, j), i \neq j$, and build $\binom{n}{2}$ relationships in all. By involving all pairs, the objective function that we optimize is:

$$\mathcal{L}_F = \sum_{1 \leq i < j \leq M} \mathcal{L}(V_i, V_j) \tag{7}$$

Both these formulations have the effect that information is prioritized in proportion to the number of views that share that information. This can be seen in the information diagrams visualized in Fig. 5. The number in each partition of the diagram indicates how many of the pairwise objectives, $\mathcal{L}(V_i, V_j)$, that partition contributes to. Under both the core view and full graph objectives, a factor, like "presence of dog", that is common to all views will be preferred over a factor that affects fewer views, such as "depth sensor noise".

The computational cost of the full graph formulation is combinatorial in the number of views. However, it is clear from Fig. 5 that this enables the full graph formulation to capture more information between different views, which may prove useful for downstream tasks. For example, the mutual information between V_2 and V_3 or V_2 and V_4 is completely ignored in the core view paradigm (as shown by a 0 count in the information diagram).



(a) Core View (b) Full Graph

Figure 5: We visualize here the information diagrams [1] associated with the core view and full graph paradigms of Fig. 4, for the case of 4 views, which gives a total of 6 learning objectives. The numbers within the regions show how much "weight" the total loss places on each partition of information (i.e. how many of the 6 objectives that partition contributes to). A region with no number corresponds to 0 weight. For example, in the full graph case, the mutual information between all 4 views is considered in all 6 objectives, and hence is marked with the number 6. The full graph view clearly considers more mutual information between views, at the expense of additional computation.

3.4. Connecting to Mutual Information

The contrastive learning paradigm is connected to mutual information maximization between the variable $z_i = f_{\theta i}(v_i)$ and $z_j = f_{\theta j}(v_j)$; mutual information is defined as:

$$I(z_i; z_j) = \mathop{\mathbb{E}}_{z_i, z_j} \left[\frac{p(z_i, z_j)}{p(z_i)p(z_j)} \right] \tag{8}$$

Intuitively, the contrastive loss discriminates between samples from joint distribution and samples from product of marginals, thus maximizing the discrepancy between the distribution of their latent representations. The formal proof is given by [46] and it shows that:

$$I(z_i; z_j) \geq \log(k) - \mathcal{L}_{contrast} \tag{9}$$

where, as above, k is the number of negative pairs in sample set S. Hence minimizing the objective $\mathcal{L}$ maximizes the lower bound on the mutual information $I(z_i; z_j)$, which on the other side is bounded above by $I(v_i; v_j)$ according to data processing inequality. The dependency on k also suggests that using more negative samples can lead to an improved representation; we show that this is indeed the case in Section 4. Finally, we note that recent work [37] shows that the bound in Eq. 9 can be very weak; and finding better estimators of mutual information is an important open problem.

Given three views V_1, V_2, V_3, the mutual information $I(V_1; V_2; V_3)$ is upper bounded by the minimum of the pairwise mutual information [36]. This leads us to the general idea of extending the framework to multiple views by considering *pairwise* mutual information maximization to learn a representation. Given M views, we consider a total of $\binom{M}{2}$

pairs; and learn for each pair of views (V_i, V_j). As above, we call this the "full graph" paradigm. For a large number of views, this paradigm can be expensive to learn; instead we can just learn as in Eq. 6, where V_1 is picked as a special view, and call this the "core view" mode. This corresponds to maximizing mutual information between V_1 and all other views, which can serve as a reasonable proxy to the "full graph" paradigm.

3.5. Approximating Full Softmax

Better representations using $\mathcal{L}_{contrast}^{V_1,V_2}$ in Eq. 2 are learnt by using many negative samples. However, computing the full softmax loss is prohibitively expensive for large N. We alleviate computational overhead in two ways: (a) resorting to Noise-Contrastive Estimation [21] to approximate the full softmax, such a trick was also used in [40] for example; (b) contrasting sub-patches rather than full images to increase the number of negatives inside each batch, similar to [38, 58, 24].

3.5.1 Leveraging Noise-Contrastive Estimation

Given an anchor v_1^i from V_1, the probablity that an atom $v_2 \in \{v_2^j | j = 1, 2, ..., N\}$ from V_2 is the best match of v_1^i, using the score h_θ is given by:

$$p(v_2|v_1^i) = \frac{h_\theta(\{v_1^i, v_2\})}{\sum_{j=1}^{N} h_\theta(\{v_1^i, v_2^j\})} \tag{10}$$

where the normalization factor $Z = \sum_{j=1}^{N} h_\theta(\{v_1^i, v_2^j\})$ is expensive to compute for large N. Here we use $h_\theta(\{v_1^i, v_2^j\}) = \exp(f_{\theta_1}(v_1^i)^T f_{\theta_2}(v_2^j)/\tau)$, where τ modulates the distribution.

Noise-Contrastive Estimation [21] (NCE) is an effective way to estimate *unnormalized* statistical models. NCE fits a density model p to data distributed as (unknown) distribution p_d, by using a binary classifier to distinguish it from noise samples distributed as p_n. To learn $p(v_2|v_1^i)$, we use a binary classifier, which treats v_2^i as the data sample when given v_1^i. The noise distribution $p_n(\cdot|v_1^i)$ we choose here is a uniform distribution over all atoms from V_2, i.e., $p_n(\cdot|v_1^i) = 1/N$. If we sample m noise samples to pair with each data sample, the posterior probability that a given atom v_2 comes from the data distribution is:

$$P(D = 1|v_2; v_1^i) = \frac{p_d(v_2|v_1^i)}{p_d(v_2|v_1^i) + m p_n(v_2|v_1^i)} \tag{11}$$

and we estimate this probability by replacing $p_d(v_2|v_1^i)$ with our model distribution $p(v_2|v_1^i)$. Minimizing the negative log-posterior probability of correct labels D over data and noise samples yields our objective, which is the NCE-based approximation of Eq. 2:

$$\begin{aligned} L_{NCE} &= -\mathop{\mathbb{E}}_{v_1^i \sim p(v_1)} \{ \mathop{\mathbb{E}}_{v_2 \sim p_d(\cdot|v_1^i)} \left[[P(D = 1|v_2; v_1^i)] \right] \\ &+ m \mathop{\mathbb{E}}_{v_2 \sim p_n(\cdot|v_1^i)} \left[1 - P(D = 1|v_2; v_1^i) \right] \} \end{aligned} \tag{12}$$

Memory bank. Following [58], we maintain a memory bank to store latent features for each training sample. Therefore, we can efficiently retrieve m noise samples from the memory bank to pair with each positive sample without recomputing their features. The memory bank is dynamically updated with features computed on the fly.

