Appendix

Include extra information in the appendix. This section will often be part of the supplemental material. Please see the call on the NeurIPS website for links to additional guides on dataset publication.

1. Submission introducing new datasets must include the following in the supplementary materials:
   (a) Dataset documentation and intended uses. Recommended documentation frameworks include datasheets for datasets, dataset nutrition labels, data statements for NLP, and accountability frameworks.
   Response: Here is a link to a folder that the reviewers can access the data: link will be shared with reviewers privately. Inside this folder, there is an IPython notebook, named "noisy_label_datasets_and_rater_features.ipynb", which contains detailed documentation for the datasets, including the dataset size, number of examples, definitions of each feature and the rater features. The intended use of the datasets is noisy label research.
   (b) URL to website/platform where the dataset/benchmark can be viewed and downloaded by the reviewers.
   Response: Here is a link to a folder that the reviewers can access the data: link will be shared with reviewers privately.
   (c) Author statement that they bear all responsibility in case of violation of rights, etc., and confirmation of the data license.
   Response: Yes, hereby the author of this paper claim that we bear all responsibility in case of violation of rights. The data licenses of the datasets that we use in this paper can be found in Appendix E. We are working on making our datasets publicly available. We intend to use the CC0 Creative Commons License for the the datasets that we generate. This license will be applied to the materials that we created, namely the noisy labels and rater features. The original images and labels are under their original dataset licenses.
   (d) Hosting, licensing, and maintenance plan. The choice of hosting platform is yours, as long as you ensure access to the data (possibly through a curated interface) and will provide the necessary maintenance.
   Response: We are planning to make the datasets publicly available by August 1, 2021. We intend to store the data on Google Cloud Platform (GCP). We mentioned the licensing plan in the previous point. The authors of this paper will be responsible for the maintenance of the datasets.
2. To ensure accessibility, the supplementary materials for datasets must include the following:
   (a) Links to access the dataset and its metadata. This can be hidden upon submission if the dataset is not yet publicly available but must be added in the camera-ready version. In select cases, e.g when the data can only be released at a later date, this can be added afterward. Simulation environments should link to (open source) code repositories.
   Response: Here is a link to a folder that the reviewers can access the data: link will be shared with reviewers privately. Inside this folder, there is an IPython notebook, named "noisy_label_datasets_and_rater_features.ipynb", which contains detailed documentation for the datasets, including the dataset size, number of examples, definitions of each feature and the rater features. Once we make the data publicly available, there will be a GitHub repository for this dataset which contains a link to the GCP bucket for the downloadable data.
   (b) The dataset itself should ideally use an open and widely used data format. Provide a detailed explanation on how the dataset can be read. For simulation environments, use existing frameworks or explain how they can be used.
   Response: Inside this folder linked above, there is an IPython notebook, named "noisy_label_datasets_and_rater_features.ipynb", which contains metadata for the datasets and detailed examples for reading the datasets, including the noisy label data and the rater features.
   (c) Long-term preservation: It must be clear that the dataset will be available for a long time, either by uploading to a data repository or by explaining how the authors themselves will ensure this.

Response: Yes, we plan to make the datasets available to the general public for a long time. Once we make the data publicly available, there will be a GitHub repository for this dataset which contains a link to the GCP bucket for the downloadable data. GCP bucket is a stable location for data storage. The authors of this paper will be responsible for the maintenance of the datasets.

(d) Explicit license: Authors must choose a license, ideally a CC license for datasets, or an open source license for code (e.g. RL environments).

Response: We intend to use the CC0 Creative Commons License for the the datasets that we generate. This license will be applied to the materials that we created, namely the noisy labels and rater features. The original images and labels are under their original dataset licenses.

(e) Add structured metadata to a dataset's meta-data page using Web standards (like schema.org and DCAT): This allows it to be discovered and organized by anyone. If you use an existing data repository, this is often done automatically.

Response: The dataset metadata are currently presented in the IPython notebook mentioned above. Once the datasets are publicly available,we will add a meta-data page using Web standards.

