Supplementary Materials for *Automated Knowledge Distillation via Monte Carlo Tree Search*

Lujun Li[1†*] Peijie Dong[2†] Zimian Wei[2] Ya Yang[3]
[1] The Hong Kong University of Science and Technology, [2] National University of Defense Technology
[3] City University of Hong Kong
[1]lilujunai@gmail.com[1]{dongpeijie, weizimian16}@nudt.edu.cn, [3] yya9@outlook.com

0.1. Detailed Analysis of Distiller Search Space

In this part, we present a detailed formulation and discussion of distillation operations in our search space.

0.1.1 Knowledge Transformations

Attention & Mask Distillation: Instead of matching original features, attention distillation focuses on transferring knowledge from attention maps. For example, methods like AT [30] summarize values across the channel dimension to transfer attention knowledge. FKD [31] uses the sum of teacher and student attention to guide the student's focus on changeable areas. FGD enforces the student to learn crucial parts from the teacher and compensates for missing global information. Mask distillation applies a mask transformation operation before feature distillation. MGD [29] uses a random mask to cover the student's features based on a threshold λ. This mask helps guide the student's learning process.

Multi-scale & Local Distillation: Multi-scale distillation [2] leverages the benefits of modeling context information at different abstract levels. It involves extracting different levels of knowledge from the features using techniques such as spatial pyramid pooling. This approach allows the student network to utilize information from multiple scales. Local distillation focuses on distinctive and repeatable patterns or structures within an image, known as local features. LKD [16] selects local parts and uses a local correlation matrix to guide the student's learning. The original feature is divided into patches, and each patch is distilled separately.

Sample-wise Distillation: Sample-wise distillation considers the relationships between input samples during the knowledge transfer process. Methods like RKD (Relational Knowledge Distillation) compare angle and structure distances between samples. CC (Correlation Congruence) captures correlations between embedding representations. The correlation matrix is used to measure sample-wise features' similarity or to compute the Kullback-Leibler (KL) divergence for sample relationships. Following SP [27], we use the correlation matrix for the sample-wise features as follows:

$$\mathcal{L}_{KD} = \left\| \frac{(\tilde{A}^T)\cdot(\tilde{A}^T)^{\top}}{\left\|(\tilde{A}^T)\cdot(\tilde{A}^{T\top})\right\|_2} - \frac{(\tilde{A}^S)\cdot(\tilde{A}^S)^{\top}}{\left\|(\tilde{A}^S)\cdot(\tilde{A}^{S\top})\right\|_2} \right\|_2, \tag{1}$$

where $\tilde{A}^T, \tilde{A}^S \in \mathbf{R}^{N\times CHW}$ is the reshaping of the original features A^T and A^S. Furthermore, we also minimize the Kullback–Leibler (KL) divergence for the sample relationships as follows:

$$\mathcal{L}_{KD} = \frac{\mathcal{T}^2}{N}\sum_{n=1}^{N}\sum_{i=1}^{C\cdot W\cdot H}\phi\left(\tilde{A}^T_{n,i}\right)\cdot\log\left[\frac{\phi\left(\tilde{A}^T_{n,i}\right)}{\phi\left(\tilde{A}^S_{n,i}\right)}\right] \tag{2}$$

where ϕ is softmax function and $\mathcal{T}$ is temperature coefficient. In addition, sample distillation combined with L_2 distance when the distilling feature normalized in the sample dimension.

Channel-wise Distillation: Channel-wise distillation focuses on the knowledge contained in each channel of the feature maps. Methods like CWD minimize the KL divergence between probability maps calculated by normalizing the feature maps. ICKD (Inter-Channel Knowledge Distillation) calculates the disparity of the channel correlation matrix. Channel-wise features are transformed and compared using various operations. Following ICKD [18], channel-wise features G^S and G^T are transformed by the channel-wise operations and then computed as $G - L_2$ loss as follows:

$$\mathcal{L}_{KD} = \left\| \frac{(G^T)\cdot(G^T)^{\top}}{\|(G^T)\cdot(G^T)^{\top}\|_2} - \frac{(G^S)\cdot(G^S)^{\top}}{\|(G^S)\cdot(G^S)^{\top}\|_2} \right\|_2, \tag{3}$$