An alternative to the NCE based approximation above, is to simply do $m + 1$-way softmax classification with m noise samples retrieved from the memory bank. We note that CPC [46] and Deep InfoMax [24] use this $m + 1$ way softmax classification as their ultimate contrastive loss rather than the NCE-based contrastive loss in Eq. 12 (but note that CPC refers to the $m + 1$ approximation as also "based on NCE"). Empirically we have found that the $m + 1$-way softmax classification approach performed worse than our NCE-based approximation, given the same number of noise samples.

3.5.2 Contrasting Sub-patches

Instead of contrasting features from the last layer, patch-based method [24] contrasts feature from the last layer with features from previous layers, hence increasing the number of negative pairs. For instance, we use features from the last layer of f_{θ_1} to contrast with feature points from feature maps produced by the first several conv layers of f_{θ_1}. This is equivalent to contrast between global patch from one view with local patches from the other view. In this fashion, we directly perform $m + 1$ way loss, the same as [46, 24] for a fair comparison in Sec. 4.1.1.

Such patch-based contrastive loss is computed within each mini-batch and does not require a memory bank. Therefore, deploying it in parallel training schemes is easy and flexible. However, patch-based contrastive loss usually yields suboptimal results compared to NCE-based contrastive loss, according to our experiments.

4. Experiments

We extensively evaluate our Contrastive Multiview Coding (CMC) framework on a number of datasets and tasks. We start by evaluating on two established image representation learning benchmarks: STL-10 and Imagenet. We further validate our framework on video representation learning tasks, where we use image and optical flow modalities, as the two views that are jointly learned. The last set of experiments extends our CMC framework to more than two views and provides empirical evidence of it's effectiveness.

4.1. CMC on Images

Given a dataset of RGB images, we convert them to the *Lab* image color space, and split each image into *L* and *ab* channels, as originally proposed in SplitBrain autoencoders [62]. During contrastive learning, L and ab from the same image are treated as the positive pair, and ab channels from other randomly selected images are treated as a negative pair (for a given L). Each split represents a view of the orginal image and is passed through a seprate encoder. This corresponds to the "full graph" model of Eq. 2 with L and ab channels as the two views. As in SplitBrain, we design these two encoders by evenly splitting a given deep network, such as AlexNet [32], into sub-networks across the channel dimension. By concatenating representations layer-wise from these two encoders, we achieve the final representation of an input image. As proposed by previous literature [46, 24, 2], the quality of such a representation is evaluated by freezing the weights of encoder and training linear or non-linear classifiers on top of each layer.

4.1.1 STL-10

STL-10 [10] is an image recognition dataset designed for developing unsupervised or self-supervised learning algorithms. It consists of 100000 unlabeled training 96×96 RGB image samples and 500 labeled samples for each of the 10 classes.

Setup. We adopt the same data augmentation strategy and network architecture as those in DIM [24]. A variant of AlexNet takes as input 64×64 images, which are randomly cropped and horizontally flipped from the original 96×96 size images. For a fair comparison with DIM, we also train our model in a patch-based contrastive fashion during unsupervised pre-training. With the weights of the pre-trained encoder frozen, a two-layer fully connected network with 200 hidden units is trained on top of different layers for 100 epochs to perform 10-way classification. We also investigated the strided crop strategy of CPC [46]. Fixed sized overlapping patches of size 16×16 with an overlap of 8 pixels are cropped and fed into the network separately. This ensures that features of one patch contain minimal information from neighbouring patches; and increases the available number of negative pairs for the contrastive loss. Additionally, we include NCE-based contrastive training and linear classifier evaluation.

Comparison. We compare CMC with the state of the art unsupervised methods in Table 1. Three columns are shown: the conv5 and fc7 columns use respectively these layers of AlexNet as the encoder (again remembering that we split across channels for L and ab views). For these two columns we can compare against the all methods except CPC, since CPC does not report these numbers in their paper [24]. In the Strided Crop setup, we only compare against the approaches that use contrastive learning, DIM and CPC, since this method was only used by those works. We note that in Table 1 for all the methods except SplitBrain, we report numbers are shown in the original paper. For SplitBrain, we reimplemented their model faithfully and report numbers based on our reimplementation (we verified the accuracy of our SplitBrain code by the fact that we get very similar results with our reimpementation as in the original paper [62] for ImageNet experiments, see below).

Method	classifier	conv5	fc7	Strided Crop
AE	MLP	62.19	55.78	-
NAT [7]		64.32	61.43	-
BiGAN [15]		71.53	67.18	-
SplitBrain† [62]		72.35	63.15	-
DIM [24]	MLP	72.57	70.00	76.97
CPC [46]		-	-	77.81
CMC†(Patch)	Linear	76.65	79.25	82.58
CMC†(Patch)	MLP	80.14	80.11	**83.43**
CMC†(NCE)	Linear	80.69	84.73	-
CMC†(NCE)	MLP	**83.03**	**85.06**	-
Supervised		68.70		

Table 1: Classification accuracies on STL-10 by using a two layer MLP as classifier for evaluating the learned representations. For all methods we compare against, we include the numbers that are reported in the DIM [24] paper, except for SplitBrain, which is our reimplementation. Methods marked with † have half the number of parameters because of splitting.