(f) Highly recommended: a persistent dereferenceable identifier (e.g. a DOI minted by a data repository or a prefix on identifiers.org) for datasets, or a code repository (e.g. GitHub, GitLab,...) for code. If this is not possible or useful, please explain why.

Response: We will have a GitHub repository that provides the link to the datasets and example code for loading the data. The GitHub repository will be online at the same time when we make the datasets publicly available.

3. For benchmarks, the supplementary materials must ensure that all results are easily reproducible. Where possible, use a reproducibility framework such as the ML reproducibility checklist, or otherwise guarantee that all results can be easily reproduced, i.e. all necessary datasets, code, and evaluation procedures must be accessible and documented.

   Response: The main focus of our paper is a framework for generating synthetic noisy label datasets; thus our paper should be considered as a *dataset* paper rather than *benchmarking* paper. Meanwhile, we made our datasets available to the reviewers through the URL above. We are also working on make the data publicly available. We also provide sufficient details in Appendix B and Section 4 for reproducing our experimental results.

4. For papers introducing best practices in creating or curating datasets and benchmarks, the above supplementary materials are not required.

   Response: Our paper can be considered as practices for creating datasets. We made our datasets available to the reviewers and are working on making them publicly available.

A Performance of existing noisy label algorithms

With our instance-dependent synthetic noisy label datasets, a follow-up question is how existing techniques for mitigating the impact of label noise perform on our benchmarks. In particular, we are interested in the difference of the algorithms' performance when using our synthetic datasets and using noisy label datasets with independent random label noise. In this section, when we mention a dataset uses random label noise, we mean with certain probability (rater error rate), the label of each data point is flipped to an incorrect label that is uniformly selected. This flipping event is independent of other data points and the image itself.

A.1 Experiment setup

We compare the following 5 algorithms: vanilla training with cross-entropy loss (Baseline), Bootstrap [Reed et al., 2014], Co-Teaching [Han et al., 2018], cross-entropy loss with Monte Carlo sampling (MCSoftMax) [Collier et al., 2020], and MentorMix [Jiang et al., 2020][3] on 4 tasks: CI-

[3] We also experimented with F-Correction [Patrini et al., 2017] and RoG [Lee et al., 2019] but did not observe significant improvement over the baseline on our synthetic datasets. Thus, we choose to not report the results of these two algorithms.

FAR10, CIFAR100, PatchCamelyon, and Cats vs Dogs.[4] For each task, we generate 3 synthetic noisy label datasets with different amount of noise using our framework. According to the rater error rate, the noisy label datasets are marked as "low", "medium", and "high" in Figures 9 and 10. Details for these datasets can be found in Appendix B. For each of our synthetic dataset, we generate another dataset that uses random label noise and has the *same* rater error rate. We compare the performance of the 5 algorithms on these paired datasets, and aim to measure the difficulty of noisy label datasets when the label errors are generated using our framework or independent random flipping. All the experiments use the ResNet50 architecture.

A.2 Results

Interestingly, we find different behavior for tasks with different number of classes. For tasks with a large number of classes such as CIFAR100, we find that most algorithms achieve better test accuracy on our synthetic datasets compared to random label noise. On binary classification problems such as PatchCamelyon and Cats vs Dogs, however, the trend is opposite, i.e., most algorithms perform worse on our synthetic datasets. On CIFAR10, we observe mixed behavior: depending on the amount of noise and the algorithm, the test accuracy can be higher either on our synthetic datasets or those with random label noise. The results are shown in Figure 9, and exact numbers are provided in Appendix D.

