Unsupervised Visual Representation Learning by Online Constrained K-Means Supplementary

Qi Qian[1] Yuanhong Xu[2] Juhua Hu[3] Hao Li[2] Rong Jin[1]
[1]Alibaba Group, Bellevue, WA, 98004, USA
[2]Alibaba Group, Hangzhou, China
[3]School of Engineering and Technology, University of Washington, Tacoma, WA 98402, USA
{qi.qian, yuanhong.xuyh, lihao.lh, jinrong.jr}@alibaba-inc.com, juhuah@uw.edu

1. Theoretical Analysis

1.1. Proof of Theorem 1

Proof. Let the Lagrangian function at the i-th iteration be

$$\mathcal{L}_i(\mu_i, \rho^{i-1}) = \sum_k s_{i,k}\mu_{i,k} + \sum_k \rho_k^{i-1}(\mu_{i,k} - \gamma_k/N)$$

where $\mu_i \in \mathcal{R}^K$, and the solution for assignment is

$$\tilde{\mu}_{i,k} = \begin{cases} 1 & k = \arg\max_k s_{i,k} + \rho_k^{i-1} \\ 0 & o.w. \end{cases} \quad (1)$$

It is the maximal solution for the subproblem, and we have

$$\forall \mu_i, \quad \mathcal{L}_i(\mu_i, \rho^{i-1}) \leq \mathcal{L}_i(\tilde{\mu}_i, \rho^{i-1}) \quad (2)$$

where μ_i is an arbitrary assignment and $\tilde{\mu}_i$ is the assignment implied in Eqn. 1. If fixing $\tilde{\mu}_i$ and assuming $\sum_k \gamma_k \leq N$, which is due to the fact that γ_k is the lower-bound for each cluster size, we have the inequality for the arbitrary dual variables ρ from the target convex domain as

$$\mathcal{L}_i(\tilde{\mu}_i, \rho^{i-1}) - \mathcal{L}_i(\tilde{\mu}_i, \rho) = \sum_k (\rho_k^{i-1} - \rho_k)(\tilde{\mu}_{i,k} - \gamma_k/N)$$
$$\leq \frac{\|\rho^{i-1} - \rho\|_2^2 - \|\rho^i - \rho\|_2^2}{2\eta} + \eta \quad (3)$$

Combining Eqns. 2 and 3, we have

$$\mathcal{L}_i(\mu_i, \rho^{i-1}) - \mathcal{L}_i(\tilde{\mu}_i, \rho) \leq \frac{\|\rho^{i-1} - \rho\|_2^2 - \|\rho^i - \rho\|_2^2}{2\eta} + \eta$$

With the constraint $\|\rho\|_1 \leq \tau$ and adding i from 1 to N, we have

$$\sum_{i=1}^N \mathcal{L}_i(\mu_i, \rho^{i-1}) - \mathcal{L}_i(\tilde{\mu}_i, \rho) \leq \frac{\tau^2}{2\eta} + \eta N$$

By setting $\eta = \frac{\tau}{\sqrt{2N}}$, it becomes

$$\sum_{i=1}^N \mathcal{L}_i(\mu_i, \rho^{i-1}) - \mathcal{L}_i(\tilde{\mu}_i, \rho) \leq \tau\sqrt{2N}$$

Taking μ as the optimal solution for the original linear programming problem as μ^*, we have

$$\mathcal{R}(\tilde{\mu}) + \sum_k \rho_k(\gamma_k - \sum_i \tilde{\mu}_{i,k})$$
$$\leq \sum_i \sum_k \rho_k^{i-1}(\gamma_k/N - \mu_{i,k}^*) + \tau\sqrt{2N}$$

Let ρ be the one-hot vector if there is violation.

$$\rho_k = \begin{cases} \tau & k = \arg\max_k \gamma_k - \sum_i \tilde{\mu}_{i,k} \; and \; \mathcal{V}(\tilde{\mu}) > 0 \\ 0 & o.w. \end{cases}$$

Then, we can obtain the relationship between regret and violation as

$$\mathcal{R}(\tilde{\mu}) + \tau\mathcal{V}(\tilde{\mu}) \leq \sum_i \sum_k \rho_k^{i-1}(\gamma_k/N - \mu_{i,k}^*) + \tau\sqrt{2N}$$

