Supplemental Material of
Distilling Cross-Task Knowledge via Relationship Matching

Han-Jia Ye
Nanjing University
yehj@lamda.nju.edu.cn

Su Lu
Nanjing University
lus@lamda.nju.edu.cn

De-Chuan Zhan
Nanjing University
zhandc@lamda.nju.edu.cn

Abstract

*In the supplemental material of the paper "Distilling Cross-Task Knowledge via Relationship Matching", we provide more discussions, analyses, experiments for our proposed **RE**lationship **F**ac**I**litated **L**ocal c**L**assifi**E**r **D**istillation (*REFILLED*) approach, which reuses the classification knowledge from a* cross-task *model to facilitate the training of the current task classifier. There are three parts in the supplementary: first, we provide concrete deviations and discussions to explain the two steps in* REFILLED*; then, we describe the concrete settings for all the experiments in detail; last are additional experimental results.*

1. Discussion on Two-Steps in REFILLED

There are two steps in REFILLED to distill the knowledge from a cross-task teacher w.r.t. the embedding and the (top-layer) classifier, respectively. In this section, we discuss the advantage of each step, and provide concrete derivations in the main paper.

1.1. Distill the Embedding

REFILLED distills the discriminative ability of the embedding by aligning the stochastic probability over triplets with the teacher [22]. Given a triplet $(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k)$, the anchor $\mathbf{x}_i$ is similar to its target neighbor $\mathbf{x}_j$ and is dissimilar to the impostor $\mathbf{x}_k$. Thus a high-quality embedding pulls similar instances together and pushes dissimilar ones far away. The similarity between instances are usually measured by their labels — we think two instances are similar if they come from the same class, and they are dissimilar if they have different labels.

Usually, we apply a loss function $\iota(\cdot)$ over the triplets to force the distance between embeddings to be matched with the relationship indicated by the triplet. Specifically, we can minimize the embedding ϕ over the sampled triplets:

$$\min_{\phi} \sum_{ijk} \iota\left(\mathrm{Dist}_{\phi}\left(\mathbf{x}_i, \mathbf{x}_k\right) - \mathrm{Dist}_{\phi}\left(\mathbf{x}_i, \mathbf{x}_j\right)\right) \ . \tag{1}$$

The summation over ijk means we sum the loss over the sampled triplets $\{\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k\}$. The loss function $\iota(x)$ over the embedding usually acts as an upper bound of the zero-one loss, where the larger the value of the input x, the smaller of the loss output. By minimizing Eq. 1, the embedding makes distance between dissimilar instances larger than the distance between similar ones. There are several options for the loss function, such as the hinge loss ($\iota(x) = \max(1-x, 0)$) [25, 19, 11] and logistic loss ($\iota(x) = \log(1+\exp(-x))$).

As in [22], we define the stochastic probability of a triplet $(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k)$ as a Bernoulli distribution $\mathcal{P}_{ijk}$, where the random variable means whether such a triplet is satisfied (with probability p_{ijk}) or not (with probability $1-p_{ijk}$) based on the current embedding ϕ.

$$p_{ijk}(\phi) = \frac{\exp(-\mathrm{Dist}_{\phi}(\mathbf{x}_i, \mathbf{x}_j)/\tau)}{\exp(-\mathrm{Dist}_{\phi}(\mathbf{x}_i, \mathbf{x}_j)/\tau) + \exp(-\mathrm{Dist}_{\phi}(\mathbf{x}_i, \mathbf{x}_k)/\tau)} \ . \tag{2}$$

In Eq. 3, we match the Bernoulli distribution of triplets based on both teacher's and student's embedding ϕ_T and ϕ_S with KL-divergence, so that the teacher's embedding supervises the optimizing of the student's embedding.

$$\min_{\phi_S} \sum_{ijk} \mathbf{KL}\left(\mathcal{P}_{ijk}(\phi_T) \parallel \mathcal{P}_{ijk}(\phi_S)\right) \ . \tag{3}$$

Sampling Semi-Hard Triplets. All triplets are composed by the current task instances. In other words, when optimizing the student model with stochastic gradient descent, we generate triplets for each sampled mini-batch. We make ℓ_2-normalization on all the embeddings before computing their distances, and only apply the temperature τ in $\mathcal{P}_{ijk}(\phi_T)$. The triplets are sampled following the semi-hard protocol [19]. In detail, we enumerate all instances in the mini-batch and treat each one as the triplet anchor. For each anchor, $\mathbf{x}_i$ we first find all target neighbors $\mathbf{x}_j$ (with the same label) in the given mini-batch. Then for each pair of $\mathbf{x}_i$ and $\mathbf{x}_j$, we set x_k as the nearest impostor with a different label to $\mathbf{x}_i$, which meanwhile locates father away from $\mathbf{x}_i$ than $\mathbf{x}_j$. The distances in the previous sampling process are measured based on the student's embedding. Therefore, if the student model finds some triplets are hard to evaluate, it will ask the teacher for help about the concrete measures of the similarity proportions.

