Contrastive Consistent Representation Distillation

supplementary material

BMVC 2023 Submission #

Abstract

These items are contained in the supplementary material:
1. Details of projection heads and predictors.
2. Network architectures.
3. Study of momentum.
4. Simple grid search on balancing factors.
5. Study of loss terms
6. Details for experiments on CIFAR100.
7. Standard Deviation of CoCoRD on CIFAR100.
8. Details for linear probing.
9. Details for experiments on ImageNet.
10. Details for experiments on object detection.
11. Other methods
12. Quantitative results on the speed-up, memory reduction and others
13. Theoretical study

1 Details of Projection Heads and Predictors

The projection head is composed of "`fc-bn-relu-fc`", where `fc` denotes a fully-connected layer, `bn` is a batch normalization layer and `relu` is the ReLU activation function. The hidden dimension of the projection head is 2048 and the output dimension is also 2048. **The predictor** consists of "`fc-bn-relu-fc`" which has a bottleneck structure. The hidden dimension of the predictor is 1/4 of the output dimension and the output dimension is 2048.

We also conduct experiments to investigate the effects of projection heads and projectors. We employ resnet32 as the student and resnet110 as the teacher. The experimental results are provided in Table 1. As we can see, CoCoRD needs both projection heads and predictors for boosting the student. The experimental results also demonstrate that it is important to distill in a representation space.

2 Network Architectures

We provide the information about the network architectures we used in experiments.

WRN-d-w represents Wide Residual Network [22] with depth d and width factor w.

	w/ projection heads		w/o projection heads	
	w/ predictor	w/o predictor	w/ predictor	w/o predictor
mean	**74.10**	73.43	72.83	73.34
std	**0.14**	0.04	0.18	0.11

Table 1: The effects of projection heads or predictors. Note that "w/o projection heads" means we remove all the projection heads. The reported results are Top-1 *accuracy* (%) on CIFAR100. The best results are shown in **bold**. Average over 5 runs. *mean* denotes the average and *std* stands for the corresponding standard deviation.

resnet-*d* is CIFAR-style resnet [4] with 3 groups of basic blocks.
ResNet-d represents ImageNet-style ResNet [4] with more channels.
MobileNetV2 [15]. We set the width multiplier to 0.5.
vgg-*d* denotes vgg [16] with depth *d* adapted from its ImageNet counterpart.
ShuffleNetV1 [24] and ShuffleNetV2 [9] are adapted to input of size 32x32.

3 Study of momentum

m_r	**0.9**						0	0.99	0.999	0.9999	1
m_c	0	0.9	0.99	**0.999**	0.9999	1			0.999		
mean	72.87	73.34	73.89	**74.10**	73.70	73.80	73.80	73.65	73.54	72.93	72.86
std	0.32	0.23	0.24	0.14	0.32	0.12	0.25	0.22	0.52	0.40	0.50

Table 2: CIFAR100 test *accuracy* (%) of student resnet32 trained with different combinations of m_c and m_r. The best performance and the corresponding m_c and m_r are shown in **bold**. The teacher is resnet110 and the accuracy of vanilla student and the teacher can be found in Table 6. *mean* denotes the average over 5 runs and *std* stands for the standard deviation.

As shown in Formulas 3 and 4 of the main paper, m_c controls the progressing speed of the teacher projection head f_t^p, while m_r manages the speed of the slow-moving student and its projection head. To investigate the impact of momentum, we employ resnet110 as the teacher to train resnet32 with different m_c and m_r. The results are reported in Table 2. With m_r fixed, a large value of m_c (*e.g.* 0.99 or 0.999) works better than m_c=0, suggesting that consistent representations in the teacher dictionary are beneficial for the distillation. With m_r fixed, CoCoRD consistently performs well when m_c is relatively large (*e.g.* 0.999, 0.9999). This observation indicates CoCoRD benefits from the slow-progressing f_t^p (*i.e.* large m_c).

4 Simple grid search on balancing factors

We conduct experiments to investigate the effects of the three balancing factors λ_{ctr}, λ_{cls} and λ_{pred}. We use resnet32-resnet110 as the student-teacher combination. For experiments on balancing factors, we set τ=0.1, N=2048, m_c=0.999 and m_r=0.9.