By assuming that the instances arrive in a stochastic order, we have $E[\gamma_k/N - \mu_{i,k}^*] \leq 0$ and the bound becomes

$$E[\mathcal{R}(\tilde{\mu})] \leq \tau\sqrt{2N}; \quad \tau E[\mathcal{V}(\tilde{\mu})] \leq \tau\sqrt{2N} - E[\mathcal{R}(\tilde{\mu})]$$

Now, we try to lower bound $\mathcal{R}(\tilde{\mu})$. The following analysis is for the case of $\mathcal{V}(\tilde{\mu}) > 0$. Since the violation is $\mathcal{V}(\tilde{\mu})$, we shrink the current solution $\tilde{\mu}$ by a factor of $\alpha = \min_k \frac{\gamma_k}{\gamma_k + K\mathcal{V}(\tilde{\mu})}$ such that there is no cluster with the number of instances more than γ_k or there is at least $K\mathcal{V}(\tilde{\mu})$ unassigned instances. The shrunk solution with the re-assignment for the extra instances can be a feasible solution for the original assignment problem, so we have

$$\alpha \sum_i \sum_k s_{i,k}\tilde{\mu}_{i,k} \leq \mathrm{OPT}$$

Algorithm 1 Online **Co**nstrained **K**-**Me**ans (CoKe)

Input: Data set $\{\mathbf{x}_i\}_{i=1}^N$, #clusters K, #epochs T, batch size b
Randomly initialize C^0
for $t = 1$ **to** T **do**
  Initialize $C_0^t = C^{t-1}$ and $m = 0$
  for $r = 1$ **to** N/b **do**
    Obtain assignment μ^t
    Update dual variables ρ^i
    Update centers C_{m+b}^t
    $m = m + b$
  end for
end for
return $\{\mu^T, C^T\}$

where OPT denotes the optimal feasible result from μ^*. The lower-bound for $\mathcal{R}(\tilde{\mu})$ is

$$E[\mathcal{R}(\tilde{\mu})] \geq E[(1-\frac{1}{\alpha})OPT] \geq -\frac{KE[\mathcal{V}(\tilde{\mu})]}{\min_k \gamma_k}\text{OPT}$$

Taking it back to the inequality for the violation and let τ be sufficiently large, the bound for the violation is obtained as

$$E[\mathcal{V}(\tilde{\mu})] \leq \frac{1}{1-\frac{K\text{OPT}}{\tau \min_k \gamma_k}}\sqrt{2N}$$

□

1.2. Proof of Corollary 1

Proof. Since $\{\mathbf{x}, C, \mu\}$ are sequentially updated, with $\mathcal{L}'(\mathbf{x}, C, \mu)$ denoting the objective

$$\min_{\mathbf{x},C,\mu\in\Delta} \sum_i \Big((K-1)\sum_{k=1}^K \mu_{i,k}\|\mathbf{x}_i - \mathbf{c}_k\|_2^2 - \sum_{q=1}^K (1-\mu_{i,q})\|\mathbf{x}_i - \mathbf{c}_q\|_2^2\Big) \tag{4}$$

we have $\mathcal{L}'(\mathbf{x}^{t-1}, C^{t-1}, \mu^{t-1}) \geq \mathcal{L}'(\mathbf{x}^t, C^{t-1}, \mu^{t-1})$ and $\mathcal{L}'(\mathbf{x}^t, C^{t-1}, \mu^t) \geq \mathcal{L}'(\mathbf{x}^t, C^t, \mu^t)$. Therefore, the convergence for the bounded loss in Eqn. 4 can be guaranteed if $\mathcal{L}'(\mathbf{x}^t, C^{t-1}, \mu^{t-1}) \geq \mathcal{L}'(\mathbf{x}^t, C^{t-1}, \mu^t)$. Since Theorem 1 indicates that μ^t is a near-optimal solution, it can reduce the loss effectively to make the inequality hold. Theoretically, we can keep μ^{t-1} when μ^t provides no loss reduction to guarantee the convergence. □

Alg. 1 summarizes the proposed online clustering method.