Interpretation of the Matching. We can rethink the objective in Eq. 3 by the following reformulation:

$$\begin{aligned}
& \mathbf{KL}\left(\mathcal{P}_{ijk}(\phi_T) \parallel \mathcal{P}_{ijk}(\phi_S)\right) \qquad (4)\\
= \ & p_{ijk}(\phi_T)\ln\frac{p_{ijk}(\phi_T)}{p_{ijk}(\phi_S)} + (1-p_{ijk}(\phi_T))\ln\frac{1-p_{ijk}(\phi_T)}{1-p_{ijk}(\phi_S)}\\
= \ & \underbrace{p_{ijk}(\phi_T)\ln p_{ijk}(\phi_T)}_{\text{constant}} - p_{ijk}(\phi_T)\ln p_{ijk}(\phi_S)\\
& + \underbrace{(1-p_{ijk}(\phi_T))\ln(1-p_{ijk}(\phi_T))}_{\text{constant}}\\
& - (1-p_{ijk}(\phi_T))\ln(1-p_{ijk}(\phi_S))\\
\cong \ & -p_{ijk}(\phi_T)\ln p_{ijk}(\phi_S) \ - \ \ln(1-p_{ijk}(\phi_S))\\
& + p_{ijk}(\phi_T)\ln(1-p_{ijk}(\phi_S))\\
\cong \ & -p_{ijk}(\phi_T)(-\mathrm{Dist}_{\phi_S}(\mathbf{x}_i,\mathbf{x}_j) - \ln\Delta) + \ln\Delta\\
& + \mathrm{Dist}_{\phi_S}(\mathbf{x}_i,\mathbf{x}_k) + p_{ijk}(\phi_T)(-\mathrm{Dist}_{\phi_S}(\mathbf{x}_i,\mathbf{x}_k) - \ln\Delta)\\
& \Delta \triangleq \exp\left(-\mathrm{Dist}_{\phi_S}(\mathbf{x}_i,\mathbf{x}_j)\right) + \exp\left(-\mathrm{Dist}_{\phi_S}(\mathbf{x}_i,\mathbf{x}_k)\right)\\
\cong \ & p_{ijk}(\phi_T)\mathrm{Dist}_{\phi_S}(\mathbf{x}_i,\mathbf{x}_j) + (1-p_{ijk}(\phi_T))\mathrm{Dist}_{\phi_S}(\mathbf{x}_i,\mathbf{x}_k)\\
& + \ln\Delta\\
\cong \ & p_{ijk}(\phi_T)\mathrm{Dist}_{\phi_S}(\mathbf{x}_i,\mathbf{x}_j) + (1-p_{ijk}(\phi_T))\mathrm{Dist}_{\phi_S}(\mathbf{x}_i,\mathbf{x}_k)\\
& + \ln\left(\exp\left(-\mathrm{Dist}_{\phi_S}(\mathbf{x}_i,\mathbf{x}_j)\right) + \exp\left(-\mathrm{Dist}_{\phi_S}(\mathbf{x}_i,\mathbf{x}_k)\right)\right)\\
\cong \ & (p_{ijk}(\phi_T) - 1)\mathrm{Dist}_{\phi_S}(\mathbf{x}_i,\mathbf{x}_j)\\
& + (1-p_{ijk}(\phi_T))\mathrm{Dist}_{\phi_S}(\mathbf{x}_i,\mathbf{x}_k)\\
& + \ln\left(1 + \exp\left(-(\mathrm{Dist}_{\phi_S}(\mathbf{x}_i,\mathbf{x}_k) - \mathrm{Dist}_{\phi_S}(\mathbf{x}_i,\mathbf{x}_j))\right)\right)\\
\cong \ & \rho_{ijk}\left(\mathrm{Dist}_{\phi_S}(\mathbf{x}_i,\mathbf{x}_k) - \mathrm{Dist}_{\phi_S}(\mathbf{x}_i,\mathbf{x}_j)\right)\\
& + \iota\left(\mathrm{Dist}_{\phi_S}(\mathbf{x}_i,\mathbf{x}_k) - \mathrm{Dist}_{\phi_S}(\mathbf{x}_i,\mathbf{x}_j)\right)
\end{aligned}$$

The notation $\cong$ neglects the constant term in the equation. Define $\rho_{ijk} = 1 - p_{ijk}(\phi_T)$ and $\iota(x) = \ln(1+\exp(-x))$ as the logistic loss. From Eq. 4, we find by matching the stochastic triplet probability based on ϕ_T and ϕ_S, the objective first optimizes embedding with a logistic loss, where the similar instances indicated by the triplets have small distances while dissimilar ones should have large distances. Furthermore, Eq. 4 *rectify the minimizing/maximizing of distances by adding different weights* upon the distances between similar/dissimilar pairs based on the teacher's estimation. For example, if a pair $(\mathbf{x}_i, \mathbf{x}_j)$ is similar, then $p_{ijk}(\phi_T)$ is large and ρ_{ijk} is small, which applies more weights on their distances $\mathrm{Dist}_{\phi_S}(\mathbf{x}_i,\mathbf{x}_j)$ and the rectification of the logistic loss has minor influence. Otherwise, ρ_{ijk} is large and the force to minimize $\mathrm{Dist}_{\phi_S}(\mathbf{x}_i,\mathbf{x}_j)$ will not be strong since $\mathbf{x}_i$ and $\mathbf{x}_j$ are not too similar measured by the teacher's embedding. In other words, different from the binary label ("similar" or "dissimilar") indicated by the triplet, the relative similarities between pairs are specified by the teacher's embedding ϕ_T.

