A Datasets

We use MNIST [16], EMNIST [6], KMNIST and Kuzushiji-49 [5], CIFAR-10 and CIFAR-100 [13], and CUB [34] datasets. Example images are shown in Figure 4. MNIST includes images of 70000 handwritten digits that belong into 10 classes. EMNIST dataset includes various characters, but we choose EMNIST letters split that includes only letters. Lowercase and uppercase letters are combined together into 26 balanced classes (145600 examples in total). KMNIST (Kuzushiji-MNIST) is a dataset that includes images of 10 classes of cursive Japanese (*Kuzushiji*) characters and is of the same size as MNIST. Kuzushiji-49 is a larger version of KMNIST with 270912 examples and 49 classes. CIFAR-10 includes 60000 colour images of various general objects, for example airplanes, frogs or ships. As the name indicates, there are 10 classes. CIFAR-100 is like CIFAR-100, but has 100 classes with 600 images for each of them. Every class belongs to one of 20 superclasses which represent more general concepts. CUB includes colour images of 200 bird species. The number of images is small, only 11788. All datasets except Kuzushiji-49 are balanced or almost balanced.

Figure 4: Example images from the different datasets that we use.

B Analysis of simple one-layer case

In this section we analyse how synthetic labels are meta-learned in the case of a simple one-layer model with sigmoid output layer σ, second-order approach and binary classification problem. We will consider one example at a time for simplicity. The model has weights $\boldsymbol{\theta}$ and gives prediction $\hat{y} = \sigma(\boldsymbol{\theta}^T \boldsymbol{x})$ for input image $\boldsymbol{x}$ with true label y. We use binary cross-entropy loss L:

$$L\left(\hat{y}, y\right) = -y \log \hat{y} - (1-y) \log\left(1-\hat{y}\right).$$

As part of the algorithm, we first update the base model, using the current base example and synthetic label:

$$\boldsymbol{\theta}' = \boldsymbol{\theta} - \alpha \nabla_{\boldsymbol{\theta}} L\left(\sigma(\boldsymbol{\theta}^T \tilde{\boldsymbol{x}}), \tilde{y}\right),$$

after which we update the synthetic label:

$$\tilde{y} \leftarrow \tilde{y} - \beta \nabla_{\tilde{y}} L\left(\sigma(\boldsymbol{\theta}'^T \boldsymbol{x}), y\right).$$

Notation: $\tilde{\boldsymbol{x}}$ is the base example, $\tilde{y}$ is the synthetic label, α is the inner-loop learning rate, β is the outer-loop learning rate, $\boldsymbol{x}$ is an example from the target set, y is the label of the example and $\boldsymbol{\theta}$ describes the model weights.

Our goal is to intuitively interpret the update of the synthetic label, which uses the gradient $\nabla_{\tilde{y}} L\left(\sigma(\boldsymbol{\theta}'^T \boldsymbol{x}), y\right)$. We will repeatedly use the chain rule and the fact that

$$\frac{\partial \sigma(x)}{\partial x} = \sigma(x)\left(1 - \sigma(x)\right).$$

Moreover, we will use the following result (for binary cross-entropy loss L introduced earlier):

$$\begin{aligned}\frac{\partial L\left(\sigma(\boldsymbol{\theta}^T \boldsymbol{x}), y\right)}{\partial \boldsymbol{\theta}} &= \frac{\partial L\left(\sigma(\boldsymbol{\theta}^T \boldsymbol{x}), y\right)}{\partial \sigma(\boldsymbol{\theta}^T \boldsymbol{x})} \frac{\partial \sigma(\boldsymbol{\theta}^T \boldsymbol{x})}{\partial \boldsymbol{\theta}} \\ &= \left(\sigma(\boldsymbol{\theta}^T \boldsymbol{x}) - y\right) \boldsymbol{x}\end{aligned}$$

Now we derive an intuitive formula for the gradient used for updating the synthetic label:

$$\begin{aligned}\frac{\partial L\left(\sigma(\boldsymbol{\theta}'^T \boldsymbol{x}), y\right)}{\partial \tilde{y}} &= \frac{\partial L\left(\hat{y}', y\right)}{\partial \tilde{y}} = \left(\frac{\partial L\left(\hat{y}', y\right)}{\partial \hat{y}'} \frac{\partial \hat{y}'}{\partial \boldsymbol{\theta}'}\right)^T \frac{\partial \boldsymbol{\theta}'}{\partial \tilde{y}} \\ &= \left(\frac{\partial L\left(\hat{y}', y\right)}{\partial \hat{y}'} \frac{\partial \hat{y}'}{\partial \boldsymbol{\theta}'}\right)^T \frac{\partial \left(\boldsymbol{\theta} - \alpha \nabla_{\boldsymbol{\theta}} L\left(\sigma(\boldsymbol{\theta}^T \tilde{\boldsymbol{x}}), \tilde{y}\right)\right)}{\partial \tilde{y}} \\ &= \left(\frac{\partial L\left(\hat{y}', y\right)}{\partial \hat{y}'} \frac{\partial \hat{y}'}{\partial \boldsymbol{\theta}'}\right)^T \frac{\partial \left(\boldsymbol{\theta} - \alpha \left(\sigma(\boldsymbol{\theta}^T \tilde{\boldsymbol{x}}) - \tilde{y}\right) \tilde{\boldsymbol{x}}\right)}{\partial \tilde{y}} \\ &= \left(\frac{\partial L\left(\hat{y}', y\right)}{\partial \hat{y}'} \frac{\partial \hat{y}'}{\partial \boldsymbol{\theta}'}\right)^T (\alpha \tilde{\boldsymbol{x}}) = \left(\left(\hat{y}' - y\right) \boldsymbol{x}\right)^T (\alpha \tilde{\boldsymbol{x}}) \\ &= \alpha \left(\sigma(\boldsymbol{\theta}'^T \boldsymbol{x}) - y\right) \boldsymbol{x}^T \tilde{\boldsymbol{x}}\end{aligned}$$

