LILAW: LIGHTWEIGHT LEARNABLE ADAPTIVE WEIGHTING FROM CONFIDENCE AND DISAGREEMENT

Anonymous authors
Paper under double-blind review

ABSTRACT

High-loss training examples may be mislabeled or simply difficult to classify, making loss alone an unreliable guide to sample weighting under label noise. We introduce Lightweight Learnable Adaptive Weighting (LILAW), which assigns sample weights using two values: the model's confidence in the observed label and its confidence in its preferred prediction. Three global parameters control these weights and are learned by evaluating the effect of a weighted classifier update on a held-out batch, without requiring clean labels. We characterize the resulting weights and identify the conditions under which the ratio of the two probabilities distinguishes correct and incorrect annotations. Across CIFAR-10N and CIFAR-100N, LILAW improves test accuracy in nearly all comparisons involving four frozen pretrained encoders. We also examine selective classification: abstaining on less confident predictions yields higher accuracy at lower coverage. With full-network training on Clothing1M, LILAW achieves consistently improved results across multiple seeds, while being very lightweight. These findings suggest that a small set of learned weighting parameters can improve learning from real-world label noise and support selective classification.

1 INTRODUCTION

Datasets often contain incorrectly labeled samples, particularly when labels are collected through crowdsourcing or web metadata [37, 39]. These errors affect what a model learns from the available data. A mislabeled sample can encourage models to predict the wrong class, while a difficult but correctly labeled sample may provide information that the model has not yet learned. A model's predictions and confidence change during training, as training dynamics studies show [35, 32]. Adjusting loss weights of individual samples can help models make better use of noisy training data.

Several methods address label noise by modifying the loss, selecting samples, or learning their weights [42, 9, 28, 31]. Many use the training loss to estimate which samples are difficult or unreliable. However, a high loss can arise when the model is uncertain across several classes or when it confidently predicts a different class from the observed label. Cross-entropy (CE) depends only on the probability assigned to that label. It does not describe how strongly the model favors another class. Using the maximum class probability as well helps distinguish these cases.

We introduce **Li**ghtweight **L**earnable **A**daptive **W**eighting (**LILAW**), which uses these two probabilities to adjust the loss weight of each sample. Three global parameters, α, β, and δ, control a smooth weighting function. The parameters are shared across the dataset, but the weight of each sample depends on its current prediction. We also include the ratio of the observed-class probability to the maximum probability. This ratio reduces the influence of labels contradicted by the model.

We learn the parameters using training and validation batches that retain their observed, potentially noisy labels. For each training batch, we make an update of a copied classifier head, evaluate it on a validation batch, and adjust the three parameters to reduce the validation loss. The classifier is trained with the weighted CE loss. This procedure adapts the weights throughout training without an auxiliary weighting network or a clean set for learning the weights.

Selective classification provides a way to avoid returning every prediction when some are unreliable [5, 6]. A model can return its more confident predictions and leave the remaining cases for review. The aim is to increase accuracy on accepted samples by reducing coverage, i.e. the fraction of predictions

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

returned. After training with LILAW, we rank predictions by maximum class probability and evaluate this trade-off. LILAW adjusts loss weights during training and confidence-based selection allows uncertain predictions to be withheld without training another model.

We evaluate LILAW on five CIFAR-10N annotation sets and CIFAR-100N with four frozen pretrained ResNet and Vision Transformer (ViT) encoders and on Clothing1M with an ImageNet-pretrained ResNet-50. We also study selective classification and examine how many errors occur among the rejected samples when using LILAW. Our contributions are as follows:

- We propose a sample-weighting method that combines model confidence and agreement with the observed label using three learned parameters.
- We identify the conditions in which relative class probabilities can distinguish between correct and incorrect annotations.
- We report CIFAR-10N, CIFAR-100N, and Clothing1M results, and perform selective classification to show the accuracy and coverage of accepted predictions.

2 Related Work

Learning with noisy labels. Robust loss functions reduce the effect of incorrect training targets. Generalized cross-entropy [42] combines properties of CE and mean absolute error, while symmetric cross-entropy [36] combines CE with reverse CE. Forward and backward loss correction [24] use a matrix describing transitions between true and observed labels. LILAW instead weights ordinary CE using the current model's predicted probabilities.

Sample-selection methods try to identify useful training data as the model learns. Co-teaching [9] trains two networks, each using small-loss samples selected by the other. Gui et al. [7] analyze this small-loss criterion and propose a refined selection rule. DivideMix [16] estimates clean and noisy subsets and uses them in semi-supervised learning. Early-learning regularization [18] uses previous predictions to reduce memorization of noisy labels. LILAW keeps the observed training targets and assigns continuous weights to their losses. It does not maintain a second predictive network or a history of per-sample predictions.