In addition, this channel-wise knowledge also can be derived from the as follows:

$$\mathcal{L}_{KD} = \frac{\mathcal{T}^2}{C}\sum_{c=1}^{C}\sum_{i=1}^{W\cdot H}\phi\left(G^T_{c,i}\right)\cdot\log\left[\frac{\phi\left(G^T_{c,i}\right)}{\phi\left(G^S_{c,i}\right)}\right] \tag{4}$$

*Corresponding author, † equal contribution.

where ϕ is softmax function and $\mathcal{T}$ is temperature coefficient. Note that channel distillation and L_2 distance are combined when normalization is achieved in the channel dimension.

Logits Distillation: Logits distillation targets the output layer of the network. Methods like logits KD [9] aim to match the logits or output probabilities between the teacher and student networks. This is achieved by calculating KL or Pearson distances for intra-class and inter-class knowledge with different temperature coefficients.

These distillation methods provide a range of techniques to transfer knowledge from the teacher network to the student network, considering different aspects such as attention, masks, multi-scale information, local features, sample relationships, channel-wise knowledge, and output logits.

0.1.2 Distance Functions and Hyperparameters.

In distillation, different distance functions are used to measure the difference between teacher and student output. Let P_i denote the predicted probability of class i by the teacher network and Q_i denote the predicted probability of class i by the student network.

L_2 **distance.** The L_2 distance measures the square root of the sum of the squared differences between the probabilities of each class in the two distributions. The L_2 distance between P and Q is defined as:

$$D_{L_2}(P,Q) = \sqrt{\sum_{i=1}^{n}(P_i - Q_i)^2}$$

Cosine distance. The cosine distance measures the cosine of the angle between the two probability vectors. This distance measure is useful when the magnitudes of the probability vectors are not important, only their directions. The cosine distance between P and Q is defined as:

$$D_{Cosine}(P,Q) = 1 - \frac{\sum_{i=1}^{n} P_i Q_i}{\sqrt{\sum_{i=1}^{n} P_i^2}\sqrt{\sum_{i=1}^{n} Q_i^2}}$$

when normalization applied to both distributions, i.e. $\sum_{i=1}^{n} P_i = \sum_{i=1}^{n} Q_i = 1$, cosine distance equivalent to L_2 distance as follows:

$$D_{L_2,norm} = \sqrt{\frac{1}{n^2}\sum_{i=1}^{n}(P_i - Q_i)^2} \tag{5}$$

$$= \sqrt{\frac{1}{n^2}\sum_{i=1}^{n} P_i^2 - \frac{2}{n^2}\sum_{i=1}^{n} P_i Q_i + \frac{1}{n^2}\sum_{i=1}^{n} Q_i^2} \tag{6}$$

$$= \sqrt{\frac{2}{n^2}\sum_{i=1}^{n}(P_i^2 + Q_i^2 - P_i Q_i)} \tag{7}$$

$$= \sqrt{2 \times \big(1 - D_{Cosine}(P,Q)\big)} \tag{8}$$

Pearson distance. The Pearson distance measures the correlation between the two probability vectors. The Pearson distance between P and Q is defined as:

$$D_{Pearson}(P,Q) = 1 - \frac{\sum_{i=1}^{n}(P_i - \bar{P})(Q_i - \bar{Q})}{\sqrt{\sum_{i=1}^{n}(P_i - \bar{P})^2}\sqrt{\sum_{i=1}^{n}(Q_i - \bar{Q})^2}}$$

where $\bar{P}$ and $\bar{Q}$ are the means of the two distributions.Similarly, Pearson distance also is correlated with the normalized L_2 distance.