The next step is to interpret the update rule. The update is proportional to the difference between the prediction on the real training set and the true label $\left(\sigma(\boldsymbol{\theta}'^T \boldsymbol{x}) - y\right)$ as well as to the similarity between the real training set example and the base example $\left(\boldsymbol{x}^T \tilde{\boldsymbol{x}}\right)$. This suggests the synthetic labels are updated so that they capture the different amount of similarity of a base example to examples from different classes in the target dataset. A similar analysis can also be done for our RR method – in such case the result would be similar and would include a further proportionality constant dependent on the base examples (not affecting the intuitive interpretation).

C Additional experimental details

Normalization We normalize greyscale images using the standardly used normalization for MNIST (mean of 0.1307 and standard deviation of 0.3081). All our greyscale images are of size 28×28. Colour images are normalized using CIFAR-10 normalization (means of about 0.4914, 0.4822, 0.4465, and standard deviations of about 0.247, 0.243, 0.261 across channels). All colour images are reshaped to be of size 32×32.

Computational resources Each experiment was done on a single GPU, in almost all cases NVIDIA 2080 Ti. Shorter (400 epochs) experiments took about 1 or 2 hours to run, while longer (800 epochs) experiments took between 2 and 4 hours.

Table 4: Comparison of training times of DD [35] and our LD (mins).

	MNIST	CIFAR-10
DD	116	205
LD	61	86
LD (RR)	65	98
Our DD	67	96
Our DD (RR)	72	90

Training time In Table 4 we compare training times of our framework and the original DD (using the same settings and hardware). Besides evaluating LD, we also use our meta-learning algorithm to implement an image-distillation strategy for direct comparison with the original DD. The results show our online approach significantly accelerates training. Our DD is comparable to LD, and both are faster than original DD. However, we

focus on LD because our version of DD was relatively unstable
(Table 17) and led to worse performance than LD, perhaps because learning synthetic images is more complex than synthetic labels. This shows we need both the labels and our re-initializing strategy.

In addition, Figure 5 illustrates the difference between a standard model used for second-order label distillation and a model that uses global ridge regression classifier weights (used for first-order RR label distillation). The two models are almost identical – only the final linear layer is different.

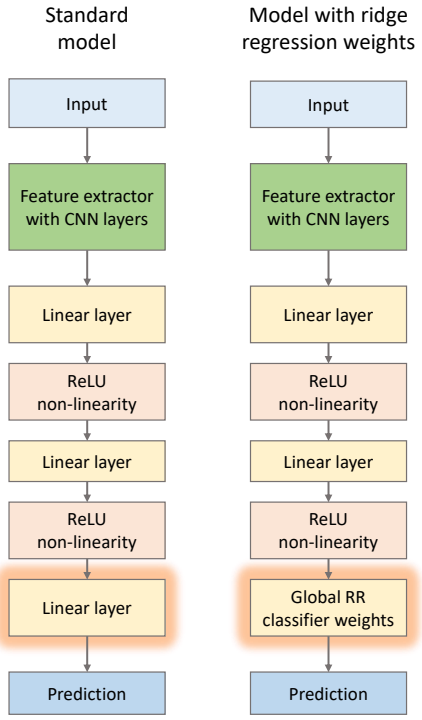


Figure 5: Comparison of a standard model used for second-order label distillation and a model that uses global ridge regression classifier weights (used for first-order RR label distillation).

D Additional experiments

Stability and dependence on choice of base examples To evaluate the consistency of our results, we repeat the entire pipeline and report the results in Table 5. In the previous experiments, we used one randomly chosen but fixed set of base examples per source task. We investigate the impact of base example choice by drawing further random base example sets. The results in Table 6 suggest that the impact of base example choice is slightly larger than that of the variability due to the distillation process, but still small overall. Note that the $\pm$ standard deviations in all cases quantify the impact of retraining from different random initializations at meta-test, given a fixed base set and completed distillation. It is likely that if the base examples were not selected randomly, the impact of using specific base examples would be larger. In fact, future work could investigate how to choose base examples so that learning synthetic labels for them improves the result further. We have tried the following strategy, but the label distillation results remained similar to the previous results:

- Try 50 randomly selected sets of examples, train a model with each three times (for robustness) and measure the validation accuracy.
- The validation accuracy is measured for various numbers of steps, in most cases we evaluate every 50 steps up to 1000 steps (or 1700 steps when there are more than 100 base examples).
- Select the set with the largest mean validation accuracy at any point of training (across the three runs).

- This strategy maximizes the performance of the baselines, but could potentially also help the label distillation since these examples could be generally better for training.