Differences with Related Methods. There are several recent methods propose to distill the knowledge from the embedding perspective [2, 1, 10, 14, 21], and the effectiveness of the distilled embedding is usually verified for the representation learning tasks or standard knowledge distillation tasks. For example, an implementation choice in [13] constructs angels over triplets first, and then matches the angels by regression; Qi et al [15] take advantage of the imprinted weights to initialize the classifier for low-shot learning. In our REFILLED approach, we emphasize to distill the classification ability from a *cross-task* teacher model, and in the embedding distillation stage, we take advantage of the stochastic triplet probability, which is more general as revealed in Eq. 4. The superiority of REFILLED is also validated in the experiments.

1.2. Distill the Local Classifier

In the standard knowledge distillation [5], we optimize the student model together with matching its prediction confidences with a (fixed) teacher:

$$\min_{f_S} \sum_{i=1}^{N} \ell(f_S(\mathbf{x}_i), \mathbf{y}_i) + \lambda\mathcal{R}(\mathbf{s}_\tau(f_T(\mathbf{x}_i)), \mathbf{s}_\tau(f_S(\mathbf{x}_i))) \ . \tag{5}$$

Denote the vanilla objective with the cross-entropy loss (i.e., the first part in Eq. 5) as $O_{\mathbf{xent}}$ and the whole objective in Eq. 5 as $O_{\mathbf{kd}}$.

Facilitated by the improved embedding, we propose to utilize a local knowledge distillation term to assist distill the classification ability from the teacher:

$$\min_{f_S} \sum_{i=1}^{N} \ell(f_S(\mathbf{x}_i), \mathbf{y}_i) + \lambda\mathbf{KL}\left(p_{\phi_T}(\mathbf{y}_i \mid \mathbf{x}_i), \mathbf{s}_\tau(f_S(\mathbf{x}_i))\right) \ . \tag{6}$$

It is notable that the $p_{\phi_T}(\mathbf{y}_i \mid \mathbf{x}_i)$ is the Nearest Center Mean (NCM) confidence of the teacher over the C' classes in the current mini-batch.[1] $f_S(\mathbf{x}_i) = W^\top\phi_S(\mathbf{x}_i)$ is the prediction of the student model, while $\mathbf{s}_\tau(f_S(\mathbf{x}_i))$ is the corresponding probability normalized with the softmax. Eq. 6 is a *local* version to distill the knowledge since it only considers the classes locally in the current sampled mini-batch. We denote the objective with the Local Knowledge Distillation (LKD) term as $O_{\mathbf{LKD}}$.

We analyze the effectiveness of the knowledge distillation by its gradient over the top-layer classifier $W \in \mathbb{R}^{d\times C}$, and we omit the bias for simplicity. The c-th column of W

[1]Recall there are C classes in total in the data set.

corresponds to the classifier $\mathbf{w}_c$. In the following, without loss of generality, we analyze the gradient of $\mathbf{w}_c$ over *one single* instance $\mathbf{x}$, whose target label is c. Denote p_c and q_c as the c-th element in the student's and teacher's normalized prediction $\mathbf{s}_\tau(f_S(\mathbf{x}))$ and $\mathbf{s}_\tau(f_T(\mathbf{x}))$ (the posterior probability of the c-th class given the student's and the teacher's embedding of the instance), respectively.

In vanilla learning scenario, we have the gradient w.r.t. $\mathbf{w}_c$ as

$$\frac{\partial O_{\mathbf{xent}}}{\partial \mathbf{w}_c} = \left[-p_c(1-p_c)\right]\phi(\mathbf{x})\ . \tag{7}$$

With an additional knowledge distillation term (refer to Eq. 5), the gradient w.r.t. $\mathbf{w}_c$ changes to

$$\frac{\partial O_{\mathbf{kd}}}{\partial \mathbf{w}_c} = \left[-p_c + \sum_{c'=1}^{C} p_{c'}q_c\right]\phi(\mathbf{x})\ . \tag{8}$$

Comparing Eq. 7 and Eq. 8, when considering the soft supervision from the teacher, not only the instance from the target class, but also those from related classes (the relatedness is specified by the student's prediction $p_{c'}$, weighted by the teacher's prediction q_c) will be incorporated to direct the update of the classifier.