The family of contrastive learning methods, such as DIM, CPC, and CMC, achieve higher classification accuracy than other methods such as SplitBrain that use predictive learning; or BiGAN that use adversarial learning. CMC significantly outperforms DIM and CPC in all cases. We hypothesize that this outperformance results from the modeling of cross-view mutual information, where view-specific noisy details are discarded. Another head-to-head comparison happens between CMC and SplitBrain, both of which modeling images as seprated L and ab streams; we achieve a nearly 8% absolute improvement for conv5 and 17% improvement for fc5. Finally, we notice that the predictive learning methods suffer from a big drop in performance when the encoding layer is switched from conv5 to fc7. On the other hand, the contrastive learning approaches are much more stable across layers, suggesting that the mutual information maximization paradigm learns more semantically meaningful representations shared by the different views. From a practical perspective, this is a significant advantage as the selection of specific layers should ideally not change downstream performance by too much.

	ImageNet Classification Accuracy				
Method	conv1	conv2	conv3	conv4	conv5
ImageNet-Labels	19.3	36.3	44.2	48.3	50.5
Random	11.6	17.1	16.9	16.3	14.1
Data-Init [31]	17.5	23.0	24.5	23.2	20.6
Context [14]	16.2	23.3	30.2	31.7	29.6
Colorization [61]	13.1	24.8	31.0	32.6	31.8
Jigsaw [43]	**19.2**	30.1	34.7	33.9	28.3
BiGAN [15]	17.7	24.5	31.0	29.9	28.0
SplitBrain† [62]	17.7	29.3	35.4	35.2	32.8
Counting [44]	18.0	30.6	34.3	32.5	25.7
Inst-Dis [58]	16.8	26.5	31.8	34.1	35.6
RotNet [18]	18.8	31.7	**38.7**	38.2	36.5
DeepCluster [9]	12.9	29.2	38.2	39.8	36.1
CMC†(Patch)	17.8	30.8	34.2	37.5	38.1
CMC†(NCE)	18.4	**33.5**	38.1	**40.4**	**42.6**

Table 2: Top-1 classification accuracy on 1000 classes of ImageNet [13] with single crop. We compare our CMC method with other unsupervised representation learning approaches by training 1000-way logistic regression classifiers on top of the feature maps of each layer, as proposed by [61]. Methods marked with † only have half the number of parameters compared to others, because of splitting.

4.1.2 ImageNet

ImageNet [13] consists of 1000 image classes and is frequently considered as a testbed for unsupervised representation learning algorithms.

Setup. To compare with other methods, we adopt standard AlexNet and split it into two encoders. Because of splitting, each layer only connects to half of the neurons in the previous layer, and therefore the number of parameters in our model halves. We remove local response layer and add batch normalization to each layer. Two variants of CMC are considered: patch-based and memory-based. For the patch-based CMC model, we set the batch size as 100 and adopt patch features from pool2 layer. For the memory-based CMC model, we adopt ideas from [58] for computing and storing a memory. We retrieve 4096 negative pairs from the memory bank to contrast each positive pair. The training details are present in Sec. B.2.

ImageNet classification task. Following [61], we evaluate task generalization of the learned representation by training 1000-way *linear* classifiers on top of different layers. Table 2 shows the results of comparing the two variants of CMC against other models, both predictive and contrastive. The NCE-based CMC variant is the best among all these methods; futhermore the CMC methods tend to perform better at higher convolutional layers, similar to the other contrastive model Inst-Dis [58]. The NCE-based CMC model consistently performs better than the patch-based model due to the use of NCE as well as many more contrasting images.

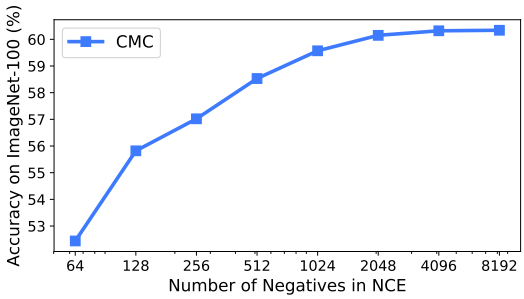


Figure 6: We plot the number of negative examples m in NCE-based contrastive loss against the accuracy for 100 randomly chosen classes of Imagenet 100. It is seen that the accuracy steadily increases with m.

Accuracy (%)	ResNet-50	ResNet-101
Top-1	58.1	60.1
Top-5	81.4	82.8

Table 3: Top-1 and Top-5 classification accuracies on ImageNet. We evaluate our CMC with ResNet-50 and ResNet-101. The unsupervised ResNet-101 has already surpassesed supervised AlexNet.

Effect of the number of negative samples. We investigate the relationship between the number of negative pairs m in NCE-based loss and the downstream classification accuracy on a randomly chosen subset of 100 classes of Imagenet (the same set of classes is used for any number of negative pairs). We train a 100-way linear classifier using CMC pre-trained features with varying number of negative pairs, starting from 64 pairs upto 8192 (in multiples of 2). Fig. 6 shows that the accuracy of the resulting classifier steadily increases but saturates at around 60.3% with $m = 4096$ samples.

CMC with ResNets. We verify the scalability of CMC with larger networks such as ResNets. The results are shown in Table 3, where ResNet-50 and ResNet-101 achieve 58.1% and 60.1% top-1 accuracies, respectively. To our best knowledge, our CMC, together with concurrent CPC++ [22] and Deep InfoMax++ [3], are the first batch of unsupervised/self-supervised learning methods that surpass supervised AlexNet on ImageNet classification task. Note that CPC on ResNet-101 achieved 48.1%, therefore for the same architecture, CMC results in a 12% absolute improvment over CPC, and nearly a 25% relative improvement.