The results for this strategy are in Table 7.

Dependence on target dataset size Our experiments use a relatively large target dataset (about 50000 examples) for meta-learning. We study the impact of reducing the amount of target data for distillation in Table 8. Using 5000 or more examples (about 10% of the original size) is enough to achieve comparable performance.

Transferability of RR synthetic labels to standard model training When using RR, we train a validation and test model with RR and global classifier weights obtained using pseudo-gradient. In this experiment we study what happens if we create synthetic labels with RR, but do validation and testing with standard models trained from scratch without RR. For a fair comparison, we use the same synthetic labels for training a new RR model and a new standard model. Validation for early stopping is done with a standardly trained model. The results in Table 16 suggest RR labels are largely transferable (even in cross-dataset scenarios), but there is some decrease in performance. Consequently, it is better to learn the synthetic labels using second-order approach if we want to train a standard model without RR during testing (comparing with the results in Table 1, 2 and 3).

Intuition on cross-dataset distillation. To illustrate the mechanism behind cross-dataset distillation, we use the distilled labels to linearly combine base EMNIST example images weighted by their learned synthetic labels in order to estimate a prototypical KMNIST/MNIST target class example as implied by learned LD labels. Although the actual mechanism is more complex than this due to the non-linearity of the neural network, we can qualitatively see individual KMNIST/MNIST target classes are approximately encoded by their linear EMNIST LD prototypes as shown in Figure 9.

E Results of analysis

Our tables report the mean test accuracy and standard deviation (%) across 20 models trained from scratch using the base examples and synthetic labels. When analysing the original DD, 200 randomly initialized models are used.

Table 5: Repeatability. Label distillation is quite repeatable. Performance change from repeating the whole distillation learning and subsequent re-training is small. We used 100 base examples for these experiments. Datasets: E = EMNIST, M = MNIST.

	Trial 1	Trial 2	Trial 3
MNIST (LD)	87.27 ± 0.69	87.49 ± 0.44	86.77 ± 0.77
MNIST (LD RR)	87.85 ± 0.43	88.31 ± 0.44	88.07 ± 0.46
E $\rightarrow$ M (LD)	77.09 ± 1.66	76.81 ± 1.47	77.10 ± 1.74
E $\rightarrow$ M (LD RR)	82.70 ± 1.33	83.06 ± 1.43	81.46 ± 1.70

Table 6: Base example sensitivity. Label distillation has some sensitivity to the specific set of base examples (chosen by a specific random seed), but the sensitivity is relatively low. We used 100 base examples for these experiments. It is likely that label distillation would be more sensitive for a smaller number of base examples.

	Set 1	Set 2	Set 3	Set 4	Set 5
MNIST (LD)	84.91 ± 0.92	87.38 ± 0.81	87.49 ± 0.44	87.12 ± 0.47	85.16 ± 0.48
MNIST (LD RR)	87.82 ± 0.60	88.78 ± 0.57	88.31 ± 0.44	88.40 ± 0.46	87.77 ± 0.60
E $\rightarrow$ M (LD)	79.34 ± 1.36	74.55 ± 1.00	76.81 ± 1.47	78.59 ± 1.05	78.55 ± 1.32
E $\rightarrow$ M (LD RR)	81.67 ± 1.39	83.30 ± 1.38	83.06 ± 1.43	82.62 ± 1.70	83.43 ± 0.98

Table 7: Optimized base examples: within-dataset distillation recognition accuracy (%). Our label distillation (LD) outperforms prior Dataset Distillation [35] (DD) and SLDD [30], and scales to synthesizing more examples. The LD results remained similar to the original results even with optimized base examples.

	Base examples	10	20	50	100	200	500
MNIST	LD	66.96 ± 2.01	74.37 ± 1.65	83.17 ± 1.28	86.66 ± 0.44	90.75 ± 0.49	93.22 ± 0.41
	Baseline	56.60 ± 3.10	64.77 ± 1.90	77.33 ± 2.51	84.86 ± 1.16	88.33 ± 1.04	92.87 ± 0.67
	Baseline LS	60.44 ± 2.05	66.41 ± 2.14	80.54 ± 1.94	86.98 ± 0.99	91.12 ± 0.79	95.56 ± 0.18
	LD RR	71.34 ± 2.19	73.34 ± 1.18	84.66 ± 0.89	88.30 ± 0.46	88.91 ± 0.36	89.73 ± 0.39
	Baseline RR	59.20 ± 2.18	65.22 ± 2.29	77.34 ± 1.68	84.70 ± 0.81	87.87 ± 0.68	92.39 ± 0.53
	Baseline RR LS	60.63 ± 1.64	65.61 ± 1.08	77.39 ± 1.67	85.63 ± 0.95	88.89 ± 0.88	94.33 ± 0.44
	DD				79.5 ± 8.1		
	SLDD				82.7 ± 2.8		
CIFAR-10	LD	26.65 ± 0.94	29.07 ± 0.62	35.03 ± 0.48	38.17 ± 0.36	42.12 ± 0.56	41.90 ± 0.28
	Baseline	17.57 ± 1.63	21.66 ± 0.91	23.59 ± 0.80	27.79 ± 1.01	33.49 ± 0.77	40.44 ± 1.33
	Baseline LS	18.57 ± 0.68	22.91 ± 0.70	24.57 ± 0.83	29.27 ± 0.85	34.83 ± 0.75	40.15 ± 0.66
	LD RR	25.08 ± 0.39	28.17 ± 0.34	34.43 ± 0.38	37.59 ± 1.68	42.48 ± 0.25	44.81 ± 0.26
	Baseline RR	18.42 ± 0.59	21.00 ± 0.73	22.45 ± 0.49	24.46 ± 1.67	30.96 ± 0.49	39.17 ± 0.47
	Baseline RR LS	18.22 ± 0.67	22.31 ± 1.01	22.27 ± 0.75	24.84 ± 2.89	30.74 ± 0.80	38.86 ± 0.88
	DD				36.8 ± 1.2		
	SLDD				39.8 ± 0.8		