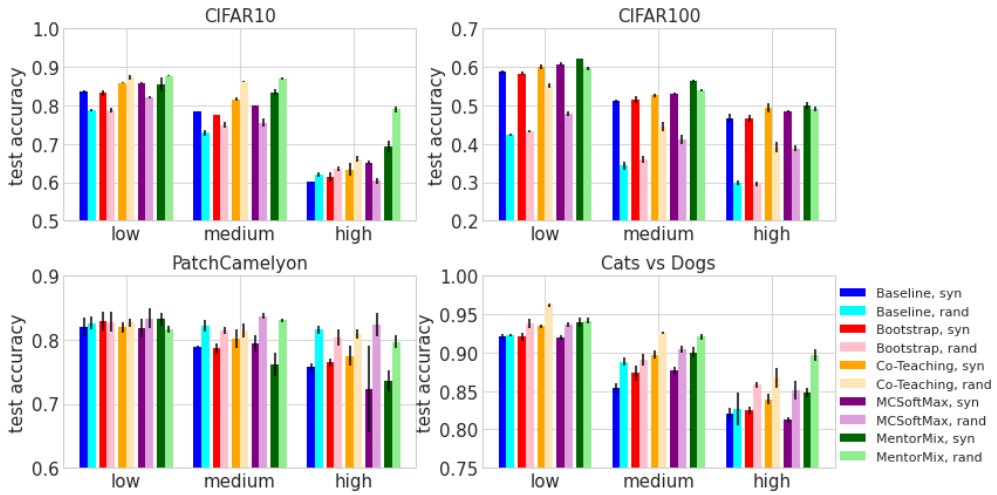

Figure 9: Benchmarking noisy label algorithms using our synthetic dataset and random label noise. Each pair of adjacent bars shows test accuracy on two datasets: our synthetic dataset (darker color) and random label noise (lighter color) with the same rater error rate. On CIFAR100, our datasets are easier than random noise, while on binary classification tasks (PCam and CvD), our datasets are harder.

This phenomenon can be explained as follows. For binary classification problems, in our synthetic framework, the mislabeled data are usually the ambiguous ones that located around the decision boundary. This label noise can hurt the models' performance more since the important information around the decision boundary is corrupted. On the contrary, for tasks with a large number of classes, especially those with tree-structured classes involving a relatively small number of high level super classes and low level fine-grained classes, such as CIFAR100, in our instance-dependent simulation framework, the label mistakes are usually among similar classes. For example, an image of a certain type of mammal may be mislabeled as another mammal, but it is unlikely to be labeled as a type of vehicles. In other words, the corruption of decision boundary only happens to similar fine-grained classes in our framework. Thus, given the same fractions of incorrect labels are the same, our synthetic label noise hurts the models' performance less compared to random noise.

[4] We also generated synthetic datasets using ImageNet. However, none of the noisy label techniques performs significantly better than vanilla training with cross entropy loss, thus we do not present the results here.

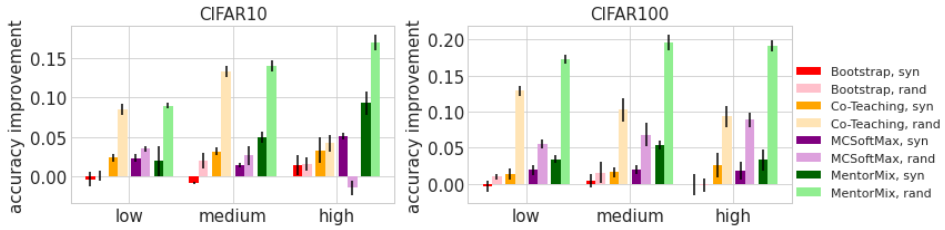

Figure 10: Improvement in test accuracy using noisy label techniques. Each pair of adjacent bars shows test accuracy improvement compared to the baseline on two datasets: our synthetic dataset (darker color) and random label noise (lighter color) with the same rater error rate. In most cases, the accuracy improvement tends to be smaller under our synthetic framework.

Another observation is that on CIFAR10 and CIFAR100, the performance improvement obtained by noisy label algorithms when compared with the baseline is usually smaller with our synthetic datasets. The performance improvement is presented in Figure 10.

We emphasize that our results demonstrate the importance of using more realistic synthetic benchmarks in the research on label noise: existing algorithms exhibit different behavior on our synthetic framework and random label noise, even if the fraction of mislabeled data is kept the same, and the performance gain observed using random label noise may not directly translate to a more realistic setting that we tested.

B Details of synthetic datasets

In this section, we provide more details of the synthetic data generation process. In particular, we provide the architectures and hyperparameters of the rater models in these datasets. All the models use standard cosine learning rate decay schedule, as well as the standard flipping and cropping data augmentation. In the following, for rater models that use the same architecture, they are randomly initialized independently.

