Supplementary Material: Contrastive Multiview Coding

Yonglong Tian[1], Dilip Krishnan[2], and Phillip Isola[1]

[1] MIT CSAIL
[2] Google Research

A Contrastive Loss

A.1 NCE approximation for high-dimensional softmax corss-entropy

In addition to the subsampled $(m+1)$-way softmax cross-entropy, Noise-Contrastive Estimation (NCE [9]) is another way to approximate the N-way (N is the dataset size) softmax cross entropy full softmax in Eqn 2. Compared with the $(m+1)$-way softmax cross-entropy, NCE is computationally faster and may result in slightly worse performance in standard linear evaluation. It has been used in [15,20][3]. We depict its general idea as below.

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:

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

where the normalization factor $Z = \sum_{j=1}^{N} h_\theta(\{v_1^i, v_2^j\})$ is expensive to compute for large N.

NCE [9] 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 (or positive) 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., $p_n(\cdot|v_1^i) = p_n(\cdot) = 1/N$. 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:

$$P(D = 1|v_2; v_1^i) = \frac{p_d(v_2|v_1^i)}{p_d(v_2|v_1^i) + m p_n(v_2|v_1^i)} \tag{2}$$

and we estimate this probability by replacing $p_d(v_2|v_1^i)$ with our *unnormalized* model distribution $h_\theta(v_1^i, v_2)/Z_0$, where Z_0 is a constant estimated from the

[3] Confusingly, the literature has previously referred to Eqn.2 as "InfoNCE" [18]. Our NCE approximation *does not* refer to the allusion to NCE in the name "InfoNCE". Rather we are here describing an NCE approximation to the "InfoNCE" softmax objective.

first batch. Minimizing the negative log-posterior probability of correct labels D over data and noise samples yields our final objective, which is the NCE-based approximation of Eq. 2 ($\hat{p}$ is the empirical data distribution):

$$L_{NCE} = - \mathop{\mathbb{E}}_{v_1^i \sim \hat{p}(v_1)} \Big\{ \mathop{\mathbb{E}}_{v_2 \sim \hat{p}(\cdot|v_1^i)} \big[\log(P(D=1|v_2;v_1^i))\big] + m \mathop{\mathbb{E}}_{v_2 \sim p_n(\cdot|v_1^i)} \big[\log(P(D=0|v_2;v_1^i))\big] \Big\} \quad (3)$$

A.2 Contrasting Sub-patches

Instead of contrasting features from the last layer, patch-based method [10] 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_2}. 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 softmax classification, the same as [18,10] for a fair comparison in Sec. C.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.

B 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. 5; (b) Minimizing the contrastive loss $\mathcal{L}_{contrast}$ maxmizes a lower bound on the mutual information between two views, as shown in Eq. 6.

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 $\mathcal{L}_{contrast}^{V_1,V_2}$ is straightforward. The overall proof follows a similar derivation introduced in [18].

B.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. 5. 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_1^0, v_2^0) in S is positive or congruent and all others $(v_1^i, v_2^i), i = 1, 2, ..., k$ are negative or incongruent. The optimal probability for the loss, $p(pos = 0|S)$, should depict the fact that (v_1^0, v_2^0) comes from the data distribution $p_d(\cdot)$ while all other pairs come from the noise distribution $p_n(\cdot)$. Therefore,

$$\begin{aligned} p(pos=0|S) &= \frac{p_d(v_1^0, v_2^0)\prod_{i=1}^{k} p_n(v_1^i, v_2^i)}{\sum_{j=0}^{k} p_d(v_1^j, v_2^j)\prod_{i\neq j} p_n(v_1^i, v_2^i)} \\ &= \frac{p(v_1^0, v_2^0)\prod_{i=1}^{k} p(v_1^i)p(v_2^i)}{\sum_{j=0}^{k} p(v_1^j, v_2^j)\prod_{i\neq j} p(v_1^i)p(v_2^i)} \\ &= \frac{\frac{p(v_1^0, v_2^0)}{p(v_1^0)p(v_2^0)}}{\sum_{j=0}^{k}\frac{p(v_1^k, v_2^k)}{p(v_1^k)p(v_2^k)}} \end{aligned}$$

where we plug in the definition of $p_d(\cdot)$ and $p_n(\cdot)$, and divide $\prod_{i=0}^{k} p(v_1^i)p(v_2^2)$ 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 $h^*(\{z_1, z_2\})$, which is proportional to $\frac{p(z_1,z_2)}{p(z_1)p(z_2)}$.