Sample weighting. Focal loss [17] emphasizes low observed-class probabilities, while MentorNet [13] learns a curriculum. Learning to Reweight [28] evaluates how sample weights affect a training update; Meta-Weight-Net [31] learns a loss-to-weight function. Subsequent work chooses difficult examples for the validation set [12] and selects pivotal examples whose labels are then cleaned for meta reweighting [38]. LILAW also uses a validation-guided update, but learns only three parameters and permits noisy validation labels. Samples with the same CE loss can receive different weights because we use both observed-class and maximum probabilities.

Weighting priorities need not remain fixed. Zhou and Wu [44] relate suitable priorities to the distribution of example difficulty, while Xu et al. [40] analyze how importance weighting affects optimization and generalization. RHO-LOSS [22] selects examples by their estimated reduction in generalization loss. LILAW adapts a smooth weighting function with three parameters.

Sample difficulty and label quality. Variance of Gradients [1] measures changes in input gradients across checkpoints; GraNd and EL2N [25] use early gradient and prediction-error norms for data pruning. Second-split forgetting [20] helps distinguish rare and mislabeled examples. Data-IQ [29] characterizes tabular-data subgroups through confidence and uncertainty, and Seedat et al. [30] compare measures of different forms of sample hardness. These studies motivate separating uncertainty from label disagreement. LILAW uses the current prediction to determine each weight, without storing sample histories. Pretrained features also affect robustness to label noise [4]. Our CIFAR-N experiments compare CE and LILAW on frozen features. Clothing1M uses full-network training.

Selective classification. Selective classification withholds predictions to improve accuracy on those returned [5, 6]. Thulasidasan et al. [34] use abstention during training to address label noise. Rabanser et al. [26] select predictions using disagreement among training checkpoints. We apply selection after LILAW training using the final model's maximum softmax probability [11].

3 METHOD

Let:

$$\mathcal{D}_t = \{(x_i, \tilde{y}_i)\}_{i=1}^{N_t}, \qquad \mathcal{D}_v = \{(x_j^v, \tilde{y}_j^v)\}_{j=1}^{N_v}$$

be disjoint training and validation sets containing N_t and N_v samples, respectively. Each pair consists of an input and an observed label in $\{1, \ldots, K\}$. The observed labels in either set may be incorrect. We denote the true label of x_i by y_i, which is unavailable when learning the sample weights. The classifier f_θ, with parameters θ, assigns a score to each class. Applying softmax gives the class-probability vector:

$$s_i = \mathrm{softmax}(f_\theta(x_i)).$$

We first explain how these probabilities determine each sample's weight, then describe how we learn the three parameters that control the weighting function.

3.1 CONFIDENCE AND AGREEMENT WITH THE OBSERVED LABEL

We use the probability of the observed class and the largest class probability, respectively:

$$p_i = s_i[\tilde{y}_i], \qquad q_i = \max_k s_i[k]. \tag{1}$$

Their relationship describes the model's prediction relative to the label:

- If $p_i = q_i$, the model assigns its highest predicted probability to the observed label. A high q_i indicates confident agreement; a low q_i indicates an uncertain prediction.
- If $p_i < q_i$, the model favors another class. A large difference between the two probabilities indicates stronger disagreement with the observed label.

These cases describe the difficulty of the observed label for the current model and do not require knowing whether that label is correct. For example, consider two samples with $p = 0.2$. Both have CE loss $-\log(0.2)$. If the first has $q = 0.25$, the model is uncertain; if the second has $q = 0.7$, it strongly favors another class. We compare these probabilities using their ratio,

$$r_i = \frac{p_i}{q_i}, \qquad 0 < r_i \leq 1. \tag{2}$$

The two samples have ratios 0.8 and approximately 0.286, respectively. A smaller ratio indicates stronger disagreement with the observed label: the model assigns it much less probability than its preferred class. Thus, the ratio distinguishes these samples even though they have the same CE loss.

3.2 THE WEIGHTING FUNCTION

We use three global learnable parameters, α, β, and δ, to adjust each sample's contribution to the loss. These parameters control three weighting components:

$$W_\alpha(p_i, q_i) = \sigma(\alpha p_i - q_i), \tag{3}$$

$$W_\beta(p_i, q_i) = \sigma(q_i - \beta p_i), \tag{4}$$

$$W_\delta(p_i, q_i) = e^{-\frac{1}{2}(\delta p_i - q_i)^2}. \tag{5}$$

Here $\sigma(z) = (1 + e^{-z})^{-1}$ is the logistic sigmoid. Figure 1 illustrates the components at initialization, where $(\alpha, \beta, \delta) = (10, 2, 6)$. Appendix C shows some representative training examples for each of the three weights. Their roles can be understood as follows:

- **Agreement with the observed label.** W_α assigns large values when the model confidently predicts the observed label: $p_i = q_i$ and both probabilities are high. Such samples are easy for the current model with respect to their observed labels. More generally, for a fixed q_i, this component increases as the model assigns more probability to the observed label.
- **Disagreement with the observed label.** W_β assigns larger values when p_i is low and q_i is high, indicating that the model confidently favors another class. These samples are not necessarily difficult to classify. They may be easy or moderately difficult samples with incorrect annotations, or samples for which the model makes a confident error.