4.2. CMC on Videos

We apply CMC on videos by drawing insight from the two-streams hypothesis [52, 19], which posits that human visual cortex consists of two distinct processing streams: the ventral stream, which performs object recognition, and the dorsal stream, which processes motion. In our formulation, given an image i_t that is a frame centered at time t, the ventral stream associates it with a neighbouring frame

Method	# of Views	UCF-101	HMDB-51
Random	-	48.2	19.5
ImageNet	-	67.7	28.0
VGAN* [56]	2	52.1	-
LT-Motion* [35]	2	53.0	-
TempCoh [41]	1	45.4	15.9
Shuffle and Learn [39]	1	50.2	18.1
Geometry [17]	2	55.1	23.3
OPN [33]	1	56.3	22.1
ST Order [8]	1	58.6	25.0
Cross and Learn [51]	2	58.7	**27.2**
CMC (V)	2	55.3	-
CMC (D)	2	57.1	-
CMC (V+D)	3	**59.1**	26.7

Table 4: Test accuracy (%) on UCF-101 which evaluates *task* transferability and on HMDB-51 which evaluates *task* and *dataset* transferability. Most methods either use single RGB view or additional optical flow view, while VGAN explores sound as the second view. * indicates different network architecture.

i_{t+k}, while the dorsal stream connects it to optical flow f_t centered at t. Therefore, we extract i_t, i_{t+k} and f_t from two modalities as three views of a video; for optical flow we use the TV-L1 algorithm [60]. Two separate contrastive learning objectives are built within the ventral stream (i_t, i_{t+k}) and within the dorsal stream (i_t, f_t). For the ventral stream, the negative sample for i_t is chosen as a random frame from another randomly chosen video; for the dorsal stream, the negative sample for i_t is chosen as the flow corresponding to a random frame in another randomly chosen video.

Pre-training. We train CMC on UCF101 [55] and use two CaffeNets [32] for extracting features from images and optical flows, respectively. In our implementation, f_t represents 10 continuous flow frames centered at t. We use batch size of 128 and contrast each positive pair with 127 negative pairs. CMC is trained with Adam for 300 epochs, with an initial learning rate of 0.001 which is decayed by a factor of 5 after 200 and 250 epochs.

Action recognition. We apply the learn representation to the task of action recognition. The spatial network from [53] is a well-established paradigm for evaluating pre-trained RGB network on action recognition task. We follow the same spirit and evaluate the transferability of our RGB CaffeNet on UCF101 and HMDB51 datasets. We initialize the action recognition CaffeNet up to conv5 using the weights from the pre-trained RGB CaffeNet. The averaged accuracy over three splits is present in Table 4. Unifying both ventral and dorsal streams during pre-training produces higher accuracy for downstream recognition than using only single stream. Increasing the number of views of the data from 2 to 3 (using both streams instead of one) provides a boost for UCF-101. Furthermore, on UCF-101, we outperform all other methods; and on HMDB-51, CMC is second-best in performance.

4.3. Extending CMC to More Views

We further extend our CMC learning framework to multi-view scenarios. We experiment on the NYU-Depth-V2 [42] dataset which consists of 1449 labeled images. We focus more on understanding the behavior and effectiveness of CMC rather than competing with the current state-of-the-art. The views we consider are: luminance (L channel), chrominance (ab channel), depth, surface normal [16], and semantic labels.

Setup. To extract features from each view, we use a neural network with 5 convolutional layers, and 1 fully connected layer. As the size of the dataset is relatively small, we adopt the patch-based contrastive objective to increase the number of negative pairs. Patches with a size of 128×128 are randomly cropped from the original images for contrastive learning (from images of size 480×640). For downstream tasks, we discard the fully connected layers and evaluate using the convolutional layers as a representation. This is because the fully convolutional network can adapt to tasks such as semantic segmentation where input size changes.

4.3.1 Measuring representation quality as the number of views increases

To measure the quality of the learned representation, we consider the task of predicting semantic labels from the representation of L. We follow the *core view paradigm* and use L are the core view, thus learning a set of representations on L by contrasting different views with L. A UNet style architecture [49] is utilized to perform the segmentation task. Contrastive training is performed on the above architecture that is equivalent of the UNet's encoder. After contrastive training is completed, we initialize the encoder weights of the UNet from the L encoder (which are equivalent architectures) and keep them frozen. Only the decoder is trained during this finetuning stage.

Since we use the patch-based contrastive loss, in the 1 view setting case, CMC coincides with DIM [24]. The 2-4 view cases contrast L with ab, and then sequentially add depth and surface normals, but in all cases, the patch based loss is used because the amount of data is small. The semantic labeling results are measured by mean IoU over all classes and pixel accuracy are shown in Fig. 7. We see that the performance steadily improves as new views are added. We have tested different orders of adding the views, and they all follow a similar pattern.

We also compare CMC with two baselines. First, we randomly initialize and freeze the encoder, and we call this the *Random* baseline; it serves as a lower bound on the quality since the representation is just a random projection. Rather

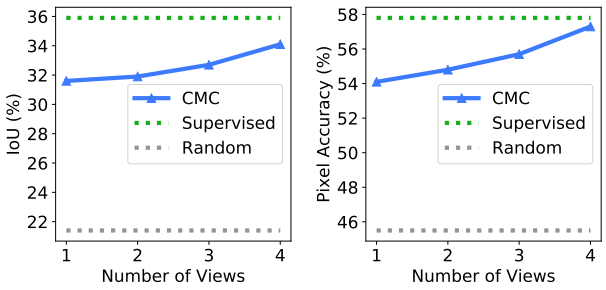


Figure 7: We show the Intersection over Union (IoU) (left) and Pixel Accuracy (right) for the NYU-Depth-V2 dataset, as CMC is trained with increasingly more views from 1 to 4. As more views are added, both these metrics steadily increase. The views are (in order of inclusion): L, ab, depth and surface normals.

	Pixel Accuracy (%)	mIoU (%)
Random	45.5	21.4
CMC (core-view)	57.1	34.1
CMC (full-graph)	57.0	34.4
Supervised	**57.8**	**35.9**

Table 5: Results on the task of predicting semantic labels from **L channel** representation which is learnt using the patch-based contrastive loss and all 4 views. We compare CMC with *Random* and *Supervised* baselines, which serve as lower and upper bounds respectively. Th core-view paradigm refers to Fig. 4(a), and full-view Fig. 4(b).

than freezing the randomly initialized encoder, we could train it jointly with the decoder. This end-to-end *Supervised* baseline serves as an upper bound. The results are presented in Table 5, which shows our CMC produces high quality feature maps even though it's unaware of the downstream task.

