Supplementary Material: Learning From Noisy Labels By Regularized Estimation Of Annotator Confusion

A. Data sets, training and architectures

Data sets. In this work, we verified our method on three classification datasets: MNIST digit classification dataset [1]; CIFAR-10 object recognition dataset [2]; the cardiac view classification (CVC) dataset from a handheld ultra-sound probe. The MNIST dataset consists of $60,000$ training and $10,000$ testing examples, all of which are 28×28 grayscale images of digits from 0 to 9. The CIFAR-10 dataset consists of $50,000$ training and $10,000$ testing examples, all of which are 32×32 coloured natural images drawn from 10 classes. The CVC data set contains $26,2000$ training and $20,000$ test examples, which are grayscale images of size 96×96 from 6 different cardiac views. Each image is labelled by a subset of 8 annotators (6 sonographers and 2 non-experts).

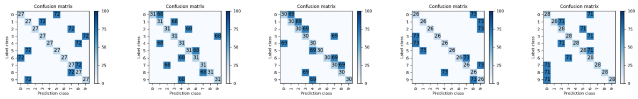
Training. For all experiments, we employ the same training scheme unless otherwise stated. We optimize parameters using Adam [3] with initial learning rate of 10^{-3} and $\beta = [0.9, 0.999]$, with minibatches of size 50 and train for 200 epochs. For our method, we set the scale of the trace regularization to $\lambda = 0.01$. For the training of the EM-based approaches (Model-Bootstrapped EM [4] and generalized EM [5]), we train the base classifier for 200 epochs in total over the course of the EM steps, following the same protocol above. For CIFAR-10, we performed two iterations of EM algorithm ($T = 2$) and 100 epochs worth of gradient descent steps during each E-step to update the parameters of the base classifier ($G = 100$ epochs), following the original implementation in [4]. For the experiments on the CVC data set, we run more rounds of EM with $T = 10$ and $G = 20$ epochs. In all cases, we hold out 10% of training images as a validation set and best model is selected based on the validation accuracy over the course of training. No data augmentation is performed during training in all three data sets. We note that for CIFAR-10, we, in addition, decreased the learning rate by a factor of 10 at every multiple of 50 in a similar fashion to the schedule used in [6, 7, 8].

Architectures. For MNIST, the base classifier was defined as a CNN architecture comprised of 4 convolution layers, each with 3×3 kernels follower by Relu. The number of kerners in respective layers are $\{32, 32, 64, 64\}$. After the first two convolution layers, we perform 2×2 max-pooling, and after the last one, we further downsample the features with Global Average Pooling (GAP) prior to the final fully connected layer. For the CVC dataset, we employed the same architecture, but with increased number of kernels i.e. $\{128, 128, 128, 128\}$. For CIFAR-10, we used a 50-layer ResNet [7].

B. Confusion matrices of *pairwise-flippers* and *hammer-spammers*

For MNIST experiments, we considered two different models of annotator types: (i) *pairwise-flipper* and (ii) *hammer-spammer*. Example confusion matrices for both cases are shown in Fig. 1. For each annotator type and skill level p, we create a group of 5 annotators by generating confusion matrices (CMs) from the associated distribution. More specifically, each CM is generated by perturbing the mean skill level p by injecting a small Gaussian noise $\epsilon \sim \text{Normal}(0, 0.01)$ and choosing the flipping target class randomly in the case of a *pairwise-flipper*.

(a) Pairwise-flippers with mean skill-level $p = 0.3$



(b) Spammer-hammers skill-level $p = 0.5$

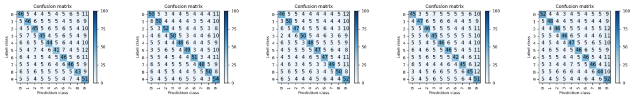


Figure 1: Examples of annotator groups. The value of diagonal entries are fixed constant for each annotator and is drawn from $\text{Normal}(p, 10^{-2})$.

C. Additional experiments on MNIST

We present results of experiments on MNIST where models are trained on noisy labels from groups of 5 "hammer-spammers" for a range of mean skil level p. Fig. 2 shows a comparison of our method against other EM-based approaches, while Fig. 3 compares our method against other noise-robust methods without explicit modelling of individual annotators. Our method consistently achieves comparable or better accuracy with respect to the baselines.