- **Ambiguity with the observed label.** W_δ is largest when $\delta p_i = q_i$ and decreases as these quantities differ. It can assign large values to uncertain predictions, where both probabilities are low. For example, at $\delta = 6$, $p_i = q_i = 0.1$ gives $W_\delta \approx 0.88$. Thus, this component may identify uncertain predictions for hard or ambiguous samples.

The components overlap, and we combine them so that each sample's weight depends on all three parameters. Learning these parameters changes how samples with different probabilities contribute to training. The initialization satisfies $\alpha > \delta > \beta > 0$; Section 3.3 describes the parameter updates and the constraints applied during training.

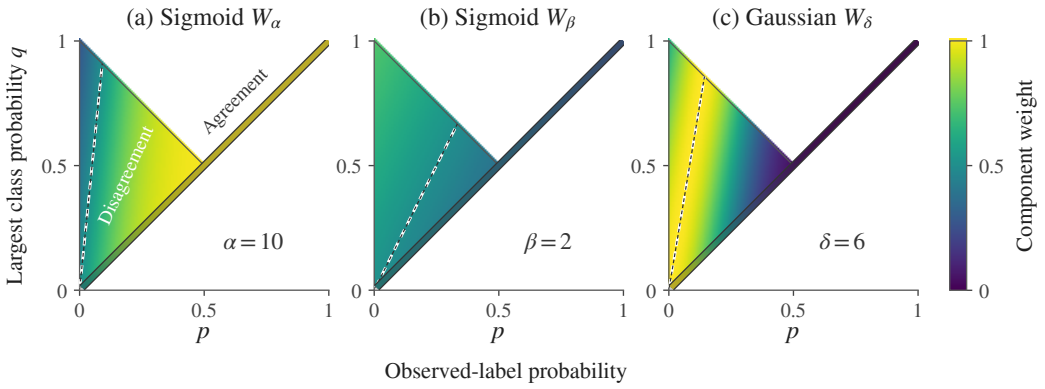

Figure 1: **LILAW weighting components.** The horizontal axis is observed-class probability p and the vertical axis is maximum class probability q. The components are shown at $(\alpha, \beta, \delta) = (10, 2, 6)$. Points on the diagonal represent agreement with the observed label; points above it represent disagreement. Colors indicate component values. The functions are displayed over the triangle $p \leq q$, including some pairs that are not attainable by a class-probability vector. The components are combined and normalized before multiplication by p/q.

We add the three components to obtain the raw LILAW weight:

$$W_i = W_\alpha(p_i, q_i) + W_\beta(p_i, q_i) + W_\delta(p_i, q_i). \tag{6}$$

Their sum is a smooth function of the predicted probabilities. Learning the three parameters changes these responses as training proceeds.

For training batch $\mathcal{B}$ of size B, we divide by the mean raw weight and then multiply by ratio $r_i = \frac{p_i}{q_i}$:

$$\bar{W} = \frac{1}{B} \sum_{j \in \mathcal{B}} W_j, \tag{7}$$

$$u_i = \frac{W_i}{\bar{W}} \frac{p_i}{q_i}. \tag{8}$$

The normalization prevents changes in the scale of W from simply rescaling the whole loss. Multiplication by p_i/q_i then reduces the contribution of strongly contradicted labels. We apply this ratio after normalization so that a batch with many such labels can have a smaller average weight.

The weighted training loss is:

$$L_{\mathcal{B}}(\theta, \alpha, \beta, \delta) = \frac{1}{B} \sum_{i \in \mathcal{B}} u_i \ \mathrm{CE}(s_i, \tilde{y}_i) \tag{9}$$

$$= \frac{1}{B} \sum_{i \in \mathcal{B}} u_i [-\log p_i]. \tag{10}$$

When updating the classifier, we treat the calculated weight u_i as a constant multiplier of CE, or another loss function. Each sample therefore contributes u_i times its ordinary CE gradient; the classifier is not updated through the weight calculation. We retain derivatives with respect to α, β, and δ for their validation-guided update. All training losses continue to use the observed label.

3.3 LEARNING THE THREE PARAMETERS

After a short warm-up using standard CE loss, we learn α, β, and δ by evaluating how a weighted training update affects validation loss. We copy the classifier head, apply one weighted update using a training batch, and evaluate unweighted CE on a separate validation batch using the updated copy. The input features remain fixed during this step. Changing the three parameters changes the training weights and, consequently, the updated head and its validation loss. We differentiate this validation loss through the head update to learn the weighting parameters.

Algorithm 1 shows the update. The copied head is used only to learn the weighting parameters and is then discarded. We update the classifier using the gradient computed before changing α, β, and δ. Their new values determine the weights for the next training batch.

