Contrastive Consistent Representation Distillation

Abstract

These items are contained in the supplementary materials:

1. Details of projection heads and predictors.
2. Network architectures.
3. Study of balancing factors.
4. Implementation details for experiments on CIFAR100.
5. Standard Deviation of CoCoRD on CIFAR100.
6. Implementation details for linear probing.
7. Implementation details for experiments on ImageNet.
8. Implementation details for experiments on object detection.
9. Other methods

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`". The predictor 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.

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 result are shown in **bold**. Average over 5 runs. *mean* denotes the average and *std* stands for the corresponding standard deviation.

	w/ projection heads		w/o projection heads	
	w/ predictor	w/o predictor	w/ predictor	w/o predictor
mean	**74.20**	73.43	72.83	73.34
std	**0.14**	0.04	0.18	0.11

2 Network Architectures

WRN-d-w represents Wide Residual Network [1] with depth d and width factor w.

resnet-d is CIFAR-style resnet [2] with 3 groups of basic blocks. resnet8x4 means a 4x wider resnet8.

ResNet-d represents ImageNet-style ResNet [2] with more channels.

MobileNetV2 [3]. We set the width multiplier to 0.5.

vgg-d denotes vgg [4] with depth d adapted from its ImageNet counterpart.

ShuffleNetV1 [5] and ShuffleNetV2 [6] are adapted to input of size 32x32.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

Table 2: 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_{pred}	0	0.5	1.0	2.0	**4.0**	8.0
mean	72.92	73.61	73.53	73.92	**74.20**	73.36
std	0.23	0.28	0.31	0.19	**0.14**	0.26

Table 3: 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.

λ_{ctr}	0	0.5	**1.0**	2.0	4.0	8.0
mean	71.81	73.12	**74.20**	73.94	73.61	73.23
std	0.42	0.27	**0.14**	0.22	0.29	0.37

3 Study of 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 2. 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 3. 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 4. 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.

4 Implementation Details for Experiments on CIFAR100

For all experiments on CIFAR100, we use SGD optimizer with momentum 0.9 to train the student models 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 5x10^{-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 for data augmentation. The independent transformation for a different view of the input is sampled from this data augmentation.

5 Standard Deviation of CoCoRD on CIFAR100

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

6 Implementation 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 of the models for 100 epochs.

7 Implementation 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.

8 Implementation 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.

8.1 PASCAL VOC Object Detection

The detector is Faster R-CNN [7] 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

Table 4: 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.

λ_{cls}	0.5	**1.0**	2.0	4.0	8.0
mean	73.97	**74.20**	73.03	72.53	71.89
std	0.26	**0.14**	0.31	0.32	0.24

Table 5: Test accuracy (%) of student models on CIFAR100 of CoCoRD. Standard deviation is provided.

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)	74.20 (±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)	74.94 (±0.16)	74.26 (±0.11)

Table 6: Test accuracy (%) of student models 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	69.26 (±0.25)	69.89 (±0.19)	74.62 (±0.10)	76.48 (±0.23)	77.39 (±0.04)	76.56 (±0.26)

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 scale is [480, 800] pixels during training and 800 at inference.

8.2 COCO Object Detection

The detector is Mask R-CNN [8] with R50-C4 backbone. The image scale 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 [9].

9 Other methods

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

- KD: Distilling the knowledge in a neural network [10]
- FitNet: FitNets: Hints for Thin Deep Nets [11]
- AT: Paying More Attention to Attention: Improving the Performance of Convolutional Neural Networks via Attention Transfer [12]
- SP: Similarity-preserving knowledge distillation [13]
- CC: Correlation congruence for knowledge distillation [14]
- VID: Variational information distillation for knowledge transfer [15]
- RKD: Relational knowledge distillation [16]
- PKT: Learning deep representations with probabilistic knowledge transfer [17]
- AB: Knowledge transfer via distillation of activation boundaries formed by hidden neurons [18]
- FT: Paraphrasing complex network: Network compression via factor transfer [19]
- FSP: A gift from knowledge distillation: Fast optimization, network minimization and transfer learning [20]
- NST: Like what you like: Knowledge distill via neuron selectivity transfer [21]
- CRD: Contrastive Representation Distillation [22]
- LCKT, GCKT, WCoRD: Wasserstein contrastive representation distillation [23]
- ReviewKD: Distilling Knowledge via Knowledge Review [24]
- SSKD: Knowledge Distillation Meets Self-Supervision [25]

References

[1] S. Zagoruyko and N. Komodakis, "Wide residual networks," in *BMVC*, pp. 87.1–87.12, 2016.