2. Experiments

2.1. Implementation Details

CoKe is learned with LARS optimizer [19], where weight decay is 10^{-6} and momentum is 0.9. Batch size is set to $1,024$, where all experiments except the one with the multi-crop trick can be implemented on a server with 8 GPUs and 16G memory for each GPU. CoKe with two 224×224 crops and six 96×96 crops costs about 20G memory on each GPU and is implemented on a server with 8 GPUs and 32G memory for each GPU. Learning rate is 1.6 with cosine decay and the first 10 epochs are used for warm-up. Batch normalization [13] is synchronized across different GPUs as in [2,3]. Augmentation is important for the performance of unsupervised representation learning [5], and we apply the same augmentation as in others [2, 3] that includes random crop, color jitter, random grayscale, Gaussian blur, and random horizontal flips. ResNet-50 [12] is adopted as the backbone and we apply a 2-layer MLP head to the backbone as suggested in [3, 5] for ablation experiments. The output dimension after MLP projection is 128, which is also the same as benchmark methods [2, 3, 5].

To compare with state-of-the-art methods, we apply more sophisticated settings proposed by recent methods [4, 9], e.g., 1000 epoch training, 3-layer projection MLP and 2-layer prediction MLP. The training epoch for CoKe with multi-view is 800. For Coke with two views, we set $\alpha = 0.2$ and the ablation study for α can be found in the supplementary. The temperature for CoKe with two/multi-view is reduced to 0.05. Other settings, including batch size, dimension of representations, etc. remain the same. Following [6], the linear classifier is optimized with SGD while the batch size and the number of epochs is $1,024$ and 90, respectively. We conduct the ablation study for these additional components.

2.2. Ablation study

2.2.1 Small Batch Training

Since our objective for representation learning is a classification problem, it is insensitive to small batch size. To validate the claim, we have the experiments with the batch size of $\{256, 512, 1024\}$ in Table 1. The learning rate for the batch size 256 and 512 is set to 0.8 and 1.2, respectively. We can observe from Table 1 that the performance

Batch Size	256	512	1,024
Acc%	64.2	64.7	64.5

Table 1. Comparison of different batch size.

of size 256 is similar to that of $1,024$. It confirms that the proposed method is applicable with small batch size. Note that the ablation study has 200 epochs for pre-training, and additional training epochs can further mitigate the gap as illustrated in SwAV [2].

2.2.2 Single View with Moving Average

Here, we investigate the effect of the proposed moving average strategy as a two-stage training scheme. To keep the label vector sparse, we fix the number of non-zero terms in a label vector to be 5 in the second stage, where the performance is quite stable with other values in $\{10, 20, 30\}$. The sparse label will be further smoothed by a Softmax operator as

$$\tilde{\mu}_{i,k} = \begin{cases} \exp(\tilde{\mu}_{i,k}/\lambda')/Z & \tilde{\mu}_{i,k} > 0 \\ 0 & \tilde{\mu}_{i,k} = 0 \end{cases}$$

where $Z = \sum_k I(\tilde{\mu}_{i,k} > 0) \exp(\tilde{\mu}_{i,k}/\lambda')$ and $\lambda' = 0.5$ in all experiments. We also update centers only after each epoch in the second stage as discussed in Sec. 3.2.2.

T'	120	160	200
Acc%	64.3	65.8	65.3

Table 2. Moving average with different T'.

T' is the number of epochs for the first stage and different settings of T' is compared in Table 2. It can be observed that a single stage training strategy achieves 65.3% accuracy, while smoothing the labels and centers in the last 40 epochs can further improve the performance to 65.8%. It shows that the averaging strategy is effective for our framework. However, the performance will degrade if we begin moving average at an early stage as $T' = 120$, which is due to that the model has not been trained sufficiently in the first stage. Given T, we will set T' according to the ratio of $160/200$ for CoKe of single view.

2.2.3 Optimization with Two Views

Now we evaluate the model with sophisticated settings. When optimizing CoKe with two views, we keep the one-stage training scheme but improve pseudo labels as follows

$$\hat{\mathbf{y}}_{i:1}^t = \alpha \tilde{\mathbf{y}}_i^{t-1} + (1-\alpha)\mathbf{p}_{i:2}^{t-1}$$

where

$$p_{i:j,q}^{t-1} = \frac{\exp(\mathbf{x}_i^{j\top}\mathbf{c}_q^{t-1}/\lambda)}{\sum_{k=1}^K \exp(\mathbf{x}_i^{j\top}\mathbf{c}_k^{t-1}/\lambda)}$$

The only parameter in the formulation is α that balances the one-hot label from the last epoch and soft label from the other view. The effect by varying α is summarized in Table 3. It demonstrates that a sufficiently large α, which contains the information from the last epoch, is essential for improving the performance. We will fix $\alpha = 0.2$ for rest experiments.