KL distance. The KL distance measures the information lost when approximating the probability distribution P with the probability distribution Q, as follows:

$$D_{KL}(P,Q) = \sum_{i=1}^{n} P_i \log \frac{P_i}{Q_i} = \sum_{i=1}^{n} P_i \log P_i - \sum_{i=1}^{n} P_i \log Q_i$$

Temperature coefficient. In our distillation process, we utilize the temperature coefficient ($\mathcal{T}$) to scale the divergence between the probability distributions of the teacher and student networks. The value of
$mathcalT$ determines the sharpness or diffusion of the teacher's distribution and influences the behavior of the student network. A smaller
$mathcalT$ value leads to a sharper teacher distribution, encouraging the student network to produce probability estimates closer to the teacher's distribution. Conversely, a larger
$mathcalT$ value allows for a more diffuse teacher distribution, enabling the student network to generate more varied and less precise probability estimates. In our search space, we consider
$mathcalT$ values of 1, 4, 8, and 16 as options.

Loss weights. Additionally, the loss weights play a crucial role in balancing different optimization objectives and shaping the behavior of the student model during distillation. By adjusting these weights, we can tailor the student model's behavior to suit the specific requirements of the task. In our search space, we consider feature loss weights of 1, 5, 25, and 50 as options, as well as logits KD loss weights of 0.1, 0.5, 1, and 5. These options provide flexibility in determining the importance given to different components of the distillation loss during the training process.

1. Detailed Experimental Settings

In this section, we provide a comprehensive overview of the experiment settings employed for the CIFAR-100 and ImageNet datasets. It is worth noting that all experiments adhere to standard training settings, without incorporating additional data augmentation [13] or other specialized training techniques [21, 4, 5, 24].

1.1. Experiments on CIFAR-100

Dataset. Indeed, CIFAR-100 [10] is widely recognized as one of the most popular datasets for evaluating the performance of distillation methods in the field of classification. It consists of a total of 60,000 images, with 50,000 images designated for training purposes and the remaining 10,000 images reserved for testing. The dataset encompasses 100 distinct classes, providing a diverse range of objects and scenes for classification tasks. Researchers often employ CIFAR-100 to assess the effectiveness of various distillation techniques in improving the performance of models on challenging classification tasks.

Implementation. In the comparison experiments involving other knowledge distillation (KD) methods, we replicate the training

settings of KDs [25, 12, 11, 15, 14, 19, 3] for implementing various KD methods. The training is conducted for 240 epochs using a mini-batch size of 64 and the SGD optimizer with a weight decay of 5×10^{-4}. The learning rate is initialized to 0.05 and decayed by a factor of 0.1 at 150, 180, and 210 epochs using a multi-step learning rate schedule. To ensure a fair comparison with existing KD methods, we adhere to the original settings and configurations of CRD while implementing different knowledge distillation techniques. This includes factors such as the configuration of the weight balance (λ). Additionally, we adopt the same standard training settings as employed in our CIFAR-100 experiments to maintain consistency across evaluations.

1.2. Experiments on ImageNet

Dataset. We also conduct experiments on the ImageNet dataset (ILSVRC12) [23], considered the most challenging classification task. It contains about 1.2 million training images and 50 thousand validation images, each belonging to one of 1,000 categories.
Implementation. In the ImageNet experiments, the student models (*i.e*., ResNet-18 [7] and MobileNet [8]) are trained with 100 epochs. The batch size is 256, and the multi-step learning rate is initialized to 0.1, decayed by 0.1 at 30, 60, and 90 epochs. Other KD methods are implemented following the hyperparameter settings in the original paper. And Auto-KD's detailed settings are the same as those on the CIFAR-100.