Table 8: Dependence on real training set size. Around 5000 examples ($\approx 10\%$ of all data) is sufficient. Similarly as before, we used 100 base examples. Using all examples means using 50000 examples.

Target examples	100	500	1000	5000	10000	20000	All
E → M (LD)	50.70 ± 2.33	61.92 ± 3.62	57.39 ± 4.58	75.44 ± 1.60	76.79 ± 1.12	77.27 ± 1.25	77.09 ± 1.66
E → M (LD RR)	60.67 ± 3.17	72.09 ± 2.40	65.71 ± 3.77	76.83 ± 2.33	80.66 ± 1.97	82.44 ± 1.64	82.70 ± 1.33

Table 9: Sensitivity to number of training steps at meta-testing. We re-train the model with different numbers of steps than estimated during meta-training. The results show our method is relatively insensitive to the number of steps. The default number of steps T_i (+ 0 column) was estimated as 278 for MNIST (LD), 217 for MNIST (LD RR), 364 steps for E → M (LD) and 311 steps for E → M (LD RR). Scenario with 100 base examples is reported.

Steps deviation	- 50	- 20	- 10	+ 0	+ 10	+ 20	+ 50	+ 100
MNIST (LD)	86.67 ± 0.51	86.91 ± 0.49	86.88 ± 0.50	86.77 ± 0.77	86.54 ± 0.68	87.05 ± 0.67	86.98 ± 0.70	86.59 ± 0.63
MNIST (LD RR)	88.21 ± 0.50	88.03 ± 0.51	88.32 ± 0.50	88.07 ± 0.46	88.10 ± 0.38	87.95 ± 0.45	87.98 ± 0.45	87.74 ± 0.62
E → M (LD)	77.28 ± 1.01	76.88 ± 1.97	77.26 ± 1.92	77.10 ± 1.74	76.40 ± 2.37	76.76 ± 2.18	77.80 ± 1.49	77.81 ± 1.23
E → M (LD RR)	81.84 ± 1.75	81.64 ± 1.76	81.45 ± 2.01	81.46 ± 1.70	81.55 ± 1.80	80.96 ± 1.74	81.34 ± 1.64	80.73 ± 2.45

Table 10: Sensitivity of original DD to number of steps. DD is very sensitive to using the specific number of steps. We take the first N steps, keep their original learning rates, and assign learning rates of 0 to the remaining steps. When we do 5 more steps than the original (30), we perform the final 5 steps with an average learning rate.

Steps	10	15	20	25	30	35
MNIST	35.65 ± 11.19	43.25 ± 11.47	54.85 ± 12.28	52.54 ± 11.40	77.32 ± 5.08	53.89 ± 10.51
CIFAR-10	21.14 ± 4.08	22.50 ± 4.24	27.08 ± 3.22	28.54 ± 2.41	35.20 ± 1.09	28.80 ± 3.33

Table 11: Sensitivity of original DD to learning rates. DD is sensitive to using the specific learning rates.

Learning rate	Optimized	Average of optimized
MNIST	77.32 ± 5.08	62.38 ± 13.11
CIFAR-10	35.20 ± 1.09	30.59 ± 3.95

Table 12: Sensitivity of original DD to order of examples. DD is sensitive to using the specific order of training examples.

Order	Original	Shuffled within epoch	Shuffled across epochs
MNIST	77.32 ± 5.08	50.20 ± 12.83	62.67 ± 10.82
CIFAR-10	35.20 ± 1.09	24.65 ± 2.16	22.59 ± 3.23

Table 13: Transferability of distilled labels across different architectures (second-order method). The upper part of the table shows performance of various test models when trained on distilled labels synthetised with AlexNet only. The middle part shows the baseline performance of training models with different architectures on true labels. The lower part shows that distilled labels work even in cross-dataset scenario (labels trained with AlexNet only). The results clearly suggest the distilled labels generalize across different architectures.