4.3.2 Is CMC improving all views?

A desirable unsupervised representation learning algorithm operating on multiple views or modalities should improve the quality of representations for all views. We therefore investigate our CMC framwork beyond L channel. To treat all views fairly, we train these encoders following the full graph paradigm, where each view is contrasted with all other views.

We evaluate the representation of each view v by predicting the semantic labels from only the representation of v, where v is L, ab, depth or surface normals. This uses the full-graph paradigm. As in the previous section, we compare CMC with *Random* and *Supervised* baselines. As shown in Table 6, the performance of the representations learned by CMC using full-graph significantly outperforms that of randomly projected representations, and approaches the performance of the fully supervised representations. Furthermore,

	Metric (%)	L	ab	Depth	Normal
Random	mIoU	21.4	15.6	30.1	29.5
	pix. acc.	45.5	37.7	51.1	50.5
CMC	mIoU	34.4	26.1	39.2	37.8
	pix. acc.	57.0	49.6	**59.4**	57.8
Supervised	mIoU	**35.9**	**29.6**	**41.0**	**41.5**
	pix. acc.	**57.8**	**52.6**	59.1	**59.6**

Table 6: Performance on the task of using single view v to predict the semantic labels, where v can be L, ab, depth or surface normal. Our CMC framework improves the quality of unsupervised representations towards that of supervised ones, for all of views investigated. This uses the full-graph paradigm Fig. 4(b).

	Accuracy on STL-10 (%)	
Views	Predictive	Contrastive
L, Depth	55.5	**58.3**
L, Normal	58.4	**60.1**
L, Seg. Map	57.7	**59.2**
Random	25.2	
Supervised	65.1	

Table 7: We compare predictive learning with contrastive learning by evaluating the learned encoder on unseen dataset and task. The contrastive learning framework consistently outperforms predictive learning.

the full-graph representation provides a good representation learnt for all views, showing the importance of capturing different types of mutual information across views.

4.4. Predictive Learning vs. Contrastive Learning?

While experiments in section 4.1 show that contrastive learning outperforms predictive learning [62] in the context of Lab color space, it's unclear whether such an advantage is due to the natural inductive bias of the task itself. To further understand this, we go beyond chrominance (ab), and try to answer this question when geometry or semantic labels are present.

We consider three view pairs on the NYU-Depth dataset: (1) L and depth, (2) L and surface normals, and (3) L and segmentation map. For each of them, we train two identical encoders for L, one using contrastive learning and the other with predictive learning. We then evaluate the representation quality by training a linear classifier on top of these encoders on the STL-10 dataset.

The comparison results are shown in Table 7, which shows that contrastive learning consistently outperforms predictive learning in this scenario where both the task and the dataset are unknown. We also include "random" and "supervised" baselines similar to that in previous sections. Though in the unsupervised stage we only use 1.3K images from a dataset much different from the target dataset STL-10,

the object recognition accuracy is close to the supervised method, which uses an end-to-end deep network directly trained on STL-10.

Given two views V_1 and V_2 of the data, the predictive learning approach approximately models $p(v_2|v_1)$. Furthermore, losses used typically for predictive learning, such as pixel-wise reconstruction losses usually impose an independence assumption on the modeling: $p(v_2|v_1) \approx \Pi_i p(v_{2i}|v_1)$. On the other hand, the contrastive learning approach by construction does not assume conditional independence across dimensions of v_2. We conjecture that this is one reason for the superior performance of constrastive learning approaches over predictive learning.

5. Conclusion

We have presented a contrastive learning framework which enables the learning of unsupervised representations from multiple views of a dataset. The principle of maximization of mutual information enables the learning of powerful representations. A number of empirical results show that our framework performs well compared to predictive learning and scales with the number of views.

Acknowledgements

Thanks to Devon Hjelm for providing implementation details of Deep InfoMax, Zhirong Wu and Richard Zhang for helpful discussion and comments. This material is based upon work supported by Google Cloud.

References

[1] Information Diagram - Wikipedia. https://en.wikipedia.org/wiki/Information_diagram. 5

[2] S. Arora, H. Khandeparkar, M. Khodak, O. Plevrakis, and N. Saunshi. A theoretical analysis of contrastive unsupervised representation learning. *arXiv preprint arXiv:1902.09229*, 2019. 2, 3, 7

[3] P. Bachman, R. D. Hjelm, and W. Buchwalter. Learning representations by maximizing mutual information across views. *arXiv preprint arXiv:1906.00910*, 2019. 3, 8

[4] A. Bellet, A. Habrard, and M. Sebban. Similarity learning for provably accurate sparse linear classification. *arXiv preprint arXiv:1206.6476*, 2012. 2

[5] Y. Bengio, A. Courville, and P. Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013. 2

[6] A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory*, pages 92–100. ACM, 1998. 2

[7] P. Bojanowski and A. Joulin. Unsupervised learning by predicting noise. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 517–526. JMLR. org, 2017. 7

[8] U. Buchler, B. Brattoli, and B. Ommer. Improving spatiotemporal self-supervision by deep reinforcement learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 770–786, 2018. 9, 14

[9] M. Caron, P. Bojanowski, A. Joulin, and M. Douze. Deep clustering for unsupervised learning of visual features. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 132–149, 2018. 8

[10] A. Coates, A. Ng, and H. Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 215–223, 2011. 3, 7

[11] C. Cortes, M. Mohri, and A. Rostamizadeh. Learning nonlinear combinations of kernels. In *Advances in neural information processing systems*, pages 396–404, 2009. 2

[12] H. E. Den Ouden, P. Kok, and F. P. De Lange. How prediction errors shape perception, attention, and motivation. *Frontiers in psychology*, 3:548, 2012. 2