D. Ablation study on trace regularization on MNIST

We compare our method on MNIST against the case where the trace norm regularization is removed (results on CIFAR-10 and CVC datasets are given in the main text). Fig. 4 shows that adding the trace norm generally improves the performance in terms of both classification accuracy and CM estimation error, and such improvement is pronounced in the presence of larger noise i.e. lower skill levels of annotators. We also observe that when the noise level is low, our model still attains very high accuracy even without trace norm regularization. This can be explained by the natural robustness of the CNN classifier; if the amount of label

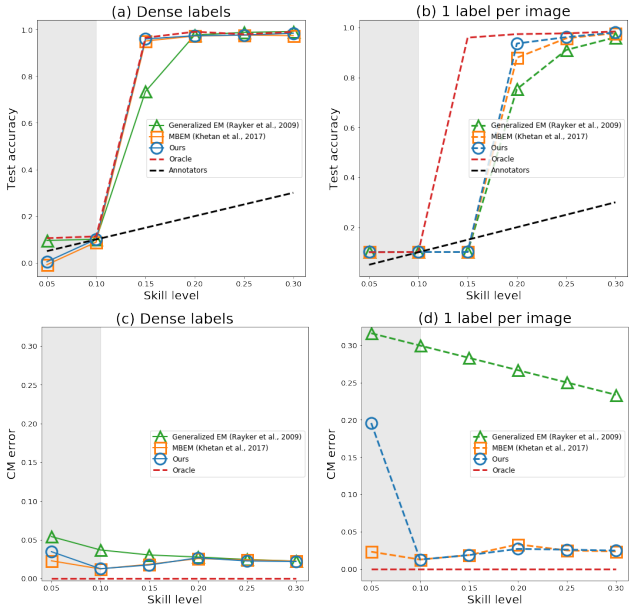


Figure 2: Comparison between our method, generalized EM, MBEM trained on noisy labels on MNIST from "hammer-spammers" for a range of mean skill level p. (a), (b) show classification accuracy in two cases, one where all annotators label each example and the other where only one label is available per example. (c), (d) quantify the CM recovery error. The shaded areas represent the cases where the average CM over the annotators are not diagonally dominant.

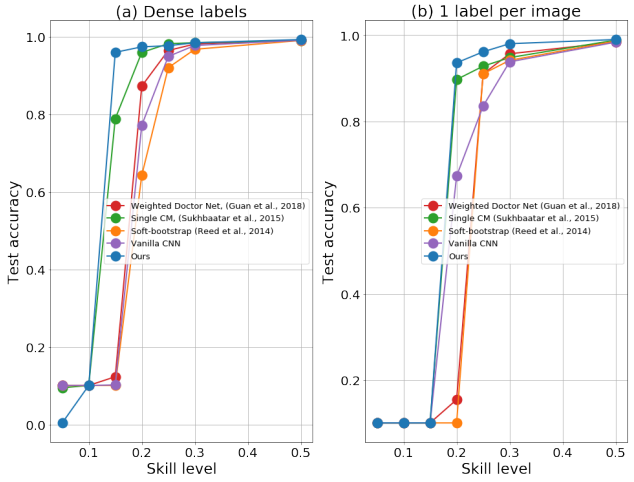


Figure 3: Classification accuracy on MNIST of different noise-robust models as a function of the mean annotator skill level p in two cases. Here, for each mean skill-level p, a group of 5 "pairwise flippers" is formed and used to generate labels. (a). each example receives labels from all the annotators. (b). each example is labelled by only 1 randomly selected annotator.

noise is sufficiently small, the base classifier is still capable of learning the true label distribution well. This, in turn, allows the model to separate annotation noise from true label distribution, improving the quality of CM estimation and thus the overall performance. However, in the presence of large label noise, having trace-norm regularization shows evident benefits.

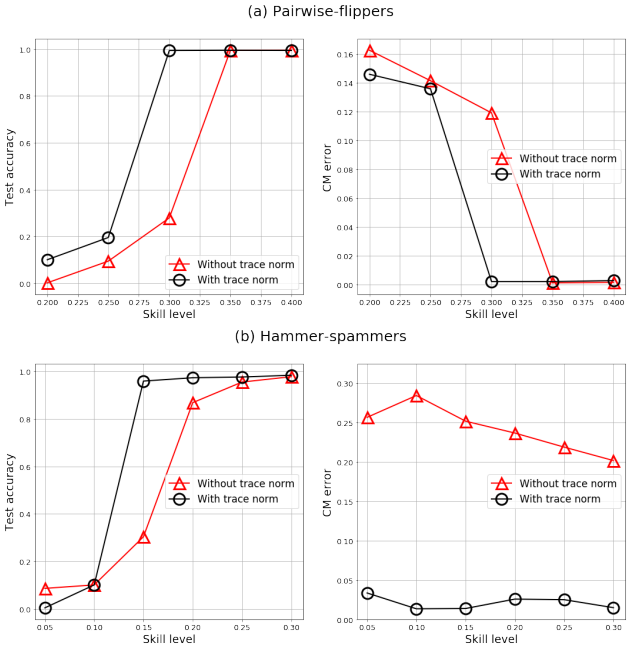


Figure 4: Comparison between our method with and without trace norm on MNIST. Results for two annotator groups, consisting of "hammer-spammers" and "pairwise-flippers" are shown for a range of mean skill level p.