Base examples	10	20	50	100	200	500
CIFAR-10 LD						
AlexNet	26.09 ± 0.58	30.41 ± 0.81	35.21 ± 0.50	38.39 ± 0.62	40.98 ± 0.50	42.78 ± 0.29
LeNet	19.33 ± 2.50	24.17 ± 1.42	28.14 ± 1.37	32.65 ± 1.22	36.60 ± 1.46	39.67 ± 0.85
ResNet-18	17.97 ± 1.23	24.64 ± 0.92	27.36 ± 1.07	31.01 ± 0.84	35.33 ± 0.97	39.32 ± 0.61
CIFAR-10 baseline						
AlexNet	14.35 ± 1.39	16.72 ± 0.76	21.08 ± 0.93	25.39 ± 0.86	31.39 ± 1.12	37.17 ± 1.58
LeNet	13.20 ± 1.86	15.31 ± 1.09	18.15 ± 0.82	21.63 ± 1.45	25.87 ± 1.26	32.99 ± 0.86
ResNet-18	13.80 ± 1.19	18.29 ± 1.43	20.56 ± 0.75	23.44 ± 1.04	28.98 ± 1.05	33.16 ± 1.12
CUB to CIFAR-10 LD						
AlexNet	25.95 ± 0.90	27.73 ± 1.09	31.00 ± 0.78	34.99 ± 0.69	37.83 ± 0.65	39.44 ± 0.53
LeNet	20.18 ± 1.56	23.15 ± 1.72	26.16 ± 1.55	28.73 ± 1.44	30.71 ± 1.80	35.41 ± 0.88
ResNet-18	17.12 ± 1.32	17.80 ± 1.26	21.22 ± 1.04	23.38 ± 0.95	23.39 ± 0.90	26.71 ± 0.89

Table 14: Transferability of distilled labels across different architectures (RR method). The upper part of the table shows performance of various test models when trained on distilled labels synthetised with AlexNet only. The middle part shows the baseline performance of training models with different architectures on true labels. The lower part shows that distilled labels work even in cross-dataset scenario (labels trained with AlexNet only). The results clearly suggest the distilled labels generalize across different architectures. Note that lower RR results for ResNet-18 may be caused by significantly lower dimensionality of the RR layer (64 features + 1 for bias), while AlexNet and LeNet have 192 features + 1 for bias in the RR layer.

Base examples	10	20	50	100	200	500
CIFAR-10 LD						
AlexNet	26.78 ± 0.84	29.51 ± 0.41	34.71 ± 0.45	38.29 ± 0.92	41.14 ± 0.37	42.71 ± 0.27
LeNet	20.24 ± 2.06	23.58 ± 1.62	28.05 ± 1.66	30.64 ± 1.47	34.52 ± 1.16	38.75 ± 0.96
ResNet-18	16.63 ± 0.88	17.98 ± 1.38	23.03 ± 1.21	26.66 ± 0.73	31.08 ± 0.92	36.37 ± 0.81
CIFAR-10 baseline						
AlexNet	13.37 ± 0.75	17.20 ± 0.50	19.07 ± 0.75	24.72 ± 0.53	29.94 ± 0.65	36.20 ± 0.97
LeNet	12.33 ± 0.88	14.28 ± 0.74	17.31 ± 0.97	20.61 ± 1.13	24.15 ± 0.98	28.92 ± 0.80
ResNet-18	14.04 ± 1.20	16.60 ± 1.25	18.75 ± 1.17	22.61 ± 1.03	28.03 ± 0.72	33.72 ± 1.74
CUB to CIFAR-10 LD						
AlexNet	26.08 ± 1.14	29.37 ± 0.36	31.46 ± 3.94	35.74 ± 0.81	37.26 ± 1.63	40.94 ± 4.61
LeNet	22.69 ± 2.09	24.42 ± 1.53	26.35 ± 1.26	29.84 ± 1.36	31.68 ± 1.09	35.66 ± 1.98
ResNet-18	16.23 ± 1.44	17.44 ± 0.84	20.06 ± 0.74	23.48 ± 1.00	21.31 ± 0.82	29.86 ± 0.93

Table 15: Sensitivity of original DD to a change in architecture (trained with AlexNet). The same order of examples used as during training, with the optimized learning rates. We have not been able to easily integrate ResNet-18 to the implementation provided by the authors [35].

	AlexNet	LeNet
CIFAR-10	35.20 ± 1.09	25.92 ± 2.35

Table 16: Transferability of RR synthetic labels to standard model training (both within-dataset and cross-dataset scenarios evaluated). 100 base examples used. Datasets: E = EMNIST, M = MNIST, K= KMNIST, B = CUB, C = CIFAR-10, K-49 = Kuzushiji-49.

	RR model	Standard model
MNIST	88.25 ± 0.37	87.07 ± 0.64
CIFAR-10	38.40 ± 0.41	37.16 ± 0.59
CIFAR-100	10.96 ± 1.08	8.93 ± 0.27
E $\rightarrow$ M	80.09 ± 1.84	76.04 ± 2.29
E $\rightarrow$ K	58.14 ± 0.91	50.37 ± 2.75
B $\rightarrow$ C	34.81 ± 6.46	35.76 ± 0.54
E $\rightarrow$ K-49	17.56 ± 1.62	13.28 ± 1.62

Table 17: Dataset distillation of images. The results show we have not obtained strong and stable results when distilling synthetic images rather than labels (likely because of the complexity of the flexible task). It may be possible to obtain better results on distilling images with our approach, but it likely requires a lot more tuning than we have done.