[13] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. 2009. 8

[14] C. Doersch, A. Gupta, and A. A. Efros. Unsupervised visual representation learning by context prediction. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1422–1430, 2015. 8

[15] J. Donahue, P. Krähenbühl, and T. Darrell. Adversarial feature learning. In *International Conference on Learning Representations*, 2017. 7, 8

[16] D. Eigen and R. Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *Proceedings of the IEEE international conference on computer vision*, 2015. 9

[17] C. Gan, B. Gong, K. Liu, H. Su, and L. J. Guibas. Geometry guided convolutional neural networks for self-supervised video representation learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5589–5597, 2018. 9

[18] S. Gidaris, P. Singh, and N. Komodakis. Unsupervised representation learning by predicting image rotations. *arXiv preprint arXiv:1803.07728*, 2018. 8

[19] M. A. Goodale and A. D. Milner. Separate visual pathways for perception and action. *Trends in neurosciences*, 1992. 8

[20] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014. 1, 2

[21] M. Gutmann and A. Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 297–304, 2010. 2, 3, 4, 6

[22] O. J. Hénaff, A. Razavi, C. Doersch, S. Eslami, and A. v. d. Oord. Data-efficient image recognition with contrastive predictive coding. *arXiv preprint arXiv:1905.09272*, 2019. 3, 8

[23] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507, 2006. 2

[24] R. D. Hjelm, A. Fedorov, S. Lavoie-Marchildon, K. Grewal, A. Trischler, and Y. Bengio. Learning deep representations by mutual information estimation and maximization. In *International Conference on Learning Representations*, 2019. 3, 6, 7, 9, 13

[25] J. Hohwy. *The predictive mind*. Oxford University Press, 2013. 2

[26] A. Hyvärinen, J. Karhunen, and E. Oja. *Independent component analysis*, volume 46. John Wiley & Sons, 2004. 2

[27] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017. 2

[28] P. Isola, D. Zoran, D. Krishnan, and E. H. Adelson. Learning visual groups from co-occurrences in space and time. *arXiv preprint arXiv:1511.06811*, 2015. 2

[29] I. Jolliffe. *Principal component analysis*. Springer, 2011. 2

[30] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 1, 2

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

[32] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. 3, 7, 9, 14

[33] H.-Y. Lee, J.-B. Huang, M. Singh, and M.-H. Yang. Unsupervised representation learning by sorting sequences. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 667–676, 2017. 9, 14

[34] Y. Li, M. Yang, and Z. M. Zhang. A survey of multi-view representation learning. *IEEE Transactions on Knowledge and Data Engineering*, 2018. 2

[35] Z. Luo, B. Peng, D.-A. Huang, A. Alahi, and L. Fei-Fei. Unsupervised learning of long-term motion dynamics for videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2203–2212, 2017. 9

[36] I. Ltkebohle. BWorld Robot Control Software. https://en.wikipedia.org/wiki/Multivariate_mutual_information, 2008. [Online; accessed 19-July-2008]. 5

[37] D. McAllester and K. Statos. Formal limitations on the measurement of mutual information. *arXiv preprint arXiv:1811.04251*, 2018. 5

[38] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013. 6

[39] I. Misra, C. L. Zitnick, and M. Hebert. Shuffle and learn: unsupervised learning using temporal order verification. In *European Conference on Computer Vision*, pages 527–544. Springer, 2016. 9, 14

[40] A. Mnih and K. Kavukcuoglu. Learning word embeddings efficiently with noise-contrastive estimation. In *Advances in neural information processing systems*, pages 2265–2273, 2013. 4, 6

[41] H. Mobahi, R. Collobert, and J. Weston. Deep learning from temporal coherence in video. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 737–744. ACM, 2009. 9

[42] P. K. Nathan Silberman, Derek Hoiem and R. Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012. 9

[43] M. Noroozi and P. Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *European Conference on Computer Vision*, pages 69–84. Springer, 2016. 8

[44] M. Noroozi, H. Pirsiavash, and P. Favaro. Representation learning by learning to count. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5898–5906, 2017. 8

[45] A. v. d. Oord, N. Kalchbrenner, and K. Kavukcuoglu. Pixel recurrent neural networks. *arXiv preprint arXiv:1601.06759*, 2016. 2

[46] A. v. d. Oord, Y. Li, and O. Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018. 2, 3, 4, 5, 6, 7, 13

[47] A. Owens, P. Isola, J. McDermott, A. Torralba, E. H. Adelson, and W. T. Freeman. Visually indicated sounds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2405–2413, 2016. 2, 4

[48] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros. Context encoders: Feature learning by inpainting. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2536–2544, 2016. 2

[49] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015. 9

[50] R. Salakhutdinov and G. Hinton. Deep boltzmann machines. In *Artificial intelligence and statistics*, pages 448–455, 2009. 1, 2

[51] N. Sayed, B. Brattoli, and B. Ommer. Cross and learn: Cross-modal self-supervision. *arXiv preprint arXiv:1811.03879*, 2018. 9, 14

[52] G. E. Schneider. Two visual systems. *Science*, 1969. 8

[53] K. Simonyan and A. Zisserman. Two-stream convolutional networks for action recognition in videos. In *Advances in neural information processing systems*, pages 568–576, 2014. 9

[54] P. Smolensky. Information processing in dynamical systems: Foundations of harmony theory. Technical report, Colorado Univ at Boulder Dept of Computer Science, 1986. 2

[55] K. Soomro, A. R. Zamir, and M. Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012. 9

[56] C. Vondrick, H. Pirsiavash, and A. Torralba. Generating videos with scene dynamics. In *Advances In Neural Information Processing Systems*, pages 613–621, 2016. 9

[57] X. Wang and A. Gupta. Unsupervised learning of visual representations using videos. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2794–2802, 2015. 2

[58] Z. Wu, Y. Xiong, S. X. Yu, and D. Lin. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3733–3742, 2018. 3, 6, 8, 14