Algorithm 1 Training with LILAW

Initialize the classifier and $(\alpha, \beta, \delta) = (10, 2, 6)$.
Warm up the classifier using standard CE.
For each subsequent training batch:

1. Compute the weighted training loss using the current α, β, and δ, and store the classifier gradient.
2. Copy the classifier head and apply one weighted update using fixed input features.
3. Evaluate unweighted CE on a validation batch using the updated copy. Differentiate through the head update to update α, β, and δ.
4. Enforce $\alpha \geq 1$ and $\delta \geq \beta$, then discard the copied head.
5. Update the classifier using the gradient stored in Step 1.

The copied head uses the head's current optimizer state, including momentum. We differentiate through only one update. Restricting this calculation to the head reduces its computational cost; classifier training can still optimize the full network from scratch, fine-tune a pretrained network, or train only the head. In our CIFAR-N experiments, the encoders are frozen and only the heads are trained. On Clothing1M, the classifier update trains the full network. We learn the three scalars using Adam, subject to $\alpha \geq 1$ and $\delta \geq \beta$. Appendix A provides the implementation details.

4 LABEL SEPARATION UNDER LILAW

This proposition and its proof are based on the setup and results from [7].

Proposition 1. *Let x be an input with true class y and observed label $\tilde{y}$, both in $\{1, \ldots, K\}$. Assume class-conditional label noise with transition matrix $T \in [0,1]^{K \times K}$, where, $T_{ab} = \mathbb{P}(\tilde{y} = b \mid y = a)$ is the probability that a sample from true class a receives observed label b. Under this assumption, the label corruption depends only on the true class.*

Let $s(x)[j]$ denote the probability assigned to class j by the model. Suppose the model exactly predicts the observed-label distribution for true class y:

$$s(x)[j] = T_{yj}, \qquad j = 1, \ldots, K.$$

Assume also that the diagonal entry in row y is strictly larger than every other entry in that row:

$$T_{yy} > \max_{j \neq y} T_{yj}.$$

Define the observed-class probability and the largest predicted class probability as:

$$p = s(x)[\tilde{y}], \qquad q = \max_j s(x)[j].$$

Then the probability ratio satisfies:

$$\frac{p}{q} = \begin{cases} 1, & \tilde{y} = y, \\ \dfrac{T_{y\tilde{y}}}{T_{yy}} < 1, & \tilde{y} \neq y. \end{cases} \tag{11}$$

Proof. Let y be the true class of an input x, and let $T_{yj} = \Pr(\tilde{y} = j \mid y)$ be the probability that a sample from class y has observed label j. The entries in this row are nonnegative and sum to one.

Suppose the predictor matches this observed-label distribution: $s(x)[j] = T_{yj}, j = 1, \ldots, K$. Assume also that the diagonal entry is strictly larger than every other entry in the row:

$$T_{yy} > T_{yj}, \text{ for every } j \neq y.$$

Under these assumptions, the true class y is the unique most likely predicted class. Therefore,

$$q = \max_j s(x)[j] = \max_j T_{yj} = T_{yy}.$$

Since the row sums to one, its largest entry is positive, so $q = T_{yy} > 0$ and division by q is valid. The observed-class probability is $p = s(x)[\tilde{y}] = T_{y\tilde{y}}$. We now consider the two possible cases.

If the annotation is correct, then $\tilde{y} = y$. In this case,

$$p = T_{yy} = q, \qquad \text{and hence} \qquad \frac{p}{q} = \frac{T_{yy}}{T_{yy}} = 1.$$

If the annotation is incorrect, then $\tilde{y} = j$ for some $j \neq y$. The strict diagonal condition gives:

$$p = T_{yj} < T_{yy} = q.$$

Dividing by the positive quantity T_{yy} gives a ratio strictly below one. Together, the two cases yield:

$$\frac{p}{q} = \begin{cases} 1, & \tilde{y} = y, \\ \dfrac{T_{y\tilde{y}}}{T_{yy}} < 1, & \tilde{y} \neq y. \end{cases} \tag{12}$$

Under these assumptions, correct annotations have $\frac{p}{q}$ being one, while incorrect annotations have a smaller ratio. In other words, agreement with the observed label therefore distinguishes correct and incorrect annotations. For example, if the predicted probabilities are $(0.6, 0.3, 0.1)$ and the true class is 1, the ratio, $\frac{p}{q}$, is 1 for the correct annotation and 0.5 or approximately 0.167 for the two incorrect annotations, respectively, with less probable observed labels giving smaller ratios.