Base examples	10	20	50	100
MNIST (DD)	55.48 ± 4.42	17.90 ± 4.54	34.40 ± 4.49	34.44 ± 6.15
MNIST (DD RR)	22.33 ± 2.75	49.30 ± 2.72	30.16 ± 5.32	36.43 ± 5.50
CIFAR-10 (DD)	12.99 ± 1.41	12.09 ± 0.99	16.01 ± 1.48	17.92 ± 1.71
CIFAR-10 (DD RR)	20.20 ± 0.38	22.71 ± 0.79	26.04 ± 1.22	27.05 ± 1.86

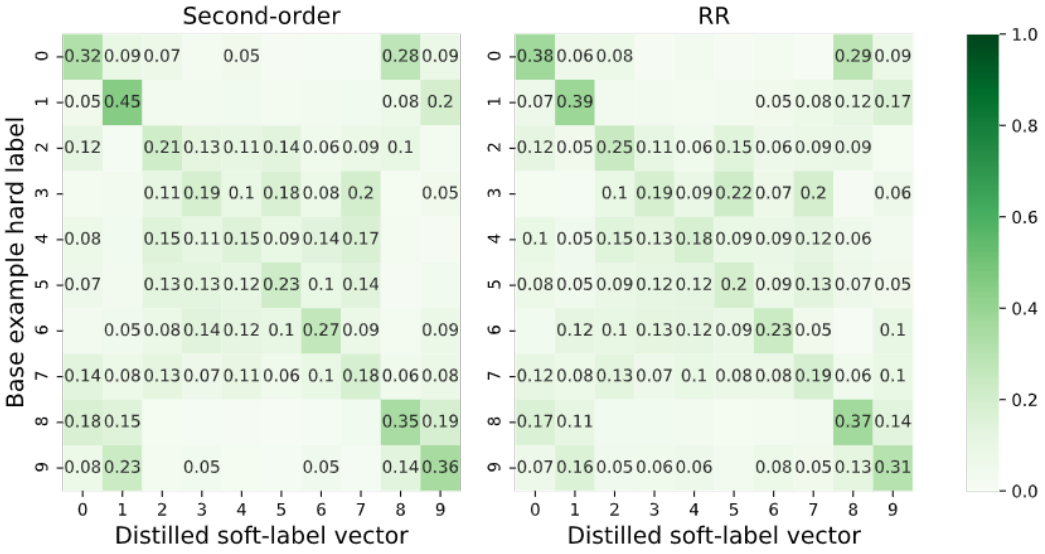


Figure 6: Distribution of label values across base example hard labels and distilled soft-label vectors. Within-dataset CIFAR-10 scenario with 100 base examples is shown. We can see that to a certain extent the original classes are recovered, but a lot of non-trivial information is added that presumably leads to strong improvements over a baseline with true or smooth labels. Numbers are shown when the values are at least 0.05.