The summation in Eq. 8 is computed over all C classes in the data set, if the number of C is large, the normalized class posterior probability q_c is small, so that the helpful related class instance will not be weighted obviously. Therefore, the supervision made by the teacher will be weakened a lot.

We verify this claim by an experiment on CIFAR-100. The averaged norm differences between the vanilla cross-entropy loss and the knowledge distillation variants over the gradient of all top-layer classifiers serve as a measure. The smaller the norm difference, the weaker the additional supervision signal provided by the teacher. Fig. 1 plots the change of the norm difference when we increase the number of randomly sampled classes in a task. All the gradients are measured during the initial optimization of the model. As demonstrated by the figure, when the number of classes grows larger, the norm of gradient difference between vanilla KD loss and cross-entropy loss decreases faster than that between local KD loss and cross-entropy loss, which means the supervision made by the vanilla KD teacher is weakened more than the supervision made by the local KD teacher.

Therefore, we consider a *local* version of the knowledge distillation term in Eq. 6, where only *the classes in the current mini-batch* are considered, i.e., the posterior probabilities are only normalized over the current set of classes.

2. Experimental Settings

We will provide detailed settings for all tasks, namely the cross-task knowledge distillation, the standard knowledge distillation, and the middle-shot classification task.

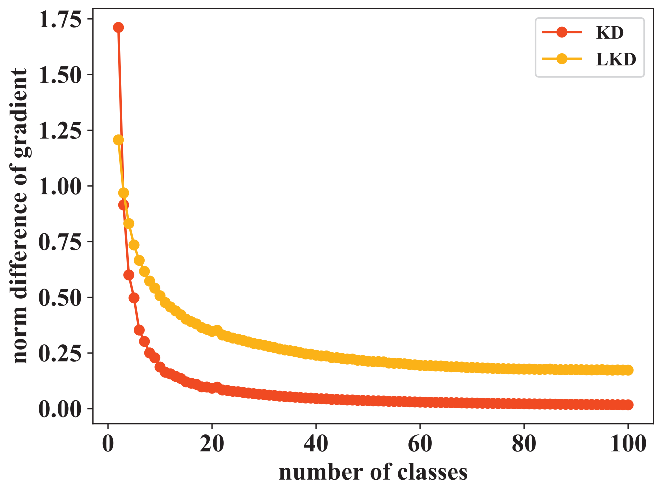

Figure 1. The averaged norm differences between the vanilla cross-entropy loss and the knowledge distillation variants (KD and LKD) over the gradient of all top-layer classifiers. When the number of classes in a task grows, the norm difference decreases fast. However, the decrease is mitigated when using local KD loss.

2.1. Cross-Task Knowledge Distillation

In the cross-task knowledge distillation task, we 'd like to verify whether a well-trained cross-task teacher model could help the training of the student model in current task.

Datasets. We investigate the Caltech-UCSD Birds-200-2011 (CUB) [24] data set, which is a fine-grained classification problem over 200 different species of birds. We do not use the attribute information of all the instances. As a pre-processing, we crop all images based on the provide bounding boxes.

Splits. We implement a cross-task knowledge transfer task by selecting two sets for teacher and student respectively with *non-overlapping* classes. Therefore, 100 of the 200 classes are selected to train a teacher model, while the remaining 100 classes are used to train the student. The classes in the CUB data set are sorted in alphabetic order, and we consider two different kinds of split criteria.

In a "hard" case, the two sets of 100 classes are split based on the given alphabetic order. While in the "easy" case, we randomly select 100 classes from the class pool to train the teacher, and the remaining classes are used for the student. The main difference between the two cases is the domain gap between the training set of the teacher and the student. Since the original classes are sorted in alphabetic order, the classes with numerically close indexes are more similar. Thus, the teacher is more "distant" from the student in the hard case than in the easy one.

For each 100-way classification task for the teacher or the student, we split 70% data in each class for training, and the remaining instances are used for test.

Implementation of the Teacher. By optimizing a cross-entropy loss, we train a teacher model based on the corresponding training set with a MobileNets [6] model, whose width multiplier is 1.0. We use the Stochastic Gradient Descent (SGD) as the default optimizer, where the momentum is 0.9, batch-size is 128, maximum epoch number is 200, the initial learning rate is 0.1, and we time the learning rate by 0.2 after 50 epochs. We hold out a part of examples from the training set for validation, from which the best set of hyper-parameters are selected. With the best selected hyper-parameters, we re-train the teacher model on the whole training set. When training the model, we first resize the image to 224×224, and use the random crop together with the horizontal flip as the data augmentation. This is the same when training the student.

Implementation of the Student. We use different configurations of the MobileNets [6] and adjust the model complexity with different width multipliers (complicated models have larger multipliers) in $\{1, 0.75, 0.5, 0.25\}$. There are two stages for the student. In both stages, the temperature τ of the teacher's model is set to 2, and we do not smooth the logits of the student. When distilling the embedding, we set the momentum 0.9, the batch-size 128, the maximum epoch 200, the initial learning rate 0.1, and we time the learning rate by 0.2 after 50 epochs. While in the second stage, we use the same hyper-parameter values. λ is set to 0.01, and the same λ is used across different experiments on a data set. We find the performance of ReFilled is not very sensitive to λ. Since the weight parameter λ in Eq. 6 is decayed during the optimization, we use $\lambda = 1000 * \exp(-0.05)\lambda$ to update the weight of the local knowledge distillation term after each epoch.