E. Pseudo-codes of our method, generalized EM and MBEM

Here we provide pseudo-codes of our method (Algorithm 1), generalized EM [5] (Algorithm 2) and model-bootstrapped EM [4] (Algorithm 3) to clarify the differences between different methods for jointly learning the true label distribution and confusion matrices of annotators in eq. 2 in the main text. Given the training set $\mathcal{D} = \{\mathbf{x}_n, \tilde{y}_n^{(1)}, ..., \tilde{y}_n^{(R)}\}_{n=1}^N$, each example may not be labelled by all the annotators. In such cases, for ease of notation, we assign pseudo class $\tilde{y}_n^{(r)} = -1$ to fill the missing labels. The comparison between these three algorithms illustrates the implementational simplicity of our method, despite the comparable or superior performance demonstrated on all three datasets.

Algorithm 1 Our method

Inputs: $\mathcal{D} = \{\mathbf{x}_n, \tilde{y}_n^{(1)}, ..., \tilde{y}_n^{(R)}\}_{n=1}^N$, λ : scale of trace regularizer
Initialize the confusion matrices $\{\hat{\mathbf{A}}^{(r)}\}_{r=1}^R$ to identity matrices
Initialize the parameters of the base classifier θ
Learn θ and $\{\hat{\mathbf{A}}^{(r)}\}_{r=1}^R$ by performing minibatch SGD on the combined loss:

$$\theta, \{\hat{\mathbf{A}}^{(r)}\}_{r=1}^R \longleftarrow \text{argmin}_{\theta, \{\hat{\mathbf{A}}^{(r)}\}} \Big[\sum_{i=1}^N \sum_{r=1}^R \mathbb{1}(\tilde{y}_i^{(r)} \neq -1) \cdot \text{CE}(\hat{\mathbf{A}}^{(r)} \hat{\mathbf{p}}_\theta(\mathbf{x}_i), \tilde{y}_i^{(r)}) + \lambda \sum_{r=1}^R \text{tr}(\hat{\mathbf{A}}^{(r)}) \Big]$$

Return: $\hat{\mathbf{p}}_\theta$ and $\{\hat{\mathbf{A}}^{(r)}\}_{r=1}^R$

Algorithm 2 Generalized EM [5]

Inputs: $\mathcal{D} = \{\mathbf{x}_n, \tilde{y}_n^{(1)}, ..., \tilde{y}_n^{(R)}\}_{n=1}^N$, T : # EM steps, G : # SGD in each M-step
Initialize posterior distribution by the mean labels: for $j = 1, ..., L, n = 1, ..., N$

$$q_{nj}^{(0)} := p(y_n = j | \mathbf{x}_n, \{\tilde{y}_n^{(r)}\}_r, \theta^{(0)}) \longleftarrow R^{-1} \sum_{r=1}^R \mathbb{1}(\tilde{y}_i^{(r)} = j)$$

Initialize the parameters of the base classifier θ
Repeat T times:
M-step for θ. Learn the base classifier $\hat{\mathbf{p}}_\theta$ by performing minibatch SGD for G iterations

$$\theta^{(t+1)} \longleftarrow \text{argmin}_\theta \Big[- \sum_{n=1}^N \sum_{l=1}^L q_{nj}^{(t)} \cdot \log p(y_n = l | \mathbf{x}_n, \theta) \Big]$$

M-step for $\{\hat{\mathbf{A}}^{(r)}\}_{r=1}^R$. Estimate the confusion matrices

$$\hat{a}_{ji}^{(r),t+1} \longleftarrow \frac{\sum_{n=1}^N \mathbb{1}(\tilde{y}_i^{(r)} \neq -1) \cdot \mathbb{1}(\tilde{y}_n^{(r)} = i) \cdot q_{nj}^{(t)}}{\sum_{n=1}^N \mathbb{1}(\tilde{y}_i^{(r)} \neq -1) \cdot q_{nj}^{(t)}}$$

