Appendix

A. Architectures Used in DB3KD and ZSDB3KD Experiments

We use several networks to evaluate the performance of DB3KD and ZSDB3KD. LeNet-5 (teacher)/ LeNet-5-Half (student) and AlexNet (teacher)/ AlexNet-Half (student) are used for both DB3KD and ZSDB3KD experiments (Table S1-S4). For the DB3KD experiments, we also design two student networks, i.e., LeNet-5-1/5 and AlexNet-Quarter for further evaluation (Table S5-S6). We also conduct experiments with ResNet-34 (teacher)/ ResNet-18 (student) with DB3KD and the architectures used are the same as the original ResNet architectures.

Index	Layer	Type	Feature map	Kernel size	Stride	Padding	Activation
0	Input	Input	1	-	-	-	-
1	conv1	conv	20	5x5	1	0	ReLu
2	maxpool1	pooling	-	2x2	2	1	-
3	conv2	conv	50	5x5	1	0	ReLu
4	maxpool2	pooling	-	2x2	2	1	-
5	fc1	fc	200	-	-	-	ReLu
6	fc2	fc	10	-	-	-	Softmax

Table S1. The architecture of the teacher model LeNet-5 with MNIST and Fashion-MNIST, for both DB3KD and ZSDB3KD experiments.

Index	Layer	Type	Feature map	Kernel size	Stride	Padding	Activation
0	Input	Input	1	-	-	-	-
1	conv1	conv	10	5x5	1	0	ReLu
2	maxpool1	pooling	-	2x2	2	1	-
3	conv2	conv	25	5x5	1	0	ReLu
4	maxpool2	pooling	-	2x2	2	1	-
5	fc1	fc	100	-	-	-	ReLu
6	fc2	fc	10	-	-	-	Softmax

Table S2. The architecture of the student model LeNet-5-Half with MNIST and Fashion-MNIST, for both DB3KD and ZSDB3KD experiments.

Index	Layer	Type	Feature map	Kernel size	Stride	Padding	Activation
0	Input	Input	1	-	-	-	-
1	conv1	conv	64	3x3	2	1	ReLu
2	maxpool1	pooling	-	3x3	2	0	-
3	bn1	batch norm	-	-	-	-	-
4	conv2	conv	192	3x3	1	2	ReLu
5	maxpool2	pooling	-	3x3	2	0	-
6	bn2	batch norm	-	-	-	-	-
7	conv3	conv	384	3x3	1	1	ReLu
8	bn3	batch norm	-	-	-	-	-
9	conv4	conv	256	3x3	1	1	ReLu
10	bn4	batch norm	-	-	-	-	-
11	conv5	conv	256	3x3	1	1	ReLu
12	maxpool3	pooling	-	3x3	2	0	-
13	bn5	batch norm	-	-	-	-	-
14	fc1	fc	4096	-	-	-	ReLu
15	bn6	batch norm	-	-	-	-	-
16	fc2	fc	4096	-	-	-	ReLu
17	bn7	batch norm	-	-	-	-	-
18	fc3	fc	10	-	-	-	Softmax

Table S3. The architecture of the teacher model AlexNet with CIFAR-10, for both DB3KD and ZSDB3KD experiments.

Index	Layer	Type	Feature map	Kernel size	Stride	Padding	Activation
0	Input	Input	1	-	-	-	-
1	conv1	conv	32	3x3	2	1	ReLu
2	maxpool1	pooling	-	3x3	2	0	-
3	bn1	batch norm	-	-	-	-	-
4	conv2	conv	96	3x3	1	2	ReLu
5	maxpool2	pooling	-	3x3	2	0	-
6	bn2	batch norm	-	-	-	-	-
7	conv3	conv	192	3x3	1	1	ReLu
8	bn3	batch norm	-	-	-	-	-
9	conv4	conv	128	3x3	1	1	ReLu
10	bn4	batch norm	-	-	-	-	-
11	conv5	conv	128	3x3	1	1	ReLu
12	maxpool3	pooling	-	3x3	2	0	-
13	bn5	batch norm	-	-	-	-	-
14	fc1	fc	2048	-	-	-	ReLu
15	bn6	batch norm	-	-	-	-	-
16	fc2	fc	2048	-	-	-	ReLu
17	bn7	batch norm	-	-	-	-	-
18	fc3	fc	10	-	-	-	Softmax

Table S4. The architecture of the student model AlexNet-Half with CIFAR-10, for both DB3KD and ZSDB3KD experiments.