B.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 $\mathcal{L}_{contrast}^{opt}$ becomes:

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

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 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.

C Additional Experiments

C.1 CMC on STL-10

STL-10 STL-10 [4] 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 [10]. 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 [18]. 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 [10]. 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 [23] for ImageNet experiments, see below).

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

Method	classifier	conv5	fc7	Strided Crop
AE	MLP	62.19	55.78	-
NAT [1]		64.32	61.43	-
BiGAN [7]		71.53	67.18	-
SplitBrain† [23]		72.35	63.15	-
DIM [10]	MLP	72.57	70.00	78.21
CPC [18]		-	-	77.81
CMC†(Patch)	Linear	76.65	79.25	82.58
CMC†(Patch)	MLP	80.14	80.11	**83.43**
CMC†(NCE)	Linear	83.28	86.66	-
CMC†(NCE)	MLP	**84.64**	**86.88**	-
Supervised	68.70			

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

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.

In this experiments we used AlexNet as backbone. Switching to more powerful networks such as ResNets is likely to further improve the representation quality.

C.2 CMC on ImageNet with AlexNet

ImageNet [5] consists of 1000 image classes and is frequently considered as a testbed for unsupervised representation learning algorithms.

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. For the memory-based CMC model, we adopt ideas from [20] for computing and storing a memory. We retrieve 4096 negative pairs from the memory bank to contrast each positive pair (the effect of the number of negatives is shown in Sec. C.3). The training details are present in Sec. D.2.

	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 [11]	17.5	23.0	24.5	23.2	20.6
Context [6]	16.2	23.3	30.2	31.7	29.6
Colorization [22]	13.1	24.8	31.0	32.6	31.8
Jigsaw [16]	19.2	30.1	34.7	33.9	28.3
BiGAN [7]	17.7	24.5	31.0	29.9	28.0
SplitBrain† [23]	17.7	29.3	35.4	35.2	32.8
Counting [17]	18.0	30.6	34.3	32.5	25.7
Inst-Dis [20]	16.8	26.5	31.8	34.1	35.6
RotNet [8]	18.8	31.7	38.7	38.2	36.5
DeepCluster [3]	12.9	29.2	38.2	39.8	36.1
AET [21]	**19.3**	32.8	**40.6**	39.7	37.7
CMC†($\{Y, DbDr\}$)	18.3	**33.7**	38.3	**40.5**	**42.8**

Table 2: Top-1 classification accuracy on 1000 classes of ImageNet [5] 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 [22]. Methods marked with † only have half the number of parameters compared to others, because of splitting.

Table 2 shows the results of comparing the CMC against other models, both predictive and contrastive. Our CMC is the best among all these methods; futhermore CMC tends to perform better at higher convolutional layers, similar to another contrasting-based model Inst-Dis [20].

C.3 Number of negatives

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. 1 shows that the accuracy of the resulting classifier steadily increases but saturates at around 60.3% with $m = 4096$ samples. We used AlexNet and the NCE approximation in this study ((m+1)-way softmax cross entropy, a.k.a. InfoNCE, also follow a similar trend).

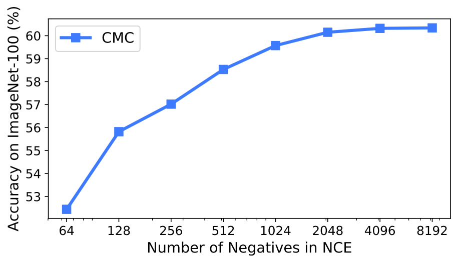

Fig. 1: We plot the number of negative examples m in NCE-based contrastive loss against the accuracy for 100 randomly chosen classes of Imagenet 100. It is seen that the accuracy steadily increases with m.

D Implementation Details

D.1 STL-10

For a fair comparison with DIM [10] and CPC [18], we adopt the same architecture as that used in DIM and split it into two encoders, each shown as in Table 3. 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 320 epochs with the learning rate initialized as 0.03 and further decayed by 10 for every 40 epochs after the first 200 epochs. The temperature τ is set as 0.1. In general, $\tau \in [0.05, 0.2]$ works reasonably well.

D.2 ImageNet

For patch-based contrastive loss, we use the same optimizer setting as in Sec. D.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. C.3, the encoder architecture used for either L or ab channels is shown in Table 4. In the unsupervised learning stage of AlexNet, we use SGD to train the network for a total of 200 epochs. The temperature τ is set as 0.07 by following previous work [20]. The learning rate is initialized as 0.03 with a decay of 10 for every 40 epochs after the first 120 epochs. Weight decay is set as 10^{-4} and momentum is kept as 0.9. For the linear classification stage, we train for 100 epochs. The learning rate is initialized as 0.1 and decayed by 0.2 every 15 epochs after the first 60 epochs. We set weight decay as 0 and momentum as 0.9.