[59] C. Xu, D. Tao, and C. Xu. A survey on multi-view learning. *arXiv preprint arXiv:1304.5634*, 2013. 2

[60] C. Zach, T. Pock, and H. Bischof. A duality based approach for realtime tv-l 1 optical flow. In *Joint pattern recognition symposium*, pages 214–223. Springer, 2007. 9

[61] R. Zhang, P. Isola, and A. A. Efros. Colorful image colorization. In *European conference on computer vision*, pages 649–666. Springer, 2016. 2, 4, 8

[62] R. Zhang, P. Isola, and A. A. Efros. Split-brain autoencoders: Unsupervised learning by cross-channel prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1058–1067, 2017. 2, 4, 7, 8, 10

Appendix

A. Proofs

We prove that: (a) the optimal score function $h^*_\theta(\{v_1, v_2\})$ is proportional to density ratio between the joint distribution $p(v_1, v_2)$ and product of marginals $p(v_1)p(v_2)$, as shown in Eq. 3; (b) Minimizing the contrastive loss $\mathcal{L}_{contrast}$ maxmizes a lower bound on the mutual information between two views, as shown in Eq. 9

We will use the most general formula of contrastive loss $\mathcal{L}_{contrast}$ shown in Eq. 1 for our derivation. But we note that replacing $\mathcal{L}_{contrast}$ with $\mathcal{L}^{V_1,V_2}_{contrast}$ is straightforward. The overall proof follows a similar derivation introduced in [46].

A.1. Score function as density ratio estimator

We first show that the optimal score function $h^*_\theta(\{v_1, v_2\})$ that minimizes Eq. 1 is proportional to the density ratio between joint distribution and product of marginals, shown as Eq. 3. For notation convenience, we denote $p(v_1, v_2)$ as data distribution $p_d(\cdot)$ and $p(v_1)p(v_2)$ as noise distribution $p_n(\cdot)$. The loss in Eq. 1 is indeed a cross-entropy loss of classifying the correct positive pair out from the given set S. Without loss of generality, we assume the first pair (v^0_1, v^0_2) in S is positive or congruent and all others $(v^i_1, v^i_2), i = 1, 2, ..., k$ are negative or incongruent. The optimal probability for the loss, $p(pos = 0|S)$, should depict the fact that (v^0_1, v^0_2) comes from the data distribution $p_d(\cdot)$ while all other pairs come from the noise distribution $p_n(\cdot)$. Therefore,

$$
\begin{aligned}
p(pos = 0|S) &= \frac{p_d(v^0_1, v^0_2)\prod_{i=1}^k p_n(v^i_1, v^i_2)}{\sum_{j=0}^k p_d(v^j_1, v^j_2)\prod_{i\neq j} p_n(v^i_1, v^i_2)} \\
&= \frac{p(v^0_1, v^0_2)\prod_{i=1}^k p(v^i_1)p(v^i_2)}{\sum_{j=0}^k p(v^j_1, v^j_2)\prod_{i\neq j} p(v^i_1)p(v^i_2)} \\
&= \frac{\frac{p(v^0_1, v^0_2)}{p(v^0_1)p(v^0_2)}}{\sum_{j=0}^k \frac{p(v^k_1, v^k_2)}{p(v^k_1)p(v^k_2)}}
\end{aligned}
$$

where we plug in the definition of $p_d(\cdot)$ and $p_n(\cdot)$, and divide $\prod_{i=0}^k p(v^i_1)p(v^2_2)$ for both the numerator and denominator. By comparing above equation with the loss function in Eq. 1, we can see that the optimal score function $h^*_\theta(\{v_1, v_2\})$ is proportional to the density ratio $\frac{p(v_1,v_2)}{p(v_1)p(v_2)}$. The above derivation is agnostic to which layer the score function starts from, e.g., h can be defined on either the raw input (v_1, v_2) or the latent representation (z_1, z_2). As we care more about the property of the latent representation, for the following derivation we will use $h^*_{W_{12}}(\{z_1, z_2\})$, which is proportional to $\frac{p(z_1,z_2)}{p(z_1)p(z_2)}$.

A.2. Maximizing lower bound on MI

Now we substitute the score function in Eq. 1 with the above density ratio, and the optimal loss objective $\mathcal{L}^{opt}_{contrast}$ becomes:

$$
\begin{aligned}
\mathcal{L}^{opt}_{contrast} &= -\mathop{\mathbb{E}}_S \log\left[\frac{h^*_{W_{12}}(\{z^0_1, z^0_2\})}{\sum_{i=0}^k h^*_{W_{12}}(\{z^i_1, z^i_2\})}\right] \\
&= -\mathop{\mathbb{E}}_S \log\left[\frac{\frac{p(z^0_1,z^0_2)}{p(z^0_1)p(z^0_2)}}{\sum_{i=0}^k \frac{p(z^i_1,z^i_2)}{p(z^i_1)p(z^i_2)}}\right] \\
&= \mathop{\mathbb{E}}_S \log\left[1 + \frac{p(z^0_1)p(z^0_2)}{p(z^0_1, z^0_2)}\sum_{i=1}^k \frac{p(z^i_1, z^i_2)}{p(z^i_1)p(z^i_2)}\right] \\
&\approx \mathop{\mathbb{E}}_S \log\left[1 + \frac{p(z^0_1)p(z^0_2)}{p(z^0_1, z^0_2)} k \mathop{\mathbb{E}}_{z_1}\left[\frac{p(z_1|z_2)}{p(z_1)}\right]\right] \\
&= \mathop{\mathbb{E}}_S \log\left[1 + \frac{p(z^0_1)p(z^0_2)}{p(z^0_1, z^0_2)} k\right] \\
&\geq \log(k) - \mathop{\mathbb{E}}_S \log\left[\frac{p(z^0_1, z^0_2)}{p(z^0_1)p(z^0_2)}\right] \\
&= \log(k) - \mathop{\mathbb{E}}_{(z_1,z_2)\sim p_{z_1,z_2}(\cdot)} \log\left[\frac{p(z_1, z_2)}{p(z_1)p(z_2)}\right] \\
&= \log(k) - I(z_1; z_2)
\end{aligned}
$$