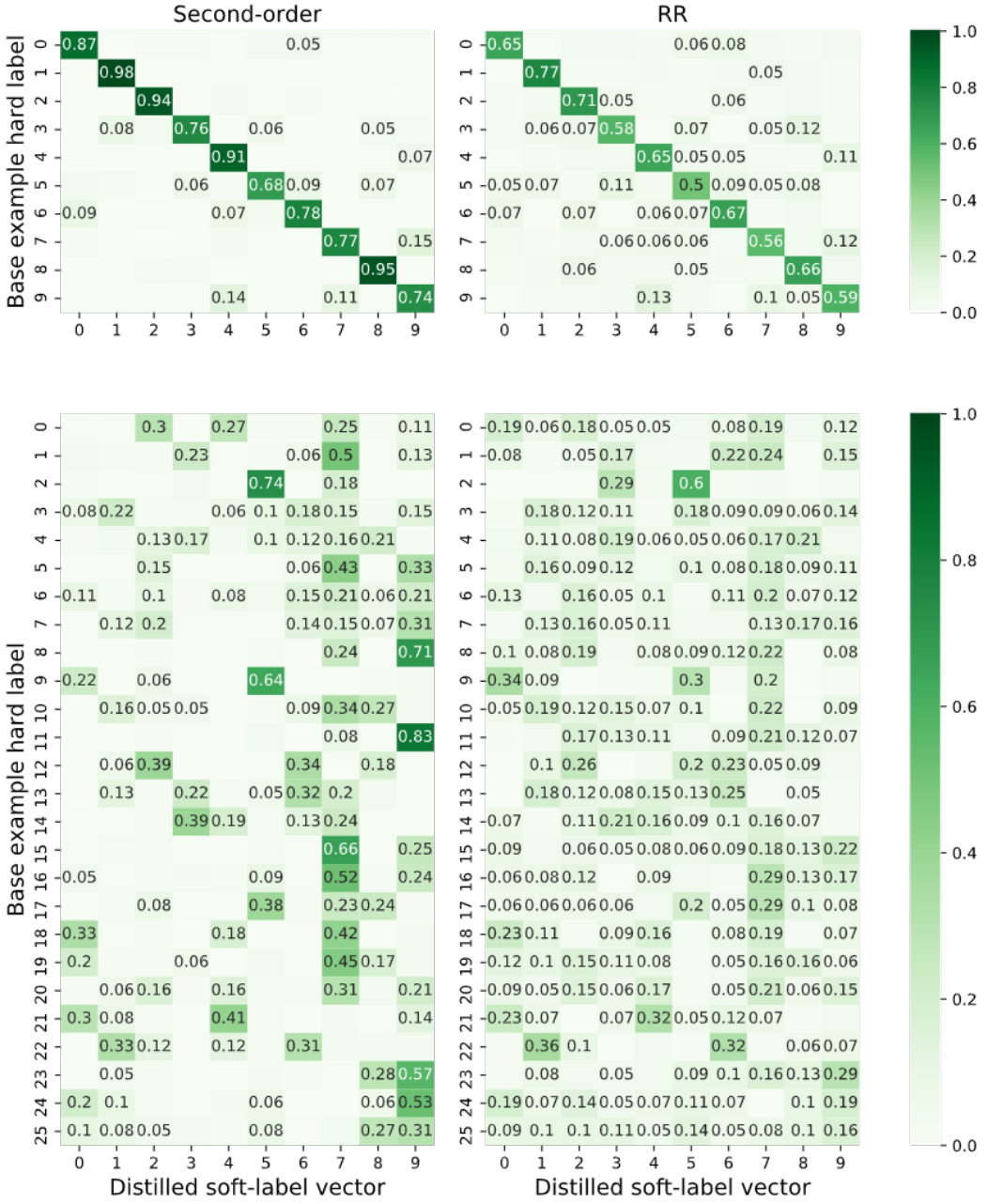
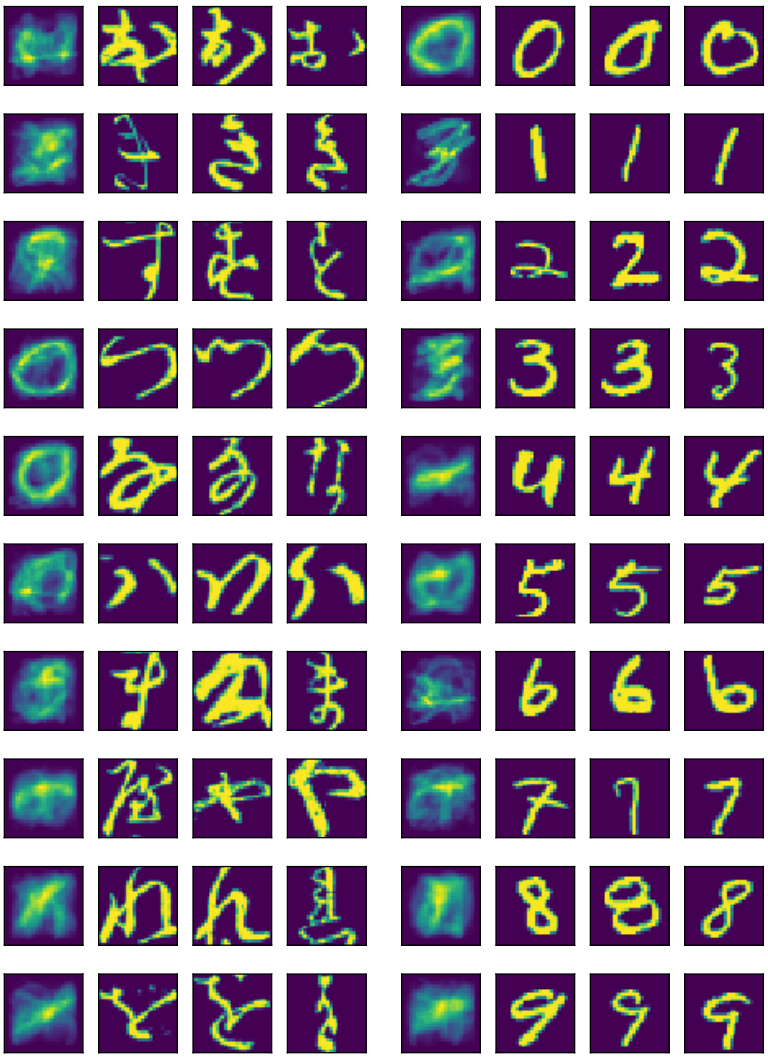


Figure 7: Distribution of label values across base example hard labels and distilled soft-label vectors. The upper row shows the mean distilled labels for different original classes for within-dataset MNIST scenario with 100 base examples. We can see that to a large extent the original labels are preserved with an additional noise on visually similar classes such as 4 and 9. At the same time, some non-trivial information is learned, especially for our RR method. The lower row shows the mean distilled labels for different original EMNIST ("English") classes used to recognize KMNIST ("Japanese") characters. 100 base examples scenario. Numbers are shown when the values are at least 0.05.