5 Experiments

5.1 Datasets and Models

CIFAR-N. We use the five CIFAR-10N label sets—Aggregate, Random 1, Random 2, Random 3, and Worst—and the CIFAR-100N noisy fine labels [37]. They provide human annotations for the original CIFAR training images [14]. We add no synthetic noise. For each seed, we split the 50,000 training images into 42,500 training samples and 7,500 validation samples, stratified by observed label. The 10,000 reference-labeled test images are used only for evaluation. We extract features using ImageNet-pretrained ResNet-18 and ResNet-50 [10], CLIP ViT-B/32 [27], and DINOv2 ViT-S/14 [23]. All four encoders are frozen. We standardize the features using the training split and train a linear classifier on them. The CE and LiLAW runs use the same features, split, head initialization, and training-data order for each seed.

Clothing1M. Clothing1M contains approximately one million clothing images with noisy labels across 14 classes [39]. We initialize a ResNet-50 [10] with ImageNet-pretrained weights and train all layers. We reserve 1% of the noisy training set for learning the weighting parameters and use the remaining to train the classifier. Appendix A provides the training and preprocessing settings.

For CIFAR-N, checkpoints are selected by highest observed-label validation accuracy. LiLAW uses this same validation set to learn its weighting parameters. For Clothing1M, a noisy training holdout is used for weight adaptation, while the official clean-validation set is used for checkpoint selection. Appendix A gives the optimizers, warm-up, preprocessing, and other training settings. We use five seeds and report means and sample standard deviations. Paired gains are calculated by subtracting CE accuracy from LiLAW accuracy within each seed before averaging. The comparison between CE and LiLAW uses 100% coverage. We additionally report LiLAW at 50, 60, 70, 80, 90, and 95% coverage. Appendix B gives all selective classification results.

5.2 CIFAR-N RESULTS

Table 1 reports full-test accuracy for all six label sets and four encoders. LiLAW improves mean accuracy in 22 of the 24 comparisons, with minor accuracy loss in the other 2 comparisons. The gains are consistent across encoders on CIFAR-100N and CIFAR-10N Worst, including every one of the five seeds for each encoder on both datasets.

On CIFAR-100N, the paired gains range from 1.27 to 2.45 percentage points. With DINOv2, accuracy increases from $76.11 \pm 0.32\%$ to $78.56 \pm 0.26\%$. ResNet-50 improves from $59.78 \pm 0.37\%$ to $61.93 \pm 0.13\%$. Other gains are 1.27 points for ResNet-18 and 1.47 for CLIP.

On CIFAR-10N Worst, the gains are 1.42 points for ResNet-18, 1.07 for ResNet-50, 0.78 for CLIP, and 1.28 for DINOv2. CLIP and DINOv2 also improve on each of the other CIFAR-10N annotation sets. The two decreases occur with ResNet-50: 0.54 ± 0.28 points on Aggregate and 0.35 ± 0.20 points on Random 1. The change for ResNet-18 on Aggregate is small, at 0.02 ± 0.13 points.

Table 1: **Accuracy at 100% coverage.** Mean ± sample SD over five seeds (%). Gain is the paired LiLAW minus CE difference (percentage points). All encoders are frozen: R18/R50 denote ResNet-18/50, CLIP denotes ViT-B/32, and DINOv2 denotes ViT-S/14.

Encoder	CE	LiLAW	Gain
CIFAR-10N Aggregate (9.03% noise)			
R18	85.12±0.19	85.15±0.12	0.02±0.13
R50	88.18±0.22	87.64±0.12	−0.54±0.28
CLIP	92.62±0.10	93.25±0.17	0.62±0.25
DINOv2	96.02±0.19	97.01±0.06	0.99±0.14
CIFAR-10N Random 2 (18.12% noise)			
R18	84.65±0.19	84.92±0.06	0.27±0.19
R50	87.06±0.20	87.51±0.35	0.45±0.37
CLIP	92.27±0.23	92.86±0.40	0.59±0.38
DINOv2	95.85±0.09	96.75±0.03	0.90±0.11
CIFAR-10N Worst (40.21% noise)			
R18	81.02±0.25	82.44±0.29	1.42±0.36
R50	83.41±0.22	84.48±0.27	1.07±0.45
CLIP	89.05±0.19	89.84±0.25	0.78±0.26
DINOv2	92.11±0.32	93.39±0.46	1.28±0.51

Encoder	CE	LiLAW	Gain
CIFAR-10N Random 1 (17.23%)			
R18	84.61±0.19	84.82±0.17	0.21±0.21
R50	87.73±0.17	87.37±0.15	−0.35±0.20
CLIP	92.39±0.22	93.15±0.23	0.76±0.17
DINOv2	96.03±0.27	96.94±0.08	0.91±0.24
CIFAR-10N Random 3 (17.64%)			
R18	84.41±0.28	84.83±0.10	0.43±0.20
R50	87.39±0.28	87.56±0.17	0.17±0.37
CLIP	92.23±0.06	92.97±0.11	0.74±0.12
DINOv2	95.86±0.11	96.91±0.06	1.05±0.09
CIFAR-100N (40.20%)			
R18	57.74±0.31	59.00±0.19	1.27±0.41
R50	59.78±0.37	61.93±0.13	2.15±0.42
CLIP	67.20±0.20	68.67±0.20	1.47±0.22
DINOv2	76.11±0.32	78.56±0.26	2.45±0.10