2.2.4 Optimization with Prediction MLP

We illustrate the different architectures with additional prediction head for existing methods [4, 9] and CoKe in Fig. 1.

α	0.1	0.2	0.3	0.4
Acc%	73.9	74.9	74.5	74.4

Table 3. CoKe of two views with different α.

Most of existing methods constrain that the representation after the prediction head is close to the representation of the other view after the projection head. Unlike those methods, CoKe tries to pull the representation of each instance to its corresponding cluster center. Therefore, both of representations after the projection MLP and prediction MLP can be leveraged for optimization as shown in Fig. 1 (b). Let ℓ_{pred} and ℓ_{proj} denote the classification loss for the representations from prediction and projection MLP, respectively. The final loss can be obtained as

$$\ell = \beta\ell_{\text{pred}} + (1-\beta)\ell_{\text{proj}}$$

The effect of β is summarized in Table 4 for CoKe with single view and two views. Note that the clustering phase including obtaining centers is applied to the representations after projection MLP only.

β	0	0.5	1
single-view	72.3	72.5	72.5
two-view	74.4	74.9	73.3

Table 4. CoKe with different settings of prediction MLP.

From the comparison, it demonstrates that the additional prediction MLP is helpful for learning better representations. Interestingly, if optimizing the loss defined on representations from the prediction only, the performance of CoKe with two views can be degenerated. It may be due to that the dense soft label in two-view optimization is generated from the representation of the projection head. Without the corresponding loss, it is hard to optimize the prediction MLP solely.

2.2.5 Long Training

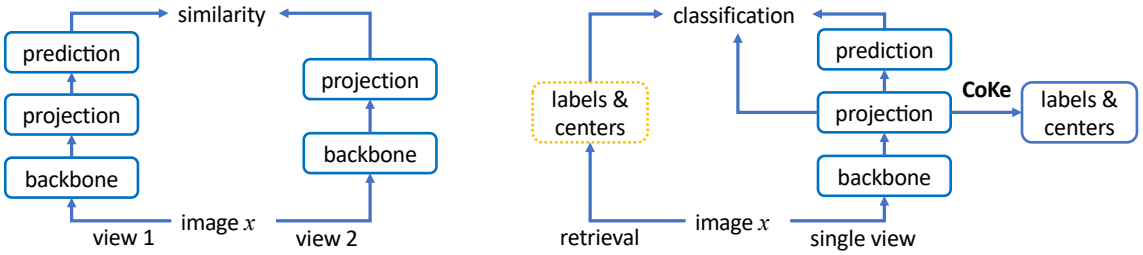
Finally, we compare the performance of 800-epoch training to that of 1000-epoch training in Table 5. Evidently, a longer training still can improve the performance slightly.

#epochs	800	1,000
Acc%	74.5	74.9

Table 5. CoKe of two views with different training epochs.

2.3. Comparison on Downstream Tasks

After evaluating the performance on ImageNet, we apply the pre-trained models on downstream tasks for object



(a) Existing methods with prediction

(b) CoKe with prediction

Figure 1. Illustration of architecture with the additional prediction head. The yellow bounding box denotes the results from the last epoch.

detection, instance segmentation and classification. Four benchmark data sets are included for comparison. Concretely, we fine-tune Faster R-CNN [17] with R50-C4 as the backbone on PASCAL VOC [7] and Mask R-CNN [11] with R50-FPN as the backbone and "$1\times$" training paradigm on COCO [16]. The codebase of MoCo [1] with Detectron2 [18] is adopted. The standard fine-tuning procedure is applied for classification on CIFAR-10 [15] and CIFAR-100 [15].