	Second-order	RR
	[0.00, 0.00, 0.01, 0.99, 0.00, 0.01, 0.00, 0.00, 0.00, 0.00]	[0.00, 0.01, 0.01, 0.76, 0.00, 0.00, 0.00, 0.20, 0.01, 0.00]
	[0.00, 0.00, 0.00, 0.00, 0.02, 0.00, 0.00, 0.02, 0.00, 0.95]	[0.00, 0.00, 0.00, 0.00, 0.36, 0.00, 0.00, 0.32, 0.06, 0.26]
	[0.00, 0.93, 0.00, 0.00, 0.00, 0.00, 0.00, 0.07, 0.00, 0.00]	[0.00, 0.73, 0.00, 0.05, 0.02, 0.00, 0.01, 0.07, 0.04, 0.09]
	[0.00, 0.00, 0.02, 0.00, 0.00, 0.00, 0.00, 0.00, 0.97, 0.00]	[0.00, 0.00, 0.13, 0.01, 0.01, 0.14, 0.17, 0.00, 0.54, 0.00]
	[0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.99, 0.00, 0.00]	[0.19, 0.07, 0.00, 0.00, 0.00, 0.05, 0.06, 0.56, 0.00, 0.06]
	[0.35, 0.00, 0.02, 0.02, 0.03, 0.01, 0.00, 0.18, 0.04, 0.34]	[0.26, 0.00, 0.01, 0.09, 0.04, 0.00, 0.02, 0.19, 0.12, 0.27]
	[0.00, 0.42, 0.03, 0.01, 0.14, 0.01, 0.31, 0.00, 0.06, 0.03]	[0.00, 0.23, 0.10, 0.00, 0.19, 0.02, 0.20, 0.01, 0.12, 0.13]
	[0.00, 0.00, 0.08, 0.01, 0.00, 0.01, 0.00, 0.79, 0.00, 0.11]	[0.01, 0.00, 0.07, 0.08, 0.00, 0.01, 0.29, 0.49, 0.00, 0.05]
	[0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.02, 0.01, 0.97]	[0.06, 0.13, 0.22, 0.00, 0.03, 0.00, 0.00, 0.04, 0.19, 0.32]
	[0.00, 0.01, 0.17, 0.29, 0.21, 0.01, 0.14, 0.16, 0.00, 0.01]	[0.00, 0.11, 0.16, 0.16, 0.11, 0.00, 0.21, 0.22, 0.00, 0.04]
	[0.03, 0.47, 0.04, 0.03, 0.03, 0.04, 0.03, 0.02, 0.11, 0.21]	[0.00, 0.48, 0.07, 0.01, 0.02, 0.11, 0.00, 0.01, 0.14, 0.17]
	[0.00, 0.01, 0.31, 0.00, 0.29, 0.01, 0.34, 0.03, 0.00, 0.01]	[0.00, 0.04, 0.25, 0.05, 0.19, 0.03, 0.38, 0.06, 0.01, 0.00]
	[0.61, 0.02, 0.04, 0.00, 0.01, 0.00, 0.00, 0.00, 0.32, 0.00]	[0.42, 0.09, 0.10, 0.01, 0.01, 0.00, 0.00, 0.00, 0.35, 0.02]
	[0.03, 0.01, 0.25, 0.07, 0.24, 0.15, 0.10, 0.14, 0.00, 0.01]	[0.00, 0.02, 0.15, 0.08, 0.10, 0.18, 0.16, 0.27, 0.00, 0.03]
	[0.01, 0.05, 0.02, 0.27, 0.07, 0.35, 0.10, 0.08, 0.02, 0.03]	[0.03, 0.01, 0.08, 0.25, 0.12, 0.31, 0.02, 0.16, 0.02, 0.00]
	[0.04, 0.37, 0.01, 0.04, 0.01, 0.05, 0.07, 0.03, 0.11, 0.28]	[0.03, 0.15, 0.04, 0.11, 0.04, 0.11, 0.11, 0.05, 0.16, 0.20]
	[0.01, 0.00, 0.24, 0.06, 0.32, 0.08, 0.13, 0.15, 0.00, 0.00]	[0.04, 0.03, 0.16, 0.10, 0.23, 0.07, 0.11, 0.22, 0.02, 0.02]
	[0.18, 0.12, 0.08, 0.08, 0.06, 0.07, 0.03, 0.06, 0.17, 0.14]	[0.20, 0.06, 0.12, 0.12, 0.06, 0.09, 0.05, 0.07, 0.13, 0.10]
	[0.00, 0.02, 0.13, 0.18, 0.13, 0.17, 0.18, 0.18, 0.00, 0.01]	[0.02, 0.04, 0.18, 0.14, 0.10, 0.13, 0.19, 0.18, 0.00, 0.02]
	[0.00, 0.01, 0.06, 0.09, 0.04, 0.20, 0.06, 0.52, 0.00, 0.01]	[0.06, 0.10, 0.12, 0.10, 0.09, 0.12, 0.05, 0.26, 0.00, 0.10]

Figure 8: Examples of distilled labels for both second-order and RR label distillation. Scenarios: 1) within-dataset MNIST, 2) cross-dataset EMNIST ("English") source to KMNIST ("Japanese") target, 3) within-dataset CIFAR-10, 4) cross-dataset CUB (birds) source to CIFAR-10 target. Five base examples from each scenario are shown in the order described.



(a) KMNIST (b) MNIST

Figure 9: Cross-dataset task: reconstructed images from KMNIST (Japanese letters) and MNIST (digits) based on combination of EMNIST base examples (English letters) (100 base examples used). Each row corresponds to a separate class, while the leftmost column shows the reconstructed image and the other three columns show actual examples from the same class. One image is reconstructed for each target dataset class. The base example images are combined pixel-wise with proportions based on the element of the synthetic label vector corresponding to the class that is being reconstructed. Note that KMNIST images in the same class can look very different because of different ways of writing the character, which makes reconstruction more challenging. Some of the reconstructed images resemble images from the target dataset classes, which shows that LD learns labels that combine base examples so that their pixel-wise combination, weighted based on the synthetic class labels, looks similar to the target class images.