5.3 CLOTHING1M RESULTS

The implementation reserves 1% of the noisy training pool for learning the weighting parameters. It does not use the official clean-training subset. The official clean-validation set is used to select the checkpoint. Thus, noisy labels guide the weighting update, while clean validation labels guide model selection. Training uses SGD with momentum 0.9, learning rate 0.002, weight decay 0.001, and batch size 32; the complete schedule is given in Appendix A. The 69.21% CE reference is 3.29 percentage points below the LiLAW mean. Note that LiLAW outperforms all the listed baselines in Table 2 even those with higher time complexity and space complexity. Although there exist other baselines with slightly higher accuracies, most or all of them require a significantly higher time complexity and/or space complexity compared to LiLAW.

Computational cost. LiLAW adds three learnable scalar parameters. Learning them requires an update of a copied classifier head and evaluation of the validation loss. For batch size B, feature dimension d, and K classes, each linear-head forward or backward pass costs $O(BdK)$. With comparable training and validation batch sizes, LiLAW requires a fixed number of additional head passes, so head training retains this asymptotic cost per batch. During full-network training, whether from scratch or by fine-tuning, the validation batch also requires a backbone forward pass without computing backbone gradients. Additional memory is needed for the copied head and the computational graph used to differentiate through its update. These introduce constant-factor overhead while preserving the asymptotic time and space complexity of CE training.

Table 2: **Clothing1M accuracy and computational requirements.** LILAW accuracy is mean ± SD over five seeds. Baseline accuracies come from the original sources and LILAW uses the same model. Time and memory are asymptotic costs rather than exact measurements.

Method	Accuracy (%)	Time complexity	Space complexity
Cross-entropy	69.21	$\mathcal{O}(ENP)$	$\mathcal{O}(M)$
APNL [19]	69.35	$\mathcal{O}(ENP)$	$\mathcal{O}(M)$
GCE [42]	69.75	$\mathcal{O}(ENP)$	$\mathcal{O}(M)$
Mixup [41]	70.48	$\mathcal{O}(ENP)$	$\mathcal{O}(M)$
SCE [36]	71.02	$\mathcal{O}(ENP)$	$\mathcal{O}(M)$
Weakly supervised [45]	71.36	$\mathcal{O}(ENP)$	$\mathcal{O}(M)$
Backward [24]	69.13	$\mathcal{O}(ENP + ENC^2 + C^3)$	$\mathcal{O}(M + C^2)$
Forward [24]	69.84	$\mathcal{O}(ENP + ENC^2)$	$\mathcal{O}(M + C^2)$
Co-teaching [9]	70.15	$\mathcal{O}(ENP + EN \log B)$	$\mathcal{O}(M + B)$
Dyn-Mixup [2]	70.33	$\mathcal{O}(ENP + EN)$	$\mathcal{O}(M + N)$
EG [21]	70.52	$\mathcal{O}(ENP)$	$\mathcal{O}(M + N)$
CIW [15]	70.52	$\mathcal{O}(ENP)$	$\mathcal{O}(M + B)$
SEAL [3]	70.63	$\mathcal{O}(rENP)$	$\mathcal{O}(M + NC)$
CICW [15]	70.84	$\mathcal{O}(ENP + ENC)$	$\mathcal{O}(M + BC)$
LRT / AdaCorr [43]	71.74	$\mathcal{O}(ENP + qNC)$	$\mathcal{O}(M + NC)$
Joint-Optim [33]	72.16	$\mathcal{O}(ENP + ENC)$	$\mathcal{O}(M + NC)$
LILAW (ours)	$\mathbf{72.50 \pm 0.06}$	$\mathcal{O}(ENP)$	$\mathcal{O}(M)$

E: epochs; N: examples per epoch; P: per-example classifier work; M: classifier working memory; B: batch size; C: classes; r: self-evolution rounds; q: correction passes.

5.4 SELECTIVE CLASSIFICATION

At test time, we predict the class with the largest probability and use that probability, q, as confidence. To evaluate a coverage of c, we sort the N test samples by decreasing confidence and retain the first $k = \lfloor cN \rfloor$. Coverage is the fraction of predictions returned. Selective accuracy is the proportion of those predictions that are correct:

$$\mathrm{Acc}(c) = \frac{1}{k} \sum_{i=1}^{k} \mathbf{1}\{\hat{y}_{(i)} = y_{(i)}\}. \tag{13}$$