For object detection and instance segmentation, we follow the settings in MoCo [10] for a fair comparison while only the learning rate is tuned. To obtain the optimal performance for each model, we search the learning rate in $[0.02, 0.12]$ and $[0.01, 0.05]$ with a step size of 0.01 for all methods on VOC and COCO, respectively. For classification, we search the learning rate in $\{1, 10^{-1}, 10^{-2}, 10^{-3}\}$ and weight decay in $\{10^{-5}, 10^{-6}, 0\}$, respectively. Besides, the learning rate for the last fully-connected layer is 10 times larger than others since it is randomly initialized without pre-training. The best performance for baseline methods is reported. CoKe with two-view optimization is evaluated in this subsection.

	VOC	COCO		C10	C100
Methods	Ap_{50}	Ap^{bb}	Ap^{mk}	Acc%	Acc%
Supervised	81.3	38.9	35.4	97.3	86.6
MoCo-v2	<u>83.0</u>	39.6	35.9	97.9	86.1
Barlow Twins	81.5	40.1	36.9	98.0	87.4
BYOL	82.9	<u>40.5</u>	36.9	<u>98.1</u>	<u>87.9</u>
SwAV*	82.1	40.4	<u>37.1</u>	97.7	87.5
DINO*	82.0	40.2	36.8	97.7	87.6
CoKe	<u>83.2</u>	<u>40.9</u>	<u>37.2</u>	<u>98.2</u>	<u>88.2</u>

Table 6. Comparison on downstream tasks. * denotes the usage of the multi-crop training trick. Top-2 best models are underlined.

Table 6 summarizes the standard metric on VOC, COCO and CIFAR. Explicitly, CoKe can outperform the supervised pre-trained model. It implies that an effective pre-trained model can be learned without supervision. Detailed reports on COCO can be found in Tables 7 and 8.

[1] https://github.com/facebookresearch/moco/tree/main/detection

Methods	Ap^{bb}	Ap^{bb}_{50}	Ap^{bb}_{75}
Supervised	38.9	59.6	42.7
MoCo-v2	39.6	60.5	43.4
Barlow Twins	40.1	61.6	43.9
BYOL	40.5	61.8	44.2
SwAV*	40.4	61.8	44.0
DINO*	40.2	61.7	43.8
CoKe	40.9	62.3	44.7

Table 7. Comparison of object detection on COCO.

Methods	Ap^{mk}	Ap^{mk}_{50}	Ap^{mk}_{75}
Supervised	35.4	56.5	38.1
MoCo-v2	35.9	57.4	38.4
Barlow Twins	36.9	58.5	39.6
BYOL	36.9	58.6	39.5
SwAV*	37.1	58.7	39.8
DINO*	36.8	58.3	39.5
CoKe	37.2	59.1	39.9

Table 8. Comparison of instance segmentation on COCO.

2.4. Comparison on Clustering

As a deep clustering method, we compare CoKe to the benchmark clustering algorithms on CIFAR-10 and CIFAR-100 [15] in Tables 9 and 10, respectively. We follow the evaluation protocol in SCAN that trains models on training set and then evaluates the performance on test set with the prediction from the model directly. For CIFAR-100, 20 superclass labels are used for comparison.

CoKe with two-view optimization is adopted for comparison. Compared with ImageNet, the resolution of images in CIFAR is only 32×32. Therefore, we change the parameter of random crop from $[0.08, 1]$ to $[0.3, 1]$ to keep the semantic information of the image. The model is optimized with SGD for 400 epochs. The batch size is 128 and the learning rate is 0.2. Since CIFAR is a balanced data set,

the lower-bound constraint is set to 0.9 and the learning rate for the dual variables is 0.1. Other settings remain the same and we have the same parameters for different data sets. To compare with SCAN, we have the same ResNet-18 as the backbone in CoKe. SCAN has 10 clustering heads with the same number of clusters for multi-clustering. To have explicit multi-clustering, CoKe has 10 clustering heads with different number of clusters. Concretely, we have $[10, 100]$ with a step of 10 for CIFAR-10 and $[20, 200]$ with a step size of 20 for CIFAR-100-20. The head with the target number of clusters is adopted for evaluation. The result averaged over 10 trails is reported.