Therefore, for any two views V_i and V_j, we have $I(z_i; z_j) \geq \log(k) - \mathcal{L}^{opt}_{contrast}(V_i, V_j)$. As the k increases, the approximation step becomes more accurate. Given any k, minimizing $\mathcal{L}_k(V_i, V_j)$ maximizes the lower bound on the mutual

Half of AlexNet[32] for STL-10					
Layer	**X**	**C**	**K**	**S**	**P**
data	64	*	–	–	–
conv1	64	48	3	1	1
pool1	31	48	3	2	0
conv2	31	96	3	1	1
pool2	15	96	3	2	0
conv3	15	192	3	1	1
conv4	15	192	3	1	1
conv5	15	96	3	1	1
pool5	7	96	3	2	0
fc6	1	2048	7	1	0
fc7	1	2048	1	1	0
fc8	1	64	1	1	0

Table 8: **The variant of AlexNet architecture used in our CMC for STL-10 (only half is present here due to splitting)**. **X** spatial resolution of layer, **C** number of channels in layer; **K** `conv` or `pool` kernel size; **S** computation stride; **P** padding; * channel size is dependent on the input source, e.g. 1 for L channel and 2 for ab channel.

information $I(z_i; z_j)$. We should note that increasing k to infinity does not always lead to a higher lower bound. While $\log(k)$ increases with a larger k, the optimization problem becomes harder and $\mathcal{L}_k(V_i, V_j)$ also increases.

B. Implementation Details

B.1. STL-10

For a fair comparison with DIM [24] and CPC [46], we adopt the same architecture as that used in DIM and split it into two encoders, each shown as in Table 8. For the implementation of the score function, we adopt similar "encoder-and-dot-product" strategy, which is tantamount to a bilinear model.

In the patch-based contrastive learning stage, we use Adam optimizer with an initial learning rate of 0.001, $\beta_1 = 0.5$, $\beta_2 = 0.999$. We train for a total of 200 epochs with learning rate decayed by 0.2 after 120 and 160 epochs. In the non-linear classifier evaluation stage, we use the same optimizer setting. For the NCE-based contrastive learning stage, we train for 200 epochs with the learning rate initialized as 0.03 and further decayed by 10 for every 40 epochs after the first 120 epochs. The temperature τ is set as 0.1. In general, $\tau \in [0.05, 0.2]$ works reasonably well.

B.2. ImageNet

For patch-based contrastive loss, we use the same optimizer setting as in Sec. B.1 except that the learning rate is initialized as 0.01.

For NCE-basd contrastive loss in both full ImageNet and ImageNet100 experiments present in Sec. 4.1.2, the encoder architecture used for either L or ab channels is shown in Table 9. In the unsupervised learning stage, we use SGD to train the network for a total of 400 epochs. The temperature τ is set as 0.07 by following previous work [58]. The learning rate is initialized as 0.03 with a decay of 10 for every 50 epochs after the first 250 epochs. Weight decay is set as 10^{-4} and momentum is kept as 0.9. For the linear classification stage, we train for 160 epochs. The learning rate is initialized as 0.1 and decayed by 0.2 every 20 epochs after the first 100 epochs. We set weight decay as 0 and momentum as 0.9.

Half of AlexNet[32] for ImageNet					
Layer	**X**	**C**	**K**	**S**	**P**
data	224	*	–	–	–
conv1	55	48	11	4	2
pool1	27	48	3	2	0
conv2	27	128	5	1	2
pool2	13	128	3	2	0
conv3	13	192	3	1	1
conv4	13	192	3	1	1
conv5	13	128	3	1	1
pool5	6	128	3	2	0
fc6	1	2048	6	1	0
fc7	1	2048	1	1	0
fc8	1	128	1	1	0

Table 9: **AlexNet architecture used in CMC for ImageNet (only half is present here due to splitting)**. **X** spatial resolution of layer, **C** number of channels in layer; **K** `conv` or `pool` kernel size; **S** computation stride; **P** padding; * channel size is dependent on the input source, e.g. 1 for L channel and 2 for ab channel.

B.3. UCF101 and HMDB51

Following previous work [39, 33, 51, 8], we use CaffeNet for the video experiments. We tailor the network and use features from the fc6 layer for contrastive learning. Dropout of 0.5 is used to alleviate overfitting.

B.4. NYU Depth-V2

While experimenting with different views on NYU Depth-V2 dataset, we encode the features from patches with a size of 128×128. The detailed architecture is shown in Table 10. In the unsupervised training stage, we use Adam optimizer with an initial learning rate of 0.001, $\beta_1 = 0.5$, $\beta_2 = 0.999$. We train for a total of 3000 epochs with learning rate decayed by 0.2 after 2000, 2400, and 2800 epochs. For the downstream semantic segmentation task, we use the same optimizer setting but train for fewer epochs. We only train 200 epochs for CMC pre-trained models, and train 1000 epochs for the *Random* and *Supervised* baselines until con-

vergence. For the classification task evaluated on STL-10, we use the same optimizer setting as in Sec. B.1.

Encoder Architecture on NYU					
Layer	**X**	**C**	**K**	**S**	**P**
data	128	*	–	–	–
conv1	64	64	8	2	3
pool1	32	64	2	2	0
conv2	16	128	4	2	1
conv3	8	256	4	2	1
conv4	8	256	3	1	1
conv5	4	512	4	2	1
fc6	1	512	4	1	0
fc7	1	256	1	1	0

Table 10: **Encoder architecture used in our CMC for playing with different views on NYU Depth-V2**. **X** spatial resolution of layer, **C** number of channels in layer; **K** `conv` or `pool` kernel size; **S** computation stride; **P** padding; * channel size is dependent on the input source, e.g. 1 for L, 2 for ab, 1 for depth, 3 for surface normal, and 1 for segmentation map.