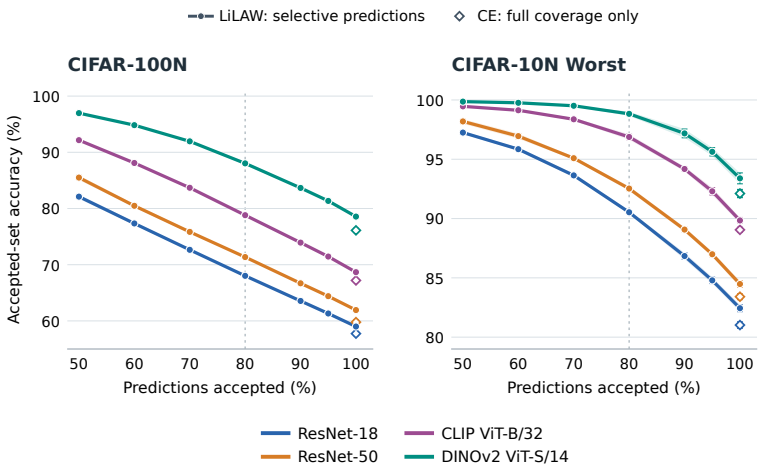

Figure 2: **The accuracy-coverage trade-off of LILAW.** These curves show accuracy on selected predictions. Points, error bars, and shaded bands show means and one sample SD over five seeds. CE is shown only at 100% coverage, using outlined diamonds. The vertical line marks 80% coverage.

At 100% coverage, all predictions are returned and this equals ordinary test accuracy. At 80% coverage, the least confident 20% are withheld. Neural network confidence may not be calibrated [8]: predictions with scores near 0.8 may not be correct 80% of the time. We use confidence to rank predictions and measure the resulting accuracy. The training weights are not needed at test time. The ratio p/q requires an observed label, while the confidence q uses the prediction alone. Section 5.5 explains why rejection and mislabel detection can identify different samples. Figure 2 and Appendix B show accuracy with LILAW at each coverage.

Accuracy increases as the least confident predictions are withheld. For DINOv2 on CIFAR-100N, the full-test accuracy is $78.56 \pm 0.26\%$. Returning 80% of predictions gives $88.03 \pm 0.30\%$ accuracy, returning 70% gives $91.95 \pm 0.11\%$, and returning 50% gives $96.97 \pm 0.11\%$. On CIFAR-10N Worst, DINOv2 reaches $97.18 \pm 0.37\%$ accuracy at 90% coverage and $98.83 \pm 0.18\%$ at 80% coverage, compared with its full-test accuracy of $93.39 \pm 0.46\%$.

5.5 Rejected predictions and label noise

We can use the reported accuracies to examine the predictions that are withheld. Let a be full-test accuracy, a_c the accuracy on accepted predictions, and a_r the accuracy on rejected predictions. For realized coverage $c = k/N$ with $0 < c < 1$,

$$a = c\,a_c + (1 - c)a_r, \qquad a_r = \frac{a - c\,a_c}{1 - c}. \tag{14}$$

For DINOv2 on CIFAR-100N at 80% coverage, the reported means give approximately 40.68% accuracy on rejected predictions, compared with 88.03% on accepted predictions. The rejected 20% accounts for approximately 55.3% of all prediction errors. On CIFAR-10N Worst, rejected accuracy is approximately 71.63%, compared with 98.83% on accepted predictions, and the rejected 20% accounts for approximately 85.8% of errors. These are calculations from rounded means.

The rejected samples are therefore more likely to be misclassified. We cannot conclude that they are mislabeled: selection is evaluated on the reference-labeled test set, while the noisy annotations belong to the training data. Confidence and label disagreement can also identify different samples. A model might assign probabilities $(0.05, 0.90, 0.05)$ to a sample whose true class is 2 but whose observed label is 1. Its prediction is correct and confident, even though the annotation is wrong. Such a sample has low p/q but high q, so it can receive less weight during training and still be accepted during selective classification. Proposition 1 explains the separate conditions under which probability ratio can identify incorrect annotations.

6 Limitations

LILAW relies on the model's predicted probabilities, so confident errors can reduce the weight of correctly labeled samples. Our label-separation result assumes diagonally dominant noise, but these conditions are not required to apply the method. Clothing1M includes classes for which an incorrect annotation is more likely than the correct label [39], extending our method beyond this theoretical setting. We also show the effectiveness of our method in very high real-world noise settings. Future work could adapt LILAW's learnable weighting functions for active learning, helping allocate a limited annotation budget between labeling new samples and reviewing potentially incorrect annotations, which could be especially useful in high-stakes fields such as healthcare. It could also be useful in federated learning and continual learning to account for differences in label quality across sites or across time.

7 Conclusion

We introduced LILAW, a lightweight adaptive sample-weighting method with three global learnable parameters. It uses the observed-class and maximum class probabilities to adjust each sample's contribution to training, with parameter updates guided by held-out observed labels. LILAW improves accuracy in the CIFAR-10N, CIFAR-100N, and Clothing1M datasets which contain realistic noise. Selective classification further avoids the need to predict on low confidence samples. These results support adaptive weighting for learning from noisy labels and motivate direct evaluation of its use for identifying mislabeled training samples.