Evaluations. The averaged classification accuracy over 3 trials (with different random seeds) is reported.

Comparison Baselines. We consider three kinds of baseline methods in this task.

- Classification based on the teacher's embedding. We can extract features with the pre-trained teacher's embedding function ϕ_T for those instances in the student's split. Based on the teacher's embedding, we can make classification by either the nearest neighbor (1NN) classifier or the linear logistic regression (LR). Besides, we fine-tune the teacher's model with the instances in the student's split with small learning rate (0.0001) and fixed number of epochs (50). Fine-tuning the teacher model requires the student has the same architecture with the teacher. Since in this case, the student has the same number of training instances as the teacher, using a large learning rate will make the student obtain the same weights as training from scratch, so we use a small initial learning rate in our experiments.
- Knowledge distillation baselines. In the cross-task knowledge distillation task, we compare our method with one representative embedding-based distillation approach, the Relational Knowledge Distillation (RKD), and fine-tune the whole student model with its distilled embedding. The hyper-parameters for RKD are tuned in the same way as our ReFilled approach.
- Variants of ReFilled. We investigate the importance of different components in ReFilled.

2.2. Standard Knowledge Distillation

In the standard knowledge distillation, we reuse the knowledge from a same-task teacher model, i.e., both the teacher and the student target the same classification task.

Datasets. Following [1], we test the knowledge distillation ability of our ReFilled on the benchmark data set CIFAR-100 [9]. CIFAR-100 is a small-image data set (with size 32×32), which contains 100 classes and 6000 images in each class. In addition, we also consider the CUB data set, which has a different split strategy compared with the one in the previous task.

Splits. In CIFAR-100, we follow the standard split, where there are 5,000 images in each class for training and 1,000 images for test. While on CUB, all 200 classes are used during training. The standard training and test partitions are used here. It is notable that both the teacher and the student models are investigated on the same training and test sets.

Implementation of the Teacher and the Student Three different families of the neural networks are taken into account to test the ability of ReFilled, namely the ResNet [4] , Wide-ResNet [27], and MobileNets [6]. Towards getting different capacities of the model, we change the depth of the ResNet (through the number of layers), the (depth, width) pair of the Wide-ResNet, and the width of the MobileNets (through the width multipliers). We set $\lambda = 0.02$ on CIFAR-100.

We mainly consider two sub-tasks in the standard knowledge distillation, i.e., setting the teacher and the student model from either the same or different model families.

- Same-family knowledge distillation. Both the teacher and the student come from the same model family. We use the same configuration as [1]. In CIFAR-100, both the teacher and the student are Wide-ResNets. We set the (depth, width) pair of the teacher as $(40, 2)$, and change such configuration parameters of the student model among $(40, 2)$, $(16, 2)$, $(40, 1)$, and $(16, 1)$. On CUB, we consider the MobileNets, by setting the teacher's width multiplier to 1, we vary the width multipliers of the student among $\{1, 0.75, 0.5, 0.25\}$.

- Different-family knowledge distillation. The model of the teacher and the student come from different architecture families. We consider the knowledge transfer flow from ResNet to MobileNets. Taking the computational burden into consideration, when in CIFAR-100, we choose the teacher as the ResNet-110, and we use ResNet-34 as the teacher in CUB. We only change the width multipliers of the student model in $\{0.75, 0.5, 0.25\}$ on CUB to keep the student model has smaller capacity when compared with the teacher.

Similar ways with the previous section are used to train both the teacher and the student model in the standard knowledge distillation task. Here we set the temperature of the teacher's model to 4. For CIFAR-100, we pad 4 for each edge before we do the random crop.

Evaluations. Both teacher and student are trained on the same set with three different seeds of initialization, and we report the mean accuracy of the student on the test set.

2.3. Middle-Shot Classification Task

Similar to the popular few-shot learning task (FSL) [23, 17, 3, 16, 12, 26, 15], we also investigate the problem learning with middle-shot of examples.

Datasets. We use the popular *Mini*ImageNet data set [23], which contains 100 classes in total and 600 images in each class. All images are resized to 84×84 before inputting into the models.

Splits. Following [23, 17], there are 64 classes (SEEN class) to train the teacher (a.k.a. the meta-train set), 16 classes for validation (a.k.a. the meta-val set), and we sample tasks from the remaining 20 classes (a.k.a. the meta-test set) to train the student.