For ResNets in CMC stage, instead of using step decay, we choose cosine annealing to gradually decrease the learning rate. In the linear evaluation stage,

Half of AlexNet[12] 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 3: **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.

we train for 100 epochs. The learning rate is initialized as 30 for ResNet-50 and ResNet-101, and 50 for ResNet-50 x2. It is decayed by 0.2 every 15 epochs after the first 60 epochs. We set weight decay as 0 and momentum as 0.9.

D.3 UCF101 and HMDB51

Following previous work [14,13,19,2], 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.

D.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 5. 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 convergence. For the classification task evaluated on STL-10, we use the same optimizer setting as in Sec. D.1 to report numbers.

Half of AlexNet[12] 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 4: **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.

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 5: **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.

References

1. Bojanowski, P., Joulin, A.: Unsupervised learning by predicting noise. In: ICML (2017) 5
2. Buchler, U., Brattoli, B., Ommer, B.: Improving spatiotemporal self-supervision by deep reinforcement learning. In: ECCV (2018) 8
3. Caron, M., Bojanowski, P., Joulin, A., Douze, M.: Deep clustering for unsupervised learning of visual features. In: ECCV (2018) 6
4. Coates, A., Ng, A., Lee, H.: An analysis of single-layer networks in unsupervised feature learning. In: AISTATS (2011) 4
5. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: CVPR (2009) 5, 6
6. Doersch, C., Gupta, A., Efros, A.A.: Unsupervised visual representation learning by context prediction. In: CVPR (2015) 6
7. Donahue, J., Krähenbühl, P., Darrell, T.: Adversarial feature learning. In: ICLR (2017) 5, 6
8. Gidaris, S., Singh, P., Komodakis, N.: Unsupervised representation learning by predicting image rotations. In: ICLR (2018) 6
9. Gutmann, M., Hyvärinen, A.: Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In: AISTATS (2010) 1
10. Hjelm, R.D., Fedorov, A., Lavoie-Marchildon, S., Grewal, K., Trischler, A., Bengio, Y.: Learning deep representations by mutual information estimation and maximization. In: ICLR (2019) 2, 4, 5, 7
11. Krähenbühl, P., Doersch, C., Donahue, J., Darrell, T.: Data-dependent initializations of convolutional neural networks. arXiv preprint arXiv:1511.06856 (2015) 6
12. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: NIPS (2012) 8, 9
13. Lee, H.Y., Huang, J.B., Singh, M., Yang, M.H.: Unsupervised representation learning by sorting sequences. In: ICCV (2017) 8
14. Misra, I., Zitnick, C.L., Hebert, M.: Shuffle and learn: unsupervised learning using temporal order verification. In: ECCV (2016) 8
15. Mnih, A., Kavukcuoglu, K.: Learning word embeddings efficiently with noise-contrastive estimation. In: NIPS (2013) 1
16. Noroozi, M., Favaro, P.: Unsupervised learning of visual representations by solving jigsaw puzzles. In: ECCV. Springer (2016) 6
17. Noroozi, M., Pirsiavash, H., Favaro, P.: Representation learning by learning to count. In: ICCV (2017) 6
18. Oord, A.v.d., Li, Y., Vinyals, O.: Representation learning with contrastive predictive coding. arXiv preprint arXiv:1807.03748 (2018) 1, 2, 4, 5, 7
19. Sayed, N., Brattoli, B., Ommer, B.: Cross and learn: Cross-modal self-supervision. arXiv preprint arXiv:1811.03879 (2018) 8
20. Wu, Z., Xiong, Y., Yu, S.X., Lin, D.: Unsupervised feature learning via non-parametric instance discrimination. In: CVPR (2018) 1, 5, 6, 7
21. Zhang, L., Qi, G.J., Wang, L., Luo, J.: Aet vs. aed: Unsupervised representation learning by auto-encoding transformations rather than data. In: CVPR (2019) 6
22. Zhang, R., Isola, P., Efros, A.A.: Colorful image colorization. In: ECCV. Springer (2016) 6
23. Zhang, R., Isola, P., Efros, A.A.: Split-brain autoencoders: Unsupervised learning by cross-channel prediction. In: CVPR (2017) 4, 5, 6