For the CIFAR10 dataset in Section 2.3, we use 10 rater models, including 3 Inception-v1 [Szegedy et al., 2015], 1 Inception-v3 [Szegedy et al., 2016], 2 Inception-ResNet-v2 [Szegedy et al., 2017], 2 MobileNet-v1 [Howard et al., 2017], 2 VGG16 [Simonyan and Zisserman, 2014] models. The models are trained with batch size 256, $80,000$ steps, and initial learning rate 0.01. For the CIFAR100 dataset, we use the "low noise" dataset in Section 4 with details given in following paragraphs.

For the PCam dataset in Section 3.3, we use 20 rater models involving 10 architectures: Inception-v1, Inception-v2 [Szegedy et al., 2016], Inception-v3, Inception-v4 [Szegedy et al., 2017], MobileNet-v1, MobileNet-v2 [Sandler et al., 2018], ResNet50, ResNet152 [He et al., 2016], VGG16, VGG19. For each architecture, we use two different initial learning rates: 0.01 and 0.001 to train two different models. All the models are trained with batch size 256 and $10,000$ steps.

For the CvD dataset in Section 3.3, we use 10 rater models, involving the same 10 architectures in the PCam dataset mentioned above. All the models are trained with batch size 128, initial learning rate 0.001, and $10,000$ steps.

For the *Easy* task based on CIFAR100 in Section 3.3, we use 10 rater models with the following architectures: Inception-v1, Inception-v2, Inception-v3, Inception-v4, Inception-ResNet-v2, MobileNet-v1, MobileNet-v2, ResNet50, ResNet101, ResNet152. We use batch size 128, initial learning rate 3×10^{-5}, and $5,000$ training steps. For the *Medium* task, we use 11 rater models, each using its own architecture. The 11 architectures include the 10 architectures for the PCam dataset in Section 3.3 with an additional ResNet101. We use batch size 128, initial learning rate 0.003, and $40,000$ steps. The *Hard* task uses 11 rater models with the same architectures as the *Medium* task, with batch size 128, initial learning rate 0.01 and 2×10^5 steps.

For the 3 CvD datasets in Section 3.1, we use 10 rater models with the same architectures as the PCam dataset in Section 3.3. All models are trained with batch size 128. For the three datasets, the (initial learning rate, number of steps) pairs are $(1 \times 10^{-2}, 5 \times 10^4)$, $(1 \times 10^{-3}, 2.5 \times 10^4)$, $(1 \times 10^{-3}, 1 \times 10^5)$, respectively.

For each of the 4 PCam datasets in Section 3.1, we use 20 raters models, which uses the same combinations of the 10 architectures and 2 initial learning rates as in the PCam dataset in Section 3.3. They all use batch size 256. The 4 datasets are generated by varying the number of training steps among $\{1, 2, 5, 8\} \times 10^4$.

In Section 3.2, we use 3 CvD datasets and 3 CIFAR10 datasets. All the CvD datasets use 10 rater models with the same set of architectures as the CvD dataset in Section 3.3. All rater models are trained with batch size 128. The (initial learning rate, number of steps) pairs are $(1 \times 10^{-3}, 1 \times 10^5)$, $(1 \times 10^{-3}, 2.5 \times 10^4)$, $(1 \times 10^{-2}, 1 \times 10^4)$, respectively. The CIFAR10 dataset with rater error rate 0.11 is the same as the dataset in Section 2.3. The CIFAR10 dataset with rater error rate 0.19 has 10 rater models, including 2 Inception-v4, 2 MobileNet-v1, 2 MobileNet-v2, 1 NASNetMobile [Zoph et al., 2018], 1 ResNet50, 1 ResNet101, 1 VGG16. All models are trained with batch size 256, initial learning rate 0.01 and $17,000$ steps. The CIFAR10 dataset with rater error rate 0.33 has 10 rater models, including 2 Inception-v2, 1 Inception-ResNet-v2, 2 MobileNet-v1, 1 MobileNet-v2, 2 ResNet50, 1 ResNet101, 1 ResNet152. All models are trained with batch size 256, initial learning rate 0.01 and $12,000$ steps.