Index	Layer	Type	Feature map	Kernel size	Stride	Padding	Activation
0	Input	Input	1	-	-	-	-
1	conv1	conv	4	5x5	1	0	ReLu
2	maxpool1	pooling	-	2x2	2	1	-
3	conv2	conv	10	5x5	1	0	ReLu
4	maxpool2	pooling	-	2x2	2	1	-
5	fc1	fc	40	-	-	-	ReLu
6	fc2	fc	10	-	-	-	Softmax

Table S5. The architecture of the student model LeNet-5-1/5 with MNIST and Fashion-MNIST, for DB3KD experiments.

Index	Layer	Type	Feature map	Kernel size	Stride	Padding	Activation
0	Input	Input	1	-	-	-	-
1	conv1	conv	16	3x3	2	1	ReLu
2	maxpool1	pooling	-	3x3	2	0	-
3	bn1	batch norm	-	-	-	-	-
4	conv2	conv	48	3x3	1	2	ReLu
5	maxpool2	pooling	-	3x3	2	0	-
6	bn2	batch norm	-	-	-	-	-
7	conv3	conv	96	3x3	1	1	ReLu
8	bn3	batch norm	-	-	-	-	-
9	conv4	conv	64	3x3	1	1	ReLu
10	bn4	batch norm	-	-	-	-	-
11	conv5	conv	64	3x3	1	1	ReLu
12	maxpool3	pooling	-	3x3	2	0	-
13	bn5	batch norm	-	-	-	-	-
14	fc1	fc	1024	-	-	-	ReLu
15	bn6	batch norm	-	-	-	-	-
16	fc2	fc	1024	-	-	-	ReLu
17	bn7	batch norm	-	-	-	-	-
18	fc3	fc	10	-	-	-	Softmax

Table S6. The architecture of the student model AlexNet-Quarter with CIFAR-10, for DB3KD experiments.

B. Experiment details

B.1. Training of the Models with Cross-Entropy

In this subsection, we introduce the details of training the models with cross-entropy loss, for both the pre-trained models used as the DB3 teachers, and the performance of the student models trained solely with the cross-entropy loss reported in Tables 1, 2, 3, and 4.

LeNet-5 on MNIST and Fashion-MNIST For the LeNet-5 architecture on MNIST and Fashion-MNIST, we train the teacher model for 200 epochs, with a batch size of 1024, an Adam optimizer with a learning rate of 0.001. For the student models trained with cross-entropy (reported in Tables 1 and 3), we use the same hyperparameters as above.

AlexNet on CIFAR-10 For the AlexNet architecture on CIFAR-10, we train the teacher model for 300 epochs, with a batch size of 1024 and an SGD optimizer. We set the momentum to 0.9, and weight decay to 0.0001. The learning rate is set to 0.1 at the beginning, and is divided by 10 at epochs 60, 120, and 180. For the student models trained with cross-entropy (reported in Tables 1 and 4), we use the same hyperparameters as above.

ResNet on CIFAR-100 For the ResNet-$\{50,34\}$ on CIFAR-100, we train the teacher models for 300 epochs, with a batch size of 256 and an SGD optimizer. We set the momentum to 0.9 and weight decay to 0.0001. The learning rate is set to 0.1 at the beginning, and is divided by 10 at epochs 60, 120, and 180. For the student model (ResNet-18) trained with cross-entropy (reported in Table 2), we use the same hyperparameters as above.

ResNet-34 on FLOWERS102 For the ResNet-34 architecture on FLOWERS102, we start with the model pre-trained on ImageNet, which is provided by Pytorch, and fine-tune the pre-trained model for 200 epochs with an SGD optimizer. We set the batch size to 64 and the momentum to 0.9. The learning rate is set to 0.01 at the beginning, and set to 0.005 and 0.001 at epochs 60 and 100, respectively. For the student model (Resnet-18) trained with cross-entropy (reported in Table 1), we use the same hyperparameters as above.

B.2. Standard Knowledge Distillation Training Details

For the standard knowledge distillation results reported in Tables 1, 2, 3, and 4, we train the student models via standard KD with the following hyperparameters. The scaling factor λ that balances the importance of cross-entropy loss and knowledge distillation loss is set to 1. The Adam optimizer is used for all experiments and the student networks are trained for 200 epochs with a temperature of 20. For the experiments with MNIST, Fashion-MNIST, and CIFAR-10, we set the batch size to 512; for the experiments with CIFAR-100 and FLOWERS102, we set the batch size to 64. The learning rate is set to 0.001 for MNIST and Fashion-MNIST, 0.005 for CIFAR-10/100, and 0.0005 for FLOWERS102.