1.3. Experiments on Vision Transformer

Vision transformer. The Transformer model [28] has gained significant traction in the field of natural language processing (NLP). Building upon its success in NLP, Google introduced the Vision Transformer (ViT) [6] and DeiT [26] to enhance the training process through data augmentation and knowledge distillation. However, training the Vision Transformer (ViT) from scratch presents a challenge due to the absence of inherent visual properties such as convolution. In recent times, knowledge distillation (KD) has emerged as an effective technique for training ViTs with convolutional neural networks (CNNs) serving as teachers. To evaluate the effectiveness of Auto-KD, we conduct a search for ViT-based distillation strategies on the CIFAR-100 dataset.
Student architectures. In the vision transformer architecture, the initial step involves dividing the input images into a sequence of patches. These patches are then processed by the transformer network to extract relevant image features for visual recognition. Initially, the patches are flattened and transformed into patch embeddings using a linear layer. Subsequently, learnable position embeddings are added to these patch embeddings to preserve positional information. To complete the input representation, a class token is concatenated with the enhanced patch embeddings. The internal structure of the vision transformer comprises position encoding, multi-head self-attention (MSA) blocks, and a feedforward network. Layernorm and residual connections are incorporated to enhance the network's performance. Furthermore, in the DeiT architecture, a distillation token is introduced to enable learning from the teacher's hard labels. In our work, we extend Auto-KD to employ DeiT-Tiny as the student model, while utilizing the same convolution teacher RegNetY-16GF
citeRegNet. Specifically, DeiT-Tiny consists of a hidden dimension of 192 and 12 layers, each with three attention heads.
Implementation. To ensure a fair comparison, we adopt the same data augmentation and regularization techniques outlined in DeiT (*e.g*., Auto-Augment, Rand-Augment, mixup). The weights of our transformers are initialized randomly by sampling from a truncated normal distribution. We conduct a distiller search using the identical settings employed in the CNN experiment. Afterward, we train the Vision Transformer (ViT) using the optimal distiller obtained, with ResNet-56 serving as the CNN teacher. The training process involves images of 224×224 resolution for 300 epochs, employing an initial learning rate of 5e-4 and a weight decay of 0.05, while utilizing the AdamW optimizer. A batch size of 128 is utilized, and the learning rate schedule follows the cosine policy. Similar to DeiT, Auto-KD incorporates a distillation token with a distilling head serving as the proxy model.

1.4. Experiments on Object Detection.

Dataset.We assess the performance of Auto-KD using the MS-COCO dataset [17]. This dataset comprises an extensive collection of over 120,000 images, spanning 80 distinct categories. To evaluate the effectiveness of Auto-KD, we conduct performance evaluations on the MS-COCO validation set.
Implementation.We employ the Auto-KD approach to enhance two popular object detection frameworks: the two-stage detector, exemplified by Faster R-CNN [22], and the one-stage detector, represented by RetinaNet
citefocal. These frameworks are widely utilized in the field of object detection. We initialize the backbone of these models with weights pre-trained on the ImageNet dataset [23]. Following common practice
citefocal, all models are trained using a $2\times$ learning schedule, spanning 24 epochs. Data augmentation includes horizontal image flipping.

2. Experiments on Semantic Segmentation

Datasets. Cityscapes is an arduous benchmark dataset that has been compiled from 50 cities, primarily focused on comprehending the urban environment. It comprises a collection of 5,000 meticulously annotated images, encompassing 19 distinct classes. The dataset is divided into three sets: training, validation, and testing, consisting of 2,975, 500, and 1,525 images, respectively. Additionally, Cityscapes includes an additional set of 20,000 coarsely labeled images, which were utilized in the knowledge distillation experiments.
Implementation details. In all experiments conducted in this section, we employ the mean Intersection-over-Union (mIoU) as the evaluation metric. The results are reported in the single-scale evaluation setting. Following prior works [20], we initially adopt DeeplabV3 [1] with a ResNet101 backbone [7] as the teacher model. For the other distillation methods, we explore DeepLabV3 models with the ResNet18 backbone. In the case of Auto-KD, we select DeepLabV3 with students obtained through the Auto-KD search on the ImageNet dataset. During the distillation process, we utilize the SGD optimizer with a poly-learning-rate policy. Each training image is randomly cropped into 512×512 pixels. The batch size is set to 8, and unless specified otherwise, the models are trained for 40K iterations.

References

[1] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *ECCV*, 2018.

[2] Pengguang Chen, Shu Liu, Hengshuang Zhao, and Jiaya Jia. Distilling knowledge via knowledge review. In *CVPR*, 2021.