In Sections A and 4, we use 3 datasets for each of the 4 tasks. The rater error raters of these datasets are provided in the tables in Appendix D. Here, we provide details of the rater models in the synthetic datasets.

CIFAR10 Low noise: the same as the CIFAR10 dataset in Section 2.3; medium noise: the same as the CIFAR10 dataset with rater error rate 0.19 in Section 3.2; high noise: the same set of architectures as the CIFAR10 dataset with rater error rate 0.33 in Section 3.2, and the batch size is 256, initial learning rate is 0.01, and number of steps is $5,000$.

CIFAR100 For all the 3 CIFAR100 datasets, we use 11 raters, with the same set of architectures as the *Medium* and *Hard* tasks in Section 3.3. The (batch size, learning rate, number of steps) tuples for the low, medium, and high noise datasets are $(128, 1 \times 10^{-3}, 1 \times 10^4)$, $(256, 0.01, 2 \times 10^5)$, and $(256, 0.01, 8 \times 10^4)$, respectively.

PCam For all the 3 PCam datasets, we use 20 raters models (for the medium noise dataset, one of the Inception-v1 models failed due to system error, so we only have 19 noisy labels for this dataset), which uses the same combinations of the 10 architectures and 2 initial learning rates as in the PCam dataset in Section 3.3. They all use batch size 256 and initial learning rate 0.01. The number of steps are 3.5×10^4, 1.5×10^4, and 1×10^4, for the low, medium, and high noise datasets, respectively.

CvD For all the 3 CvD datasets, we use 10 rater models with the same set of architectures as the CvD dataset in Section 3.3. All rater models are trained with batch size 128. The (initial learning rate, number of steps) pairs are $(1 \times 10^{-3}, 5 \times 10^4)$, $(1 \times 10^{-2}, 1 \times 10^4)$, $(1 \times 10^{-3}, 1 \times 10^4)$, for low, medium, and high noise, respectively.

C Details for three CIFAR100-based datasets in Section 3.3

As we know, the CIFAR100 dataset contains 20 super classes, each of which contains 5 fine-grained classes. We create the easy, medium, and hard tasks in the following way.

- For the easy task, we select one fine-grained class from each of the 20 super classes, and form a 20-way classification task.
- For the medium task, we select 4 super classes that are semantically similar, i.e., large carnivores, large omnivores and herbivores, small mammals, and medium-sized mammals. We use all the fine-grained classes from these 4 super classes to form a 20-way classification task.
- For the hard task, we simply classify the 20 super classes, and we randomly subsample the data in order to match the total number of data points in the other two tasks. We note that this task is harder since the data in each super class is a mixture of 5 fine-grained classes.

D Experimental results

We provide exact numbers for the experimental results in Appendix A (Tables 1, 2, 3, and 4) and Section 4 in the main paper (Tables 5, 6, 7, and 8).

Table 1: Test accuracy ± std (%) of noisy label algorithms on CIFAR10

noise	low (err=0.11)		medium (err=0.19)		high (err=0.48)	
dataset	synthetic	random	synthetic	random	synthetic	random
Baseline	83.6 ± 0.5	78.7 ± 0.3	78.4 ± 0.1	72.9 ± 0.7	60.1 ± 0.2	61.9 ± 0.6
Bootstrap	83.2 ± 0.8	78.8 ± 0.6	77.6 ± 0.1	74.8 ± 0.7	61.5 ± 1.2	63.5 ± 0.7
Co-Teaching	85.9 ± 0.2	87.2 ± 0.6	81.6 ± 0.5	86.2 ± 0.2	63.4 ± 1.6	66.1 ± 0.9
MCSoftMax	85.9 ± 0.1	82.2 ± 0.3	79.8 ± 0.2	75.5 ± 1.1	65.2 ± 0.4	60.4 ± 0.8
MentorMix	85.5 ± 1.9	87.7 ± 0.2	83.4 ± 0.7	86.9 ± 0.3	69.4 ± 1.5	78.9 ± 0.9