Implementation of the Teacher and the Student. We set the student as a 4-layer ConvNet [23, 20, 3], and consider two types of the teacher model, i.e., the same 4-layer ConvNet (but trained on different classes in the meta-train set) and the ResNet [12, 26]. The ConvNet contains 4 identical blocks, and each block is a sequential of convolution operator, batch normalization [8], ReLU, and Max pooling. We add another global max pooling layer to reduce the computational burden after the 4 blocks, which gives rise to a 64-dimensional embedding before the top-layer classifier. For ResNet, we follow the architecture in [12, 26], which removes the two down-sampling layers in the vanilla ResNet. The ResNet outputs 640-dimension embeddings.

We train a teacher model on the SEEN classes set (meta-train set) with ResNet and ConvNet. Supervised by the cross-entropy loss, we use random crop and horizontal flip as the data augmentation, SGD w/ momentum 0.9 as the optimizer, 128 as the batch size. Then the student is trained with the help of the teacher and the tasks are sampled from the UNSEEN classes (meta-test set).

Evaluations. Define a K-shot C-way task as a C-class classification problem, and there are K instances in each class. Different from the few-shot learning where $C = 5$ and $K \in \{1, 5\}$, here we consider there are a bit more instances in each class, i.e., $K = \{10, 30\}$. Although the value of K increases in the middle-shot learning, it is still small to train a complicated neural network from scratch. We sample tasks from the 20-class split (meta-test set) to train the student model and evaluate the results by classifying another 15 instances from each of the C classes. We evaluate the performance by mean accuracy over 600 trials.

Comparison Methods. We compare our methods with two branches of baselines:

- Meta-learning methods. Meta-learning is a popular way to solve the few-shot classification problem. To mimic the test case, it samples C-Way K-Shot tasks from the SEEN class set to learn task-level inductive bias like embedding [23, 20]. However, the computational burden (e.g., the batch size) becomes large when the number of shots increases. Besides, meta-learning needs to specify the way to obtain a meta-model from the SEEN classes. We compare our methods with the embedding-based meta-learning approaches like and ProtoNet [20] and FEAT [26].
- Embedding-based baselines. We can make predictions directly with the teacher's embedding, the penultimate layer of the teacher model, by leveraging the nearest neighbor. Based on the embedding, we also train linear classifiers like SVM on the current task's middle-shot data or fine-tune the whole model. It is notable that we tune the hyper-parameter of such methods with sampled middle-shot tasks on the validation split.

3. Additional Experimental Results

This section shows the additional experimental results on the knowledge distillation and middle-shot learning tasks.

Table 1. The cross-task distillation mean accuracy on CUB data set, where teacher and student are trained for non-overlapping 100 classes with MobileNets. Two split scenarios are considered, i.e., the "Easy" and "Hard" cases. The three values in the teacher's correspond to baselines: applying 1NN based on teacher's embedding, train a linear LR classifier based on fixed teacher's embedding, and Fine-Tune (FT) based on teacher's embedding. More details can be found in the text.

	Easy				Hard			
Channel	1	0.75	0.5	0.25	1	0.75	0.5	0.25
Teacher	1NN: 49.23, LR: 56.77, FT: 66.94				1NN: 45.31, LR: 53.82, FT: 65.72			
Student	70.04	68.13	66.44	64.63	71.25	67.56	66.85	64.48
RKD [13]	71.10	68.81	67.15	64.28	70.83	68.8	67.44	63.97
Vanilla	71.62	70.27	70.15	66.75	71.90	69.14	68.91	65.38
LKD	71.93	70.73	70.88	67.41	72.53	70.01	69.50	66.42
REFILLED	72.48	71.04	71.35	67.87	73.38	70.42	69.77	67.10

3.1. Cross-Task Knowledge Distillation

The results of cross-task distillation could be found in Table 1. Two splits of the data are considered, where the domain gaps between the teacher and student are different.

We first adapt the teacher model for the cross-task classification via three baselines, i.e., the 1NN based on teacher's embedding, train a linear Logistic Regression (LR) classifier based on the fixed teacher's embedding, and Fine-Tune (FT) the model based on teacher's embedding. The teacher has the width multiplier 1, so it achieves the test accuracy around 70.04 when directly training on the student's split (equal the student's performance with width multiplier 1). The test accuracy of the student becomes higher when learning with more complicated models (larger width multiplier values). It can be found that the baselines of the teacher model perform higher when trained in the easy scenario (the left part in Table 1).

We also compare with one representative embedding-based approach Relation Knowledge Distillation (RKD) [13], and fine-tune the model after obtaining the distilled embedding from the cross-task teacher. RKD sometimes gets better accuracy than the vanilla student model (denoted as "student" in the table).

Since REFILLED distills both the embedding and the classification ability from the teacher, we perform two baselines upon the improved embedding of REFILLED. We test the quality of the embedding by fine-tune the model with cross-entropy based on the REFILLED's distilled embedding, which is denoted as "Vanilla" in the table. It gets better classification ability than RKD, which verifies our approach gets high-quality embedding in the first stage.

LKD denotes the local version in REFILLED utilizing the local knowledge distillation but without decayed weights. It can further improve the vanilla training. Our REFILLED approach achieves the best classification performance in all cases. Benefited from reusing the knowledge from the teacher, the classification achieves a further improvement w.r.t. the vanilla training.

