Appendix

A TRAINING DETAILS

A.1 DATASET DETAILS

We conduct experiments using the following real-world datasets to showcase the effectiveness of our approach,

CIFAR-100 (Krizhevsky, 2009). The dataset consists of a total of 60K examples, distributed across 100 distinct classes. Each example in this dataset comprises images with a resolution of $32 \times 32 \times 3$. Specifically, the training set encompasses 50,000 examples, while the remaining 10K serve as the testing set. In our experimental setup, approximately 5K examples are allocated for use as a validation set for our C-NET. For the other baseline models, these validation examples are utilized for the purpose of hyper-parameter tuning.

Tiny-ImageNet (Le & Yang, 2015). This dataset is derived from the extensive ImageNet-1K dataset (Russakovsky et al., 2015). This dataset encompasses a total of 100K images, which have been downsampled to a 64×64 resolution, representing a subset of 200 classes mirroring those in the ImageNet-1K dataset, with each class containing precisely 500 images. As part of our experimental protocol, we have set aside an independent validation set comprising 10K examples, which is utilized for both our proposed method and the baseline models.

For both the datasets, we use the data augmentations methods, using the torchvision's transforms module. We use RandomCrop, RandomResizedCrop, RandomSizedCrop, RandomHorizontalFlip, Normalize, ColorJitter for our purpose. Augmentation methods applied on both dataset, over all experiments, baselines, over all models.

A.2 MODEL DETAILS

In Section 4.1, we introduce several smaller ResNet models, specifically ResNet10-xxxs, ResNet10-xxs, ResNet10-xs, ResNet10-s, ResNet10-m, and ResNet10-l. These models are adopted from Kag et al. (2023), and in Table 3, we compare their architectural details with well-established ResNet models like ResNet10, ResNet18, and ResNet34. Similar to the standard ResNet models, these newer, more compact versions also employ the traditional 'BasicBlock' as their fundamental building block. The architectural structure consists of a convolutional block, four stages of residual blocks, an adaptive average pooling layer, a convolutional block, and a classifier layer. The sole variation among the various capacity variants within this experimental setup is the number of filters in each stage and the residual block. Additionally, Table 3 provides information about the number of parameters and multiply-addition (MAC) operations for each model for the datasets CIFAR-100 and Tiny-ImageNet.

Architecture	Filters	Basic block Repeats	CIFAR-100		Tiny-Imagenet	
			MACs	Params	MACs	Params
ResNet10-xxs	[8, 8, 16, 16]	[1, 1, 1, 1]	2 M	13 K	8 M	15 K
ResNet10-xs	[8, 16, 16, 32]	[1, 1, 1, 1]	3 M	28 K	12 M	31 K
ResNet10-s	[8, 16, 32, 64]	[1, 1, 1, 1]	4 M	84 K	16 M	90 K
ResNet10-m	[16, 32, 64, 128]	[1, 1, 1, 1]	16 M	320 K	64 M	333 K
ResNet10-l	[32, 64, 128, 256]	[1, 1, 1, 1]	64 M	1.25 M	255 M	1.28 M
ResNet10	[64, 128, 256, 512]	[1, 1, 1, 1]	253 M	4.92 M	1013 M	5 M
ResNet18	[64, 128, 256, 512]	[2, 2, 2, 2]	555 M	11.22 M	2221 M	11.27 M
ResNet34	[64, 128, 256, 512]	[3, 4, 6, 3]	1159 M	21.32 M	4637 M	21.38 M

Table 3: Comparision of newly introduced smaller ResNet with the standard ResNet models

A.3 HYPER-PARAMETERS

For all the KD baselines listed in Section 4.2 we use temperature $\tau = 2$. We employ the SGD optimizer to train the student models and ADAM optimizer (Kingma & Ba, 2014) to train C-NET. We use a batch size of 400 for both the CIFAR100 and TinyImagenet datasets, We train the student models for 500 epochs and update C-NET every 20 epoch (*i.e.*, L = 20). We use cosine annealing (Loshchilov & Hutter, 2017) as the learning rate schedule for training the student models. We warm start each student model by first training it using the cross entropy loss without using the teacher model for all KD baselines. For both the datasets, we use a learning rate of 0.05, set weight decay to $1e-4$ and momentum to 0.95. For the C-NET training we use a learning rate of $1e-3$ and set weight decay to $1e-4$.

B ADDITIONAL EXPERIMENTS

We have released anonymized code at the URL: `https://anonymous.4open.science/r/Multinet-E01D`.

B.1 EFFECT ADDING POOLED STUDENT

Teacher	Student	MC-DISTIL-NoPS	MC-DISTIL
ResNet10-l	ResNet10-xxs	36.29	**36.44**
	ResNet10-xs	47.14	**47.65**
	ResNet10-s	57.37	**57.55**
	ResNet10-m	68.47	**68.51**
ResNet10	ResNet10-xxs	36.57	**36.78**
	ResNet10-xs	47.55	**48.03**
	ResNet10-s	57.59	**58.2**
	ResNet10-m	69.03	**69.5**

Table 4: We present the analysis of the effect of adding loss component based on the pooled student to MC-DISTIL. Here MC-DISTIL-NoPS represents the test result obtained with model trained without pooled student based loss component.

We analyze the effect of adding an additional loss component associated with a fake teacher introduced in equation 4 in Section 3.3. The PooledStudent logit is composed of logits from fellow student models and was introduced to improve communication amongst the student models. We present test accuracy obtained on CIFAR100 dataset in Table 4 where MC-DISTIL-NoPS represents training without addition of loss component associated with a fake teacher. We note that training with this new logit is helpful to boost the performance gains obtained via using meta-collaborative learning.