Table 2: Test accuracy ± std (%) of noisy label algorithms on CIFAR100

noise	low (err=0.25)		medium (err=0.38)		high (err=0.43)	
dataset	synthetic	random	synthetic	random	synthetic	random
Baseline	58.8 ± 0.5	42.3 ± 0.3	51.1 ± 0.5	34.4 ± 1.1	46.7 ± 1.2	29.9 ± 0.6
Bootstrap	58.4 ± 0.6	43.3 ± 0.3	51.5 ± 0.8	35.9 ± 0.9	46.6 ± 0.9	29.6 ± 0.7
Co-Teaching	60.1 ± 0.6	55.2 ± 0.7	52.7 ± 0.5	44.6 ± 1.2	49.2 ± 1.2	39.2 ± 1.4
MCSoftMax	60.7 ± 0.4	47.8 ± 0.5	53.0 ± 0.4	41.2 ± 1.2	48.5 ± 0.3	38.8 ± 0.8
MentorMix	62.2 ± 0.1	59.6 ± 0.6	56.4 ± 0.3	53.9 ± 0.4	50.0 ± 0.8	49.0 ± 0.5

E Data license

The synthetic datasets that we generate in this paper are based on the following 4 existing public datasets: CIFAR10 [Krizhevsky and Hinton, 2009], CIFAR100 [Krizhevsky and Hinton, 2009], PatchCamelyon [Veeling et al., 2018, Bejnordi et al., 2017], and Cats vs Dogs [Elson et al., 2007].

The original CIFAR10 and CIFAR100 datasets do not have licenses. However, given the wide use of these two datasets in the research community, we do not believe there are license issues with using these datasets for research and publishing noisy label datasets based on them. The PatchCamelyon dataset is under the CC0 Creative Commons License, which allows the use of this dataset for research purpose, distribution, and modification of the dataset. The Cats vs Dogs dataset has a license that permits the use for research purposes; analysing and testing purposes; and publishing (or presenting papers/articles) on the results from the dataset. Following the license, when we publish our noisy label datasets, we will not include the raw images and labels from the original Cats vs Dogs dataset. Instead, we will only publish the noisy labels and rater features.

In this paper, we also used the CIFAR10-H dataset [Peterson et al., 2019]. This dataset is under the Creative Commons BY-NC-SA 4.0 license, which allows the data to be used for non-commercial research purposes.

We plan to publish the synthetic noisy label datasets that we generated. The current plan is to have our datasets under the CC0 Creative Commons License. This license will be applied to the materials that we created, namely the noisy labels and rater features. The original images and labels are under their original dataset licenses.

Table 3: Test accuracy ± std (%) of noisy label algorithms on PatchCamelyon

noise	low (err=0.10)		medium (err=0.18)		high (err=0.23)	
dataset	synthetic	random	synthetic	random	synthetic	random
Baseline	82.1 ± 1.3	82.6 ± 1.0	78.9 ± 0.2	82.2 ± 1.0	75.7 ± 0.7	81.6 ± 0.6
Bootstrap	82.9 ± 1.4	82.8 ± 1.6	78.6 ± 0.9	81.4 ± 0.5	76.4 ± 0.7	80.3 ± 1.3
Co-Teaching	82.0 ± 0.8	82.7 ± 0.6	80.1 ± 1.4	81.4 ± 1.2	77.5 ± 1.5	80.9 ± 0.8
MCSoftMax	81.9 ± 1.5	83.4 ± 1.6	79.4 ± 1.2	83.7 ± 0.5	72.3 ± 6.7	82.3 ± 1.8
MentorMix	83.2 ± 1.1	81.7 ± 0.6	76.2 ± 1.9	83.0 ± 0.3	73.6 ± 1.6	79.7 ± 1.0