To investigate the effects of λ_{pred}, λ_{ctr} and λ_{cls} are set to 1. The results are provided in Table 3. To investigate the effects of λ_{ctr}, λ_{pred} is set to 4 and λ_{cls} is set to 1. The results are shown in Table 4. To investigate the effects of λ_{cls}, λ_{pred} is set to 4 and λ_{ctr} is set to 1. The results are shown in Table 5. Based on the experimental results above, λ_{ctr}=1, λ_{cls}=1 and λ_{pred}=4 are chosen as the default setting for CoCoRD on CIFAR100 and ImageNet.

λ_{pred}	0	0.5	1.0	2.0	**4.0**	8.0
mean	72.92	73.61	73.53	73.92	**74.10**	73.36
std	0.23	0.28	0.31	0.19	**0.14**	0.26

Table 3: The effects of λ_{pred}. We set λ_{ctr} and λ_{cls} to 1. CIFAR100 test *accuracy* (%) is reported. The best performance is shown in **bold**. Average over 5 runs. *mean* denotes the average and *std* stands for the corresponding standard deviation.

λ_{ctr}	0	0.5	**1.0**	2.0	4.0	8.0
mean	71.81	73.12	**74.10**	73.94	73.61	73.23
std	0.42	0.27	**0.14**	0.22	0.29	0.37

Table 4: The effects of λ_{ctr}. We set λ_{pred} to 4 and λ_{cls} to 1. CIFAR100 test *accuracy* (%) is reported. The best performance is shown in **bold**. Average over 5 runs. *mean* denotes the average and *std* stands for the corresponding standard deviation.

5 Details for experiments on CIFAR100

For all experiments on CIFAR100, we use SGD optimizer with momentum 0.9 to train the students with CoCoRD. λ_{ctr}=1, λ_{cls}=1 and λ_{pred}=4. The training batch size is set to 64 and the weight decay is set to $5\text{x}10^{-4}$. For experiments in Table 1 of the main paper, we initialize the learning rate as 0.05 and decay the learning rate by 0.1 at the {150, 180, 210}-*th* epochs.

For experiments in Table 2 of the main paper, we use a learning rate of 0.015 for MobileNetV2, a learning rate of 0.03 for ShuffleNetV1 and ShuffleNetV2 and a learning rate of 0.05 for vgg8 based on simple grid searches. Each learning rate is also multiplied by 0.1 at {150, 180, 210}-*th* epochs. Images of size 32x32 are randomly cropped from zero-padded 40x40 images and are horizontally flipped with a probability of 0.5. The independent transformation for a different view of the input is sampled from this data augmentation.

6 Standard deviation of CoCoRD on CIFAR100

The standard deviation of CoCoRD over 5 runs on CIFAR100 dataset is provided in Table 6 for student and teacher models of the same architecture style, and in Table 7 for student and teacher models of different architecture style.

7 Details for linear probing

For experiments in Table 3 of the main paper, we initialize the learning rate as 0.001. We use SGD optimizer with momentum 0.9 and the weight decay is set to 0. We freeze the features and train a supervised linear classifier on the global average pooling features for 100 epochs.

8 Details for experiments on ImageNet

For experiments on ImageNet with ResNet18 or ResNet50, we follow the standard PyTorch practice but train for 100 epochs in total. The learning rate starts at 0.1 and the batch size is set to 256. Note that we only compute $\mathcal{L}_{pred}$ on ImageNet. λ_{ctr}=1, λ_{cls}=1 and λ_{pred}=4.

λ_{cls}	0.5	**1.0**	2.0	4.0	8.0
mean	73.97	**74.10**	73.03	72.53	71.89
std	0.26	**0.14**	0.31	0.32	0.24

Table 5: The effects of λ_{cls}. We set λ_{pred} to 4 and λ_{ctr} to 1. CIFAR100 test *accuracy* (%) is reported. The best performance is shown in **bold**. Average over 5 runs. *mean* denotes the average and *std* stands for the corresponding standard deviation.

Teacher **Student**	**WRN-40-2** **WRN-16-2**	**WRN-40-2** **WRN-40-1**	**resnet56** **resnet20**	**resnet110** **resnet20**	**resnet110** **resnet32**	**resnet32x4** **resnet8x4**	**vgg13** **vgg8**
Teacher	75.61	75.61	72.34	74.31	74.31	79.42	74.64
Student	73.26	71.98	69.06	69.06	71.14	72.50	70.36
CoCoRD (ours)	75.48 (±0.16)	75.17 (±0.17)	71.74 (±0.11)	72.11 (±0.29)	74.10 (±0.14)	75.29 (±0.07)	73.99 (±0.13)
CoCoRD+KD	75.90 (±0.05)	75.25 (±0.14)	72.09 (±0.31)	72.18 (±0.07)	74.37 (±0.18)	75.42 (±0.16)	74.26 (±0.11)

Table 6: Test accuracy (%) of students on CIFAR100 of CoCoRD. Standard deviation is provided.