Methods	CIFAR-10		
	ACC	NMI	ARI
Supervised	93.8	86.2	87.0
DeepCluster [1]	37.4	N/A	N/A
IIC [14]	61.7	51.1	41.1
Pretext [3]+k-means	65.9±5.7	59.8±2.0	50.9±3.7
SCAN [8]	81.8±0.3	71.2±0.4	66.5±0.4
CoKe	85.7±0.2	76.6±0.3	73.2±0.4

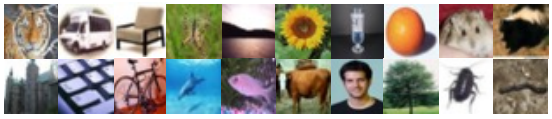
Table 9. Comparison of clustering on CIFAR-10. SCAN is a two-stage method including pre-training and fine-tuning for clustering.

Methods	CIFAR-100-20		
	ACC	NMI	ARI
Supervised	80.0	68.0	63.2
DeepCluster [1]	18.9	N/A	N/A
IIC [14]	25.7	22.5	11.7
Pretext [3]+k-means	39.5±1.9	40.2±1.1	23.9±1.1
SCAN [8]	42.2±3.0	44.1±1.0	26.7±1.3
CoKe	49.7±0.7	49.1±0.4	33.5±0.4

Table 10. Comparison of clustering on CIFAR-100-20.

Tables 9 and 10 show that CoKe as an end-to-end clustering framework can achieve the superior performance without any fine-tuning. On the contrary, SCAN has a two-stage training strategy that learns representations in the first stage with instance discrimination and fine-tunes the model for clustering with different objectives and augmentations in the second stage. Therefore, the representation from the first stage may degenerate the performance of clustering. Finally, Fig. 2 shows the exemplars that are close to cluster centers from CoKe on CIFAR. We can find that CoKe can recover the exact classes on CIFAR-10, which confirms the effectiveness of CoKe for clustering.

(a) CIFAR-10



(b) CIFAR-100-20

Figure 2. Exemplars obtained by CoKe on CIFAR.

References

[1] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *ECCV*, 2018. 5

[2] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. In *NeurIPS*, 2020. 2

[3] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, volume 119, pages 1597–1607, 2020. 2, 5

[4] Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey E. Hinton. Big self-supervised models are strong semi-supervised learners. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *NeurIPS*, 2020. 2, 3

[5] Xinlei Chen, Haoqi Fan, Ross B. Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *CoRR*, abs/2003.04297, 2020. 2

[6] Xinlei Chen, Saining Xie, and Kaiming He. An empirical study of training self-supervised vision transformers. *CoRR*, abs/2104.02057, 2021. 2

[7] Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John M. Winn, and Andrew Zisserman. The pascal visual object classes (VOC) challenge. *Int. J. Comput. Vis.*, 88(2):303–338, 2010. 4

[8] Wouter Van Gansbeke, Simon Vandenhende, Stamatios Georgoulis, Marc Proesmans, and Luc Van Gool. SCAN: learning to classify images without labels. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *ECCV*, volume 12355 of *Lecture Notes in Computer Science*, pages 268–285. Springer, 2020. 5

[9] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H. Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Ávila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, Bilal Piot, Koray Kavukcuoglu, Rémi Munos, and Michal Valko. Bootstrap your own latent - A new approach to self-supervised learning. In *NeurIPS*, 2020. 2, 3

[10] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross B. Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, pages 9726–9735, 2020. 4

[11] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. Mask R-CNN. In *ICCV*, pages 2980–2988, 2017. 4

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 2

[13] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, volume 37, pages 448–456, 2015. 2

[14] Xu Ji, Andrea Vedaldi, and João F. Henriques. Invariant information clustering for unsupervised image classification and segmentation. In *ICCV*, pages 9864–9873. IEEE, 2019. 5

[15] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. 2009. 4

[16] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. In *ECCV*, volume 8693, pages 740–755, 2014. 4

[17] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. Faster R-CNN: towards real-time object detection with region proposal networks. *IEEE Trans. Pattern Anal. Mach. Intell.*, 39(6):1137–1149, 2017. 4

[18] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. https://github.com/facebookresearch/detectron2, 2019. 4

[19] Yang You, Igor Gitman, and Boris Ginsburg. Scaling SGD batch size to 32k for imagenet training. *CoRR*, abs/1708.03888, 2017. 2