[2] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, pp. 770–778, 2016.

[3] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *CVPR*, pp. 4510–4520, 2018.

[4] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[5] X. Zhang, X. Zhou, M. Lin, and J. Sun, "Shufflenet: An extremely efficient convolutional neural network for mobile devices," in *CVPR*, pp. 6848–6856, 2018.

[6] N. Ma, X. Zhang, H.-T. Zheng, and J. Sun, "Shufflenet v2: Practical guidelines for efficient cnn architecture design," in *ECCV*, pp. 116–131, 2018.

[7] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *Advances in neural information processing systems*, vol. 28, 2015.

[8] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *ICCV*, pp. 2961–2969, 2017.

[9] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, and R. Girshick, "Detectron2." https://github.com/facebookresearch/detectron2, 2019.

[10] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," in *NeurIPS Deep Learning and Representation Learning Workshop*, 2015.

[11] A. Romero, N. Ballas, S. E. Kahou, A. Chassang, C. Gatta, and Y. Bengio, "Fitnets: Hints for thin deep nets," in *ICLR*, 2015.

[12] S. Zagoruyko and N. Komodakis, "Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer," in *ICLR*, 2017.

[13] F. Tung and G. Mori, "Similarity-preserving knowledge distillation," in *ICCV*, pp. 1365–1374, 2019.

[14] B. Peng, X. Jin, J. Liu, D. Li, Y. Wu, Y. Liu, S. Zhou, and Z. Zhang, "Correlation congruence for knowledge distillation," in *ICCV*, pp. 5007–5016, 2019.

[15] S. Ahn, S. X. Hu, A. Damianou, N. D. Lawrence, and Z. Dai, "Variational information distillation for knowledge transfer," in *CVPR*, pp. 9163–9171, 2019.

[16] W. Park, D. Kim, Y. Lu, and M. Cho, "Relational knowledge distillation," in *CVPR*, pp. 3967–3976, 2019.

[17] N. Passalis and A. Tefas, "Learning deep representations with probabilistic knowledge transfer," in *ECCV*, pp. 268–284, 2018.

[18] B. Heo, M. Lee, S. Yun, and J. Y. Choi, "Knowledge transfer via distillation of activation boundaries formed by hidden neurons," in *AAAI*, pp. 3779–3787, 2019.

[19] J. Kim, S. Park, and N. Kwak, "Paraphrasing complex network: Network compression via factor transfer," in *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[20] J. Yim, D. Joo, J. Bae, and J. Kim, "A gift from knowledge distillation: Fast optimization, network minimization and transfer learning," in *CVPR*, pp. 4133–4141, 2017.

[21] Z. Huang and N. Wang, "Like what you like: Knowledge distill via neuron selectivity transfer," *arXiv preprint arXiv:1707.01219*, 2017.

[22] Y. Tian, D. Krishnan, and P. Isola, "Contrastive representation distillation," in *ICLR*, 2020.

[23] L. Chen, D. Wang, Z. Gan, J. Liu, R. Henao, and L. Carin, "Wasserstein contrastive representation distillation," in *CVPR*, pp. 16296–16305, 2021.

[24] P. Chen, S. Liu, H. Zhao, and J. Jia, "Distilling knowledge via knowledge review," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5008–5017, 2021.

[25] G. Xu, Z. Liu, X. Li, and C. C. Loy, "Knowledge distillation meets self-supervision," in *European Conference on Computer Vision (ECCV)*, 2020.