Teacher **Student**	**vgg13** **MobileNetV2**	**ResNet50** **MobileNetV2**	**ResNet50** **vgg8**	**resnet32x4** **ShuffleNetV1**	**resnet32x4** **ShuffleNetV2**	**WRN-40-2** **ShuffleNetV1**
Teacher	74.64	79.34	79.34	79.42	79.42	75.61
Student	64.60	64.60	70.36	70.50	71.82	70.50
CoCoRD (ours)	69.86 (±0.22)	70.22 (±0.07)	74.52 (±0.12)	75.99 (±0.12)	77.28 (±0.08)	76.42 (±0.10)
CoCoRD+KD	69.90 (±0.25)	70.30 (±0.19)	74.62 (±0.10)	76.48 (±0.23)	77.39 (±0.04)	76.56 (±0.26)

Table 7: Test accuracy (%) of students on CIFAR100 of CoCoRD. Standard deviation is provided.

9 Details for experiments on Object Detection

We initialize the backbones of the object detection models with the CoCoRD-distilled ResNet50. The teacher is ResNet101 during CoCoRD distillation.

9.1 PASCAL VOC object detection

The detector is Faster R-CNN [13] with a backbone of R50-C4 which is available in Detectron2. The backbone ends with the conv_4 stage and the box prediction head consists of the conv_5 stage (including global pooling) followed by a BN layer. The same setup is applied to all entries in PASCAL VOC detection in Table 7 of the main paper. The detector is fine-tuned on VOC `trainval07+12` for 24k iterations in an end-to-end manner. We evaluate the default VOC metric of AP_{50} and COCO-style metrics of AP and AP_{75}. The evaluation is on the VOC `test2007`. The image is [480, 800] pixels during training and 800 at inference.

9.2 COCO object detection

The detector is Mask R-CNN [6] with R50-C4 backbone. The image is in [640,800] pixels during training and is 800 at inference. We fine-tune the detector on the COCO `train2017` in an end-to-end manner and evaluate on COCO `val2017`. The schedule is 2x as in [19].

10 Other methods

We compare CoCoRD with the following state-of-the-art methods from the literature.

- KD: Distilling the knowledge in a neural network [7]
- FitNet: FitNets: Hints for Thin Deep Nets [14]
- AT: Paying More Attention to Attention: Improving the Performance of Convolutional Neural Networks via Attention Transfer [23]
- SP: Similarity-preserving knowledge distillation [18]
- CC: Correlation congruence for knowledge distillation [12]
- VID: Variational information distillation for knowledge transfer [1]
- RKD: Relational knowledge distillation [10]
- PKT: Learning deep representations with probabilistic knowledge transfer [11]
- AB: Knowledge transfer via distillation of activation boundaries formed by hidden neurons [6]
- FT: Paraphrasing complex network: Network compression via factor transfer [8]
- FSP: A gift from knowledge distillation: Fast optimization, network minimization and transfer learning [21]
- CRD: Contrastive Representation Distillation [17]
- LCKT, WCoRD: Wasserstein contrastive representation distillation [2]
- ReviewKD: Distilling Knowledge via Knowledge Review [3]
- SSKD: Knowledge Distillation Meets Self-Supervision [20]

11 Quantitative results on the achieved speed-up, memory reduction and others

In the following three tables, we provide quantitative results on the achieved speed-up, memory cost reduction, and other quantitative information about the teacher/student (T/S) combinations used on CIFAR100 (in Tabs. 1 and 2) and those T/S combinations used on ImageNet (Tabs. 4 and 8). The results are measured with Intel Core i7-8700 CPU on Ubuntu 20.04 operating system and memory cost is measured by Pytorch Profiler in a forward pass.

Additionally, we compare the size of the teacher dictionary in the proposed CoCoRD with the size of the memory banks in CRD. Note that the keys in CRD memory banks are only 128-d while the keys in the proposed CoCoRD teacher dictionary are 2048-d. Even with higher dimensions of the stored keys, CoCoRD are still more storage efficient.