E-step. Estimate the posterior label distribution

$$q_{nj}^{(t+1)} \longleftarrow \frac{p(y_n = j | \mathbf{x}_n, \theta^{(t+1)}) \cdot \prod_{r=1}^R \left(\hat{a}_{j\tilde{y}_n^{(r)}}^{(r),t+1}\right)^{\mathbb{1}(\tilde{y}_i^{(r)} \neq -1)}}{\sum_{l=1}^L p(y_n = l | \mathbf{x}_n, \theta^{(t+1)}) \cdot \prod_{r=1}^R \left(\hat{a}_{l\tilde{y}_n^{(r)}}^{(r),t+1}\right)^{\mathbb{1}(\tilde{y}_i^{(r)} \neq -1)}}$$

Return: $\hat{\mathbf{p}}_{\theta^{(T)}}$ and $\{\hat{\mathbf{A}}^{(r),T}\}_{r=1}^R$

Algorithm 3 Model-Bootstrapped EM [4]

Inputs: $\mathcal{D} = \{\mathbf{x}_n, \tilde{y}_n^{(1)}, ..., \tilde{y}_n^{(R)}\}_{n=1}^N$, T : # EM steps, G : # SGD in each M-step
Initialize posterior distribution by the mean labels: for $j = 1, ..., L, n = 1, ..., N$

$$q_{nj}^{(0)} := p(y_n = j | \mathbf{x}_n, \{\tilde{y}_n^{(r)}\}_r, \theta^{(0)}) \longleftarrow R^{-1} \sum_{r=1}^R \mathbb{1}(\tilde{y}_i^{(r)} = j)$$

Initialize the parameters of the base classifier θ
Repeat T times:
M-step for θ. Learn the base classifier $\hat{\mathbf{p}}_\theta$ by performing minibatch SGD for G iterations

$$\theta^{(t+1)} \longleftarrow \text{argmin}_\theta \Big[- \sum_{n=1}^N \sum_{l=1}^L q_{nj}^{(t)} \cdot \log p(y_n = l | \mathbf{x}_n, \theta) \Big]$$

Predict on training examples. for $n = 1, ..., N$:

$$c_n \longleftarrow \text{argmax}_{l \in \{1,...,L\}} p(y_n = l | \mathbf{x}_n, \theta^{(t+1)})$$

M-step for $\{\hat{\mathbf{A}}^{(r)}\}_{r=1}^R$. Estimate the confusion matrices. For $i, j = 1, ..., L$ and $r = 1, ..., R$:

$$\hat{a}_{ji}^{(r),t+1} \longleftarrow \frac{\sum_{n=1}^N \mathbb{1}(\tilde{y}_i^{(r)} \neq -1) \cdot \mathbb{1}(\tilde{y}_n^{(r)} = i) \cdot \mathbb{1}(c_n = j)}{\sum_{n=1}^N \mathbb{1}(\tilde{y}_i^{(r)} \neq -1) \cdot \mathbb{1}(c_n = j)}$$

Update prior label distribution. for $l = 1, ..., L$:

$$p_l \longleftarrow N^{-1} \sum_{n=1}^N \mathbb{1}(c_n = l)$$

E-step. Estimate the posterior label distribution

$$q_{nj}^{(t+1)} \longleftarrow \frac{p_j \cdot \prod_{r=1}^R \left(\hat{a}_{j\tilde{y}_n^{(r)}}^{(r),t+1}\right)^{\mathbb{1}(\tilde{y}_i^{(r)} \neq -1)}}{\sum_{l=1}^L p_l \cdot \prod_{r=1}^R \left(\hat{a}_{l\tilde{y}_n^{(r)}}^{(r),t+1}\right)^{\mathbb{1}(\tilde{y}_i^{(r)} \neq -1)}}$$

Return: $\hat{\mathbf{p}}_{\theta^{(T)}}$ and $\{\hat{\mathbf{A}}^{(r),T}\}_{r=1}^R$

References

[1] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[2] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. 2009.

[3] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, 2014.

[4] Ashish Khetan, Zachary C Lipton, and Anima Anandkumar. Learning from noisy singly-labeled data. In *International Conference on Learning Representations*, 2018.

[5] Vikas C Raykar, Shipeng Yu, Linda H Zhao, Anna Jerebko, Charles Florin, Gerardo Hermosillo Valadez, Luca Bogoni, and Linda Moy. Supervised learning from multiple experts: whom to trust when everyone lies a bit. In *Proceedings of the 26th Annual international conference on machine learning*, pages 889–896. ACM, 2009.

[6] Furong Huang, Jordan T. Ash, John Langford, and Robert E. Schapire. Learning deep resnet blocks sequentially using boosting theory. *CoRR*, 2017.

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[8] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for simplicity: The all convolutional net. *CoRR*, 2015.