[3] Peijie Dong, Lujun Li, and Zimian Wei. Diswot: Student architecture search for distillation without training. In *CVPR*, 2023.

[4] Peijie Dong, Xin Niu, Lujun Li, Linzhen Xie, Wenbin Zou, Tian Ye, Zimian Wei, and Hengyue Pan. Prior-guided one-shot neural architecture search. *arXiv preprint arXiv:2206.13329*, 2022.

[5] Peijie Dong, Xin Niu, Zhiliang Tian, Lujun Li, Xiaodong Wang, Zimian Wei, Hengyue Pan, and Dongsheng Li. Progressive meta-pooling learning for lightweight image classification model. In *ICASSP*, 2023.

[6] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[8] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint, arXiv:1704.04861*, 2017.

[9] Tao Huang, Shan You, Fei Wang, Chen Qian, and Chang Xu. Knowledge distillation from a stronger teacher. *arXiv preprint arXiv:2205.10536*, 2022.

[10] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. *Tech Report*, 2009.

[11] Lujun Li. Self-regulated feature learning via teacher-free feature distillation. In *ECCV*, 2022.

[12] Lujun Li and Zhe Jin. Shadow knowledge distillation: Bridging offline and online knowledge transfer. In *NeuIPS*, 2022.

[13] Lujun Li and Anggeng Li. A2-aug: Adaptive automated data augmentation. In *CVPRW*, 2023.

[14] Lujun Li, Liang Shiuan-Ni, Ya Yang, and Zhe Jin. Boosting online feature transfer via separable feature fusion. In *IJCNN*, 2022.

[15] Lujun Li, Liang Shiuan-Ni, Ya Yang, and Zhe Jin. Teacher-free distillation via regularizing intermediate representation. In *IJCNN*, 2022.

[16] Xiaojie Li, Jianlong Wu, Hongyu Fang, Yue Liao, Fei Wang, and Chen Qian. Local correlation consistency for knowledge distillation. In *ECCV*, 2020.

[17] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollar, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014.

[18] Li Liu, Qingle Huang, Sihao Lin, Hongwei Xie, Bing Wang, Xiaojun Chang, and Xiaodan Liang. Exploring inter-channel correlation for diversity-preserved knowledge distillation. In *ICCV*, 2021.

[19] Xiaolong Liu, Lujun Li, Chao Li, and Anbang Yao. Norm: Knowledge distillation via n-to-one representation matching. In *ICLR*, 2023.

[20] Yifan Liu, Changyong Shun, Jingdong Wang, and Chunhua Shen. Structured knowledge distillation for dense prediction. *arXiv preprint, arXiv:1903.04197*, 2019.

[21] Jie Qin, Jie Wu, Xuefeng Xiao, Lujun Li, and Xingang Wang. Activation modulation and recalibration scheme for weakly supervised semantic segmentation. In *AAAI*, 2022.

[22] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *arXiv preprint, arXiv:1506.01497*, 2015.

[23] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C Berg, and Fei-Fei Li. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 2015.

[24] Shitong Shao, Xu Dai, Shouyi Yin, Lujun Li, Huanran Chen, and Yang Hu. Catch-up distillation: You only need to train once for accelerating sampling. *arXiv preprint arXiv:2305.10769*, 2023.

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

[26] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Herve Jegou. Training data-efficient image transformers amp; distillation through attention. In *ICML*, 2021.

[27] Frederick Tung and Greg Mori. Similarity-preserving knowledge distillation. In *ICCV*, 2019.

[28] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017.

[29] Zhendong Yang, Zhe Li, Mingqi Shao, Dachuan Shi, Zehuan Yuan, and Chun Yuan. Masked generative distillation. *arXiv preprint arXiv:2205.01529*, 2022.

[30] Sergey Zagoruyko and Nikos Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. *arXiv preprint arXiv:1612.03928*, 2016.

[31] Linfeng Zhang and Kaisheng Ma. Improve object detection with feature-based knowledge distillation: Towards accurate and efficient detectors. In *International Conference on Learning Representations*, 2020.