Combination(T/S)	Inference Latency (ms)	Speed-up	Memory Cost (MB)	Mult-Add	Parameters (K)
WRN-40-2 / WRN-16-2	21.28 / 7.98	62.50%	11.73 / 4.39	327.62M / 101.12M	2255 / 703
WRN-40-2 / WRN-40-1	21.28 / 10.51	50.61%	11.73 / 5.87	327.62M / 83.29M	2255 / 570
resnet56 / resnet20	21.75 / 11.48	47.22%	8.72 / 3.21	125.76M / 40.82M	862 / 278
resnet110 / resnet20	56.09 / 11.48	79.53%	16.97 / 3.21	253.16M / 40.82M	1737 / 278
resnet110 / resnet32	56.09 / 17.01	69.67%	16.97 / 5.05	253.16M / 69.13M	1737 / 473
resnet32x4 / resnet8x4	24.93 / 7.52	69.84%	20.71 / 6.03	1.08G / 177.07M	7434 / 1234
vgg13 / vgg8	8.72 / 3.89	55.39%	4.20 / 2.10	285.2M / 96.33M	9462 / 3965

Table 8: Quantitative results on the achieved speed-up, parameter compression and memory cost reduction. The combinations are from Tab. 1 of the main paper. The inference latency is measured on image of size 32x32.

Combination(T/S)	Inference Latency (ms)	Speed-up	Memory Cost (MB)	Mult-Add	Parameters (K)
Vgg13 / MobileNetV2	8.72 / 14.97	-	4.20 / 3.39	285.2M / 6.54M	9462 / 813
ResNet50 / MobileNetV2	17.01 / 14.97	11.99%	3.63 / 3.39	83.67M / 6.54M	23713 / 813
ResNet50 / vgg8	17.01 / 3.89	77.13%	3.63 / 2.10	83.67M / 96.33M	23713 / 3965
resnet32x4 / ShuffleNetV1	24.93 / 31.84	-	20.71 / 13.90	1.08G / 38.72M	7434 / 949
resnet32x4 / ShuffleNetV2	24.93 / 19.01	23.75%	20.71 / 9.01	1.08G / 44.52M	7434 / 1356
WRN-40-2 / ShuffleNetV1	21.28 / 31.84	-	11.73 / 13.90	327.62M / 38.72M	2255 / 949

Table 9: Quantitative results on the achieved speed-up, parameter compression and memory cost reduction. The combinations are from Tab. 2 of the main paper. The inference latency is measured on image of size 32x32.

Combination(T/S)	Inference Latency (ms)	Speed-up	Memory Cost (MB)	Mult-Add	Parameters (M)
ResNet34 / ResNet18	43.97 / 28.52	35.14%	59.82 / 39.75	3.66G / 1.81G	21.80 / 11.69
ResNet101 / ResNet50	104.37 / 50.84	51.29%	259.72 / 177.83	7.80G / 4.09G	44.55 / 25.56

Table 10: Quantitative results on the achieved speed-up, parameter compression and memory cost reduction. The combinations are from Tabs. 4 and 6 of the main paper. The inference latency is measured on image of size 224x224.

	CRD	CoCoRD	Relative Size
CIFAR100	51.20MB	16.78MB	32.77%
ImageNet	1311.92MB	536.87MB	40.92%

Table 11: Comparison on the size of memory bank(s). Note that there is one teacher dictionary in the proposed CoCoRD while there are two memory banks in CRD.

12 Theoretical study

Given two deep neural networks, a teacher f^T and a student f^S, and let x be the network input. We denote representations at the penultimate layer as $f^T(x)$ and $f^S(x)$, respectively. We would like to bring $f^S(x_i)$ and $f^T(x_i)$ close while pushing apart $f^S(x_i)$ and $f^T(x_j)$ (x_i and x_j represent different training samples).

For clear notation, we define variables S and T for the student representations and the teacher ones of the data, respectively: $x \sim p_(x)$; $S = f^S(x)$; $T = f^T(x)$.