Table 2. The mean classification accuracy of knowledge distillation methods on CUB data set. Teacher is trained with ResNet-34, which gets 75.31% test accuracy. Student is learned with MobileNets, whose width multiplier is changed.

width multiplier	0.75	0.5	0.25
Student	74.87	72.41	69.72
KD [5]	76.02	74.17	71.97
FitNet [18]	75.03	72.17	70.03
AT [28]	76.11	72.94	70.99
NST [7]	75.89	73.82	71.92
KD+VID-I [1]	76.41	74.04	72.20
RKD [13]	76.11	75.24	72.84
REFILLED	78.01	75.90	73.15

3.2. Standard Knowledge Distillation

We use REFILLED to distill the knowledge from a cross-family teacher on CUB. We set the teacher as ResNet-34, and use the MobileNets with different width multipliers (from $\{0.75, 0.5, 0.25\}$) as the student model. Table 2 demonstrates the results, where we re-implement all the comparison methods in this case. REFILLED keeps its superiority in this case.

3.3. Few-Shot Learning and Middle-Shot Learning

The same configuration of REFILLED on the middle-shot learning tasks could also be applied to the few-shot learning scenarios. In addition to the middle-shot learning results, we provide additional few-shot learning results (where the number of shots is equal to 1 or 5) in Table 3. Both ProtoNet and FEAT are meta-learned over the pre-trained embeddings from the SEEN classes set (meta-train set). REFILLED gets better results when reusing a strong teacher (i.e., the ResNet), and works well with larger shots.

Table 3. The mean accuracy over 600 trials of few-shot and middle-shot tasks. We set the student model as the ConvNet, and investigate both ResNet and ConvNet as the teacher model, for our REFILLED approach. Detailed results and configurations are in the supp. REFILLED^1 denotes the results reusing a ResNet teacher and REFILLED^2 stands for the results reusing a ConvNet teacher.

Tasks	1-Shot 5-Way	5-Shot 5-Way	10-Shot 5-Way	30-Shot 5-Way
1NN	49.73	63.11	66.56	69.80
SVM	51.61	69.17	74.24	77.87
Fine-Tune	45.89	68.61	74.95	78.62
MAML [3]	48.70	63.11	-	-
ProtoNet [20]	51.79	70.38	74.42	78.10
FEAT [26]	55.15	71.61	74.86	78.84
REFILLED^1	54.82	71.97	76.42	80.33
REFILLED^2	53.44	70.60	75.37	78.94

3.4. Ablation Study, 1-Stage vs. 2-Stage Learning

We train REFILLED in a two-stage manner as [18, 1]. We discuss the pros and cons of the two ways first, and then provide the ablation study results in Table 4 and Table 5. Experiment settings in Table 4 and Table 5 are the same as those in Section 5.1 and Section 5.2 of the paper. In both tables, REFILLED$^\gamma$ means training REFILLED in a one-stage manner with balancing hyper-parameter γ, *i.e.*,

$$\min_{\phi_S, f_S} \gamma \sum_{ijk} \mathbf{KL}\big(\mathcal{P}_{ijk}(\phi_T) \parallel \mathcal{P}_{ijk}(\phi_S)\big) + \sum_{i=1}^{N} \ell\left(f_S(x_i), \mathbf{y}_i\right) + \lambda \mathbf{KL}\big(p_{\phi_T}(\mathbf{y}_i \mid \mathbf{x}_i),\ \mathbf{s}_\tau(f_S(\mathbf{x}_i))\big) \tag{9}$$

From the model design perspective: The two-stage training in REFILLED works well since the distilled discriminative embedding acts as a better initialization hence to *improves the discerning ability* of the model; while training with a combined objective *regularizes* the classifier by matching the predictions between student and teacher, so that it relies on a suitable strength of the regularization.

From the implementation perspective: An important issue for the joint training of the combined objective is to *set the right balance* among the embedding learning (relationship distillation), classification (cross-entropy), and knowledge transition (local knowledge distillation) losses. In our empirical study, it is a bit hard to tune the weights among them. While in the two-stage training strategy, we can fist learn a good embedding till convergence, and then use such embedding to initialize the second stage, where the balance between classification and distillation is solved with an annealing strategy.

From the results in both Table 4 and Table 5 on two different data sets, the two-stage training makes REFILLED easier to achieve higher performance. (The best result in each configuration is in bold).

Table 4. The mean accuracy of cross-task distillation on CUB data set. REFILLED$^{0.1}$ and REFILLED$^{0.2}$ denote the results using one-stage training protocol with balancing hyper-parameter 0.1 and 0.2 respectively.