Table 4: Test accuracy ± std (%) of noisy label algorithms on Cats vs Dogs

noise	low (err=0.09)		medium (err=0.20)		high (err=0.29)	
dataset	synthetic	random	synthetic	random	synthetic	random
Baseline	92.0 ± 0.4	92.3 ± 0.2	85.4 ± 0.7	88.8 ± 0.5	82.0 ± 0.7	82.7 ± 2.1
Bootstrap	92.1 ± 0.6	93.8 ± 0.6	87.3 ± 1.0	89.0 ± 0.8	82.4 ± 0.6	85.8 ± 0.4
Co-Teaching	93.4 ± 0.2	96.2 ± 0.2	89.7 ± 0.5	92.6 ± 0.2	84.0 ± 0.6	86.7 ± 1.3
MCSoftMax	92.0 ± 0.3	93.7 ± 0.3	87.6 ± 0.5	90.4 ± 0.5	81.2 ± 0.4	85.1 ± 1.2
MentorMix	94.0 ± 0.6	94.2 ± 0.4	90.0 ± 0.7	92.0 ± 0.4	84.8 ± 0.6	89.6 ± 0.8

Table 5: Training with LQM outputs with various techniques. Test accuracy ± std (%) on CIFAR10

algorithm	low (err=0.11)	medium (err=0.19)	high (err=0.48)
Baseline	84.1 ± 0.2	78.8 ± 0.2	62.4 ± 1.1
LQM	85.6 ± 0.4	81.9 ± 0.3	73.4 ± 0.3
LQM + Bootstrap	82.8 ± 0.9	79.8 ± 0.2	73.5 ± 0.3
LQM + Co-Teaching	86.3 ± 0.2	80.9 ± 0.1	74.2 ± 0.4
LQM + MCSoftMax	85.7 ± 0.2	81.6 ± 0.1	74.2 ± 0.2
LQM + MentorMix	86.3 ± 0.1	84.2 ± 0.3	78.4 ± 0.2

Table 6: Training with LQM outputs with various techniques. Test accuracy ± std (%) on CIFAR100

algorithm	low (err=0.25)	medium (err=0.38)	high (err=0.43)
Baseline	59.2 ± 1.2	52.9 ± 0.6	47.3 ± 1.0
LQM	59.4 ± 1.7	53.9 ± 0.6	51.3 ± 0.9
LQM + Bootstrap	59.8 ± 1.1	52.8 ± 0.8	49.8 ± 1.4
LQM + Co-Teaching	61.0 ± 1.3	56.8 ± 0.6	50.4 ± 0.7
LQM + MCSoftMax	63.0 ± 0.2	57.0 ± 0.4	55.2 ± 0.7
LQM + MentorMix	62.4 ± 0.5	58.5 ± 0.2	53.7 ± 0.4

Table 7: Training with LQM outputs with various techniques. Test accuracy ± std (%) on Patch-Camelyon

algorithm	low (err=0.1)	medium (err=0.18)	high (err=0.23)
Baseline	78.1 ± 1.8	75.1 ± 0.9	72.9 ± 0.3
LQM	80.0 ± 0.1	79.1 ± 0.9	77.6 ± 0.7
LQM + Bootstrap	80.7 ± 0.7	80.7 ± 0.9	78.2 ± 1.1
LQM + Co-Teaching	78.9 ± 0.1	78.2 ± 1.7	76.9 ± 0.3
LQM + MCSoftMax	80.3 ± 0.4	80.0 ± 0.7	78.2 ± 0.9
LQM + MentorMix	79.5 ± 0.5	77.0 ± 0.4	75.8 ± 0.2

Table 8: Training with LQM outputs with various techniques. Test accuracy $\pm$ std (%) on Cats vs Dogs

algorithm	low (err=0.09)	medium (err=0.20)	high (err=0.29)
Baseline	91.9 ± 0.3	87.6 ± 0.5	82.0 ± 0.5
LQM	94.9 ± 0.4	92.3 ± 0.4	85.8 ± 0.4
LQM + Bootstrap	92.0 ± 0.4	90.5 ± 0.3	89.5 ± 0.8
LQM + Co-Teaching	94.1 ± 0.8	91.5 ± 0.5	90.9 ± 0.5
LQM + MCSoftMax	93.5 ± 0.2	91.3 ± 0.2	89.1 ± 0.1
LQM + MentorMix	94.6 ± 0.6	92.1 ± 0.5	91.1 ± 0.3