Let us define a distribution q with variable C. The latent variable C decides whether the

tuple $(f^S(x_i), f^T(x_j))$ is drawn form the joint distribution $p(T,S)$ (when C=1) or drawn from the product of marginal distributions $p(S)p(T)$ (when C=0).

$$q(T,S|C=1) = p(T,S), q(T,S|C=0) = p(T)p(S)$$

Suppose we are given 1 congruent pair drawn from the joint distribution (i.e. the same input provided to T and S) for every N incongruent pairs drawn from the product of marginals (independent randomly inputs provided to T and S). Then the priors on the latent C are:

$$q(C=1) = \frac{1}{N+1}, q(C=0) = \frac{N}{N+1}.$$

By Bayes' rule and simple manipulations, the posterior for $C=1$ is given by:

$$q(C=1|T,S) = \frac{q(T,S|C=1)q(C=1)}{q(T,S|C=0)q(C=0)+q(T,S|C=1)q(C=1)} = \frac{p(T,S)}{p(T,S)+Np(T)p(S)}.$$

We can observe a connection with mutual information:

$$\log q(C=1|T,S) = -\log(1+N\frac{p(T)p(S)}{p(T,S)}) \leq -\log(N) + \log\frac{p(T,S)}{p(T)p(S)}.$$

Taking expectation on both sides w.r.t $p(T,S)$ and rearranging gives us:

$$I(T;S) \geq \log(N) + \mathbb{E}_{q(T,S|C=1)} \log q(C=1|T,S),$$

where $I(T;S)$ is the mutual information between the distributions of the teacher and student representations. Though we do not know the true distribution $q(C=1|T,S)$, a neural network can be used to estimate whether a pair comes from the joint distribution or the marginals.

By maximizing KL divergence between the joint distribution $p(T,S)$ and the product of marginal distributions $p(T)p(S)$, we can maximize the mutual information between the student representations and the teacher representations.

References

[1] Sungsoo Ahn, Shell Xu Hu, Andreas Damianou, Neil D Lawrence, and Zhenwen Dai. Variational information distillation for knowledge transfer. In *CVPR*, pages 9163–9171, 2019.

[2] Liqun Chen, Dong Wang, Zhe Gan, Jingjing Liu, Ricardo Henao, and Lawrence Carin. Wasserstein contrastive representation distillation. In *CVPR*, pages 16296–16305, 2021.

[3] Pengguang Chen, Shu Liu, Hengshuang Zhao, and Jiaya Jia. Distilling knowledge via knowledge review. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5008–5017, 2021.

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

[5] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *ICCV*, pages 2961–2969, 2017.

[6] Byeongho Heo, Minsik Lee, Sangdoo Yun, and Jin Young Choi. Knowledge transfer via distillation of activation boundaries formed by hidden neurons. In *AAAI*, pages 3779–3787, 2019.

[7] Geoffrey Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. In *NeurIPS Deep Learning and Representation Learning Workshop*, 2015.

[8] Jangho Kim, SeongUk Park, and Nojun Kwak. Paraphrasing complex network: Network compression via factor transfer. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

[9] Ningning Ma, Xiangyu Zhang, Hai-Tao Zheng, and Jian Sun. Shufflenet v2: Practical guidelines for efficient cnn architecture design. In *ECCV*, pages 116–131, 2018.

[10] Wonpyo Park, Dongju Kim, Yan Lu, and Minsu Cho. Relational knowledge distillation. In *CVPR*, pages 3967–3976, 2019.

[11] Nikolaos Passalis and Anastasios Tefas. Learning deep representations with probabilistic knowledge transfer. In *ECCV*, pages 268–284, 2018.

[12] Baoyun Peng, Xiao Jin, Jiaheng Liu, Dongsheng Li, Yichao Wu, Yu Liu, Shunfeng Zhou, and Zhaoning Zhang. Correlation congruence for knowledge distillation. In *ICCV*, pages 5007–5016, 2019.

[13] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015.

[14] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets. In *ICLR*, 2015.

[15] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *CVPR*, pages 4510–4520, 2018.

[16] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[17] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive representation distillation. In *ICLR*, 2020.

[18] Frederick Tung and Greg Mori. Similarity-preserving knowledge distillation. In *ICCV*, pages 1365–1374, 2019.

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

[20] Guodong Xu, Ziwei Liu, Xiaoxiao Li, and Chen Change Loy. Knowledge distillation meets self-supervision. In *European Conference on Computer Vision (ECCV)*, 2020.

[21] Junho Yim, Donggyu Joo, Jihoon Bae, and Junmo Kim. A gift from knowledge distillation: Fast optimization, network minimization and transfer learning. In *CVPR*, pages 4133–4141, 2017.

[22] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In *BMVC*, pages 87.1–87.12, 2016. URL https://dx.doi.org/10.5244/C.30.87.

[23] Sergey Zagoruyko and Nikos Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. In *ICLR*, 2017.

[24] Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In *CVPR*, pages 6848–6856, 2018.