Width Multiplier	1	0.75	0.5	0.25
Teacher	1NN: 45.31 , LR: 53.82 , FT: 65.72			
Student	71.25	67.56	66.85	64.48
RKD [13]	70.83	68.80	67.44	63.97
REFILLED	**73.38**	**70.42**	**69.77**	**67.10**
REFILLED$^{0.1}$	72.64	70.13	69.28	66.41
REFILLED$^{0.2}$	72.42	70.22	68.65	66.85

Table 5. The average classification results of knowledge distillation methods on CIFAR-100 data set based on the Wide-ResNet. REFILLED$^{0.1}$ and REFILLED$^{0.2}$ denote the results using one-stage training protocol with balancing hyper-parameter 0.1 and 0.2 respectively.

(depth, width)	(40, 2)	(16, 2)	(40, 1)	(16, 1)
Teacher	74.44			
Student	74.44	70.15	68.97	65.44
KD [5]	75.47	71.87	70.46	66.54
FitNet [18]	74.29	70.89	68.66	65.38
AT [28]	74.76	71.06	69.85	65.31
NST [7]	74.81	71.19	68.00	64.95
VID-I [1]	75.25	73.31	71.51	66.32
KD+VID-I [1]	76.11	73.69	72.16	67.19
RKD [13]	76.62	72.56	72.18	65.22
REFILLED	**77.49**	**74.01**	**72.72**	**67.56**
REFILLED$^{0.1}$	77.03	73.81	72.30	67.14
REFILLED$^{0.2}$	76.95	73.90	71.64	67.34

References

[1] Sungsoo Ahn, Shell Xu Hu, Andreas C. Damianou, Neil D. Lawrence, and Zhenwen Dai. Variational

information distillation for knowledge transfer. In *CVPR*, pages 9163–9171, 2019. 2, 4, 5, 6, 7

[2] Hanting Chen, Yunhe Wang, Chang Xu, Chao Xu, and Dacheng Tao. Learning student networks via feature embedding. *CoRR*, abs/1812.06597, 2018. 2

[3] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*, pages 1126–1135, 2017. 5, 7

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

[5] Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. *CoRR*, abs/1503.02531, 2015. 2, 6, 7

[6] 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. *CoRR*, abs/1704.04861, 2017. 4

[7] Zehao Huang and Naiyan Wang. Like what you like: Knowledge distill via neuron selectivity transfer. *CoRR*, abs/1707.01219, 2017. 6, 7

[8] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, pages 448–456, 2015. 5

[9] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009. 4

[10] Yufan Liu, Jiajiong Cao, Bing Li, Chunfeng Yuan, Weiming Hu, Yangxi Li, and Yunqiang Duan. Knowledge distillation via instance relationship graph. In *CVPR*, pages 7096–7104, 2019. 2

[11] R. Manmatha, Chao-Yuan Wu, Alexander J. Smola, and Philipp Krähenbühl. Sampling matters in deep embedding learning. In *CVPR*, pages 2859–2867, 2017. 1

[12] Boris N. Oreshkin, Pau Rodríguez López, and Alexandre Lacoste. TADAM: task dependent adaptive metric for improved few-shot learning. In *NeurIPS*, pages 719–729. 2018. 5

[13] Wonpyo Park, Dongju Kim, Yan Lu, and Minsu Cho. Relational knowledge distillation. In *CVPR*, pages 3967–3976, 2019. 2, 6, 7

[14] 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. 2

[15] Hang Qi, Matthew Brown, and David G. Lowe. Low-shot learning with imprinted weights. In *CVPR*, pages 5822–5830, 2018. 2, 5

[16] Siyuan Qiao, Chenxi Liu, Wei Shen, and Alan L. Yuille. Few-shot image recognition by predicting parameters from activations. In *CVPR*, pages 7229–7238, 2018. 5

[17] Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. In *ICLR*, 2017. 5

[18] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets. In *ICLR*, 2015. 6, 7

[19] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *CVPR*, pages 815–823, 2015. 1

[20] Jake Snell, Kevin Swersky, and Richard S. Zemel. Prototypical networks for few-shot learning. In *NeurIPS*, pages 4080–4090. 2017. 5, 7

[21] Frederick Tung and Greg Mori. Similarity-preserving knowledge distillation. In *ICCV*, pages 1365–1374, 2019. 2

[22] Laurens van der Maaten and Kilian Q. Weinberger. Stochastic triplet embedding. In *MLSP*, pages 1–6, 2012. 1

[23] Oriol Vinyals, Charles Blundell, Tim Lillicrap, Koray Kavukcuoglu, and Daan Wierstra. Matching networks for one shot learning. In *NeurIPS*, pages 3630–3638. 2016. 5

[24] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. 2011. 3

[25] Kilian Q. Weinberger and Lawrence K. Saul. Distance metric learning for large margin nearest neighbor classification. *JMLR*, 10:207–244, 2009. 1

[26] Han-Jia Ye, Hexiang Hu, De-Chuan Zhan, and Fei Sha. Learning embedding adaptation for few-shot learning. *CoRR*, 2018. 5, 7

[27] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In *BMVC*, 2016. 4

[28] Sergey Zagoruyko and Nikos Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. In *ICLR*, 2017. 6, 7