B.3. Surrogate Knowledge Distillation Training Details

Training the student networks by transferring the knowledge from a surrogate, low-capacity white-box teacher whose parameters can be fully accessed is sensitive to hyperparameter selection. We did an extensive hyperparameter search in our experiments and report the best numbers in Table 1. We use the hyperparameters listed below. The optimizer and batch size used for surrogate KD are the same as in standard KD. We train the student models for 300 epochs for all experiments. For MNIST and Fashion-MNIST, the scaling factor λ is set to 0.7, the temperature is set to 3, and the learning rate is set to 0.005. For CIFAR-10/100, λ is set to 0.5, the temperature is set to 5, and the learning rate is set to 0.005. For FLOWERS102, λ is set to 1, the temperature is set to 10, and the learning rate is set to 0.001.

B.4. Data Augmentation Used in ZSDB3KD Experiments

In ZSDB3KD experiments, we found that data augmentation can improve the performance. Since the number of queries for the soft label construction of the samples is trivial to the performance, as shown in the DB3KD experiments (Fig. 4), we can apply various augmentation strategies to enrich the transfer set with affordable extra computing cost. In our study, we implement the following data augmentation strategies.

- **(1) Padding and crop.** We first pad two pixels on each side of the generated samples and crop it to the original size, starting from the upper left corner to the bottom right corner, with an interval of 1.
- **(2) Horizontal and vertical flip.** We flip the generated samples horizontally and vertically to create mirrored samples.

- **(3) Rotation.** We rotate each generated image starting from $-15°$ to $15°$ with an interval of $5°$ to create 6 more rotated samples.
- **(4) Flip after padding and crop.** We flip the images after (1), horizontally and vertically.
- **(5) Rotation after padding and crop.** We rotate the images after (1), using the same operation as (3).

For the MNIST and Fashion-MNIST datasets, only the strategies (1) and (2) are used. For the CIFAR-10 dataset, all five strategies are used. For the DB3KD experiment with CIFAR-100, we also use the above five strategies.

It is also worth mentioning that after generating images with the above operations, some of the samples' top-1 classes change to others. If this happens, we use the approach described in Section 3 to find the sample's corresponding point on the targeted decision boundary, i.e., x^*, to recover its top-1 class back to the top-1 class of the sample before augmentation.

Table S7 presents the performance comparison with and without data augmentation on each dataset used in the ZSDB3KD experiments. It is observed that training the student networks with more samples augmented with the above strategies can improve the performance.

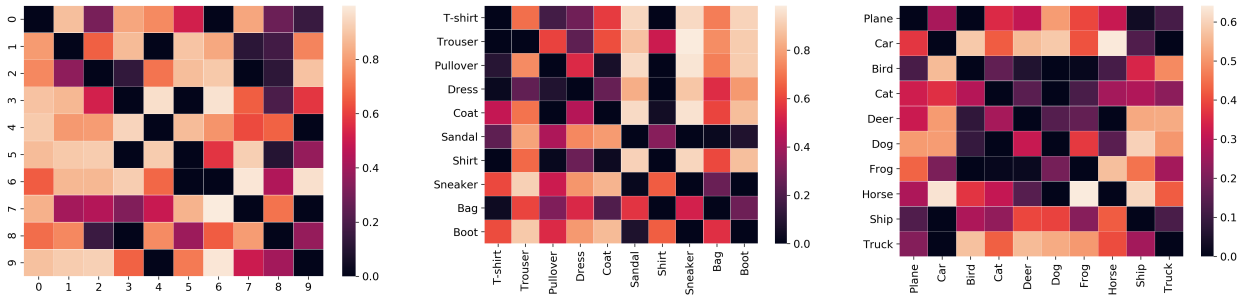
Dataset	Acc. without augmentation	Acc. with augmentation
MNIST	94.20%	96.54%
Fashion-MNIST	67.24%	72.31%
CIFAR-10	37.58%	59.46%

Table S7. Performance comparison of the ZSDB3KD experiments with and without data augmentation, with LeNet-5-Half on the MNIST and Fashion-MNIST datasets, and with AlexNet-Half on the CIFAR-10 dataset, respectively.

C. More Experiment Results

C.1. Comparison of the Sample Robustness Computed with DB3KD and the Logits Generated by the Teacher

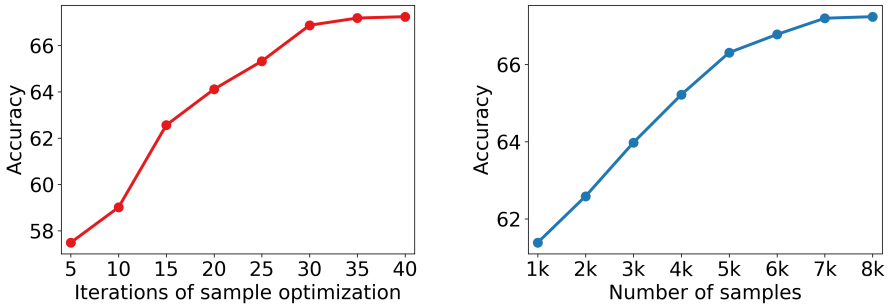
To further understand the effectiveness of the label construction with sample robustness in our DB3KD approach, we visualize the sample distances that are computed with the softmax outputs of the teacher networks, by accessing the teachers' parameters. We first feed the training samples to the teacher model and get the softmax output. For a training sample, if a bigger probability is assigned to a class, it means the distance between this sample to the specific class is smaller. Therefore, we simply use *1 - class probability* to represent the sample distance. The results are presented in Fig. S1. It can be observed that the visualized heatmaps look similar to those visualized with the sample robustness computed with our DB3 approach (Fig. 5(b-d)). For example, both of the MNIST heatmaps indicate that digit '4' is close to digit '9'. For the Fashion-MNIST, Fig. S1(b) shows that class T-shirt is semantically close to class 'Shirt' and 'Pullover', which is consistent with the results in Fig. 5(c). These results further validate that our proposed approach to construct soft labels with sample robustness is meaningful.



(a) MNIST (b) Fashion-MNIST (c) CIFAR-10

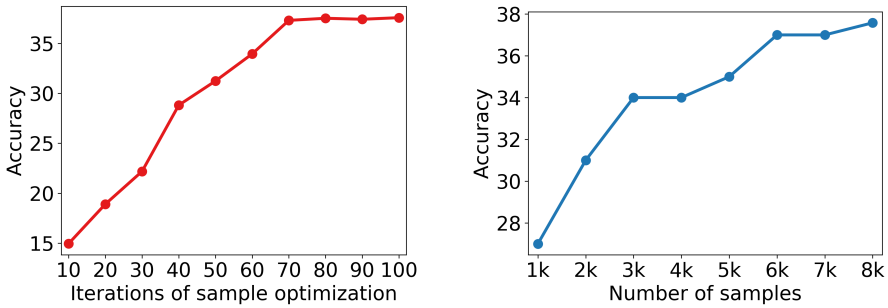
Figure S1. Normalized average distances of the samples of different classes, computed with the softmax outputs of the pre-trained teachers. Darker colors indicate smaller distances between two classes.

C.2. Ablation Studies of ZSDB3KD on Fashion-MNIST and CIFAR-10



(a) (b)

Figure S2. Performance of ZSDB3KD on the Fashion-MNIST dataset with (a) different numbers of iterations for sample generation and (b) pseudo samples used for KD training. Data augmentation is not used for the study.



(a) (b)

Figure S3. Performance of ZSDB3KD on the CIFAR-10 dataset with (a) different numbers of iterations for sample generation and (b) pseudo samples used for KD training. Data augmentation is not used for the study.

Similar to the ablation studies of ZSDB3KD on the MNIST dataset, we also investigate the effect of (1) different numbers of iterations for sample generation and (2) different numbers of pseudo samples used for KD training on the performance of the student networks (without using data augmentation). The results are presented in Fig. S2 and Fig. S3, respectively.

Similar to the results of MNIST, it is observed that, with more iterations for the sample optimization, more robust pseudo samples can be generated and the performance of the student networks are increased via DB3KD. For example, when optimizing the randomly generated noises for only 5 iterations, the performance of the student network on the Fashion-MNIST is less than 58% without data augmentation. After 40 iterations, the performance increases by around 7%. The performance of the AlexNet-Half network on CIFAR-10 is only around 15% when using pseudo samples that are optimized for only 10 iterations. On the other hand, the performance increases to 37% after 70 iterations.

The test accuracies of the student networks are also higher when using more pseudo samples as the transfer set. For the Fashion-MNIST dataset, the performance increases from 61.39% to 67.24% as the number of pseudo samples used as the transfer set increases from 1000 to 8000 per category. For the CIFAR-10 dataset, the performance is less than 28% when using only 1000 samples per class. When the number of samples for each class increases to 8000, an accuracy of 37.58% can be achieved.