AI USE STATEMENT

In this work, we used generative AI tools to assist in the checking the correctness of mathematical proofs, provide feedback on methodology and experiments, implement pre-processing and post-processing methods, and support data analysis. Additionally, we used generative AI tools for creating figures and tables, helping debug code, analyzing literature, brainstorming, and editing for readability and flow. We have not used generative AI tools for other tasks or tasks that are not applicable to this work. We have reviewed all AI-assisted work including doing a manual literature survey and verifying our writing and code. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

A model can assign low weight to correctly labeled samples from underrepresented groups. Aggregate accuracy alone cannot determine whether such groups are adversely affected. Applications of selective classification should evaluate the rejected cases as well as the predictions returned to users.

REPRODUCIBILITY STATEMENT

Section 3 defines LILAW and its training procedure. Appendix A describes the implementation details. Appendix B gives all selective CIFAR-N results. Appendix C shows some representative training examples for each of the three weights. Full code will be released upon acceptance.

REFERENCES

[1] Chirag Agarwal, Daniel D'souza, and Sara Hooker. Estimating example difficulty using variance of gradients. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10358–10368. IEEE, 2022.

[2] Eric Arazo, Diego Ortego, Paul Albert, Noel O'Connor, and Kevin McGuinness. Unsupervised label noise modeling and loss correction. In *International conference on machine learning*, pp. 312–321. PMLR, 2019.

[3] Pengfei Chen, Junjie Ye, Guangyong Chen, Jingwei Zhao, and Pheng-Ann Heng. Beyond class-conditional assumption: A primary attempt to combat instance-dependent label noise. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pp. 11442–11450, 2021.

[4] Francesco Di Salvo, Sebastian Doerrich, Ines Rieger, and Christian Ledig. An embedding is worth a thousand noisy labels. *arXiv preprint arXiv:2408.14358*, 2024.

[5] Yonatan Geifman and Ran El-Yaniv. Selective classification for deep neural networks. *Advances in neural information processing systems*, 30, 2017.

[6] Yonatan Geifman and Ran El-Yaniv. Selectivenet: A deep neural network with an integrated reject option. In *International conference on machine learning*, pp. 2151–2159. PMLR, 2019.

[7] Xian-Jin Gui, Wei Wang, and Zhang-Hao Tian. Towards understanding deep learning from noisy labels with small-loss criterion. *arXiv preprint arXiv:2106.09291*, 2021.

[8] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *International conference on machine learning*, pp. 1321–1330. PMLR, 2017.

[9] Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama. Co-teaching: Robust training of deep neural networks with extremely noisy labels. *Advances in neural information processing systems*, 31, 2018.

[10] 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*, pp. 770–778, 2016.

[11] Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *arXiv preprint arXiv:1610.02136*, 2016.

[12] Nishant Jain, Arun S Suggala, and Pradeep Shenoy. Improving generalization via meta-learning on hard samples. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 27590–27599. IEEE, 2024.

[13] Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *International conference on machine learning*, pp. 2304–2313. PMLR, 2018.

[14] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images, 2009.

[15] Abhishek Kumar and Ehsan Amid. Constrained instance and class reweighting for robust learning under label noise. *arXiv preprint arXiv:2111.05428*, 2021.

[16] Junnan Li, Richard Socher, and Steven CH Hoi. Dividemix: Learning with noisy labels as semi-supervised learning. *arXiv preprint arXiv:2002.07394*, 2020.

[17] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pp. 2980–2988, 2017.

[18] Sheng Liu, Jonathan Niles-Weed, Narges Razavian, and Carlos Fernandez-Granda. Early-learning regularization prevents memorization of noisy labels. *Advances in neural information processing systems*, 33:20331–20342, 2020.

[19] Xingjun Ma, Hanxun Huang, Yisen Wang, Simone Romano, Sarah Erfani, and James Bailey. Normalized loss functions for deep learning with noisy labels. In *International conference on machine learning*, pp. 6543–6553. PMLR, 2020.

[20] Pratyush Maini, Saurabh Garg, Zachary Lipton, and J Zico Kolter. Characterizing datapoints via second-split forgetting. *Advances in Neural Information Processing Systems*, 35:30044–30057, 2022.

[21] Negin Majidi, Ehsan Amid, Hossein Talebi, and Manfred K Warmuth. Exponentiated gradient reweighting for robust training under label noise and beyond. *arXiv preprint arXiv:2104.01493*, 2021.

[22] Sören Mindermann, Jan M Brauner, Muhammed T Razzak, Mrinank Sharma, Andreas Kirsch, Winnie Xu, Benedikt Höltgen, Aidan N Gomez, Adrien Morisot, Sebastian Farquhar, et al. Prioritized training on points that are learnable, worth learning, and not yet learnt. In *International Conference on Machine Learning*, pp. 15630–15649. PMLR, 2022.

[23] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

[24] Giorgio Patrini, Alessandro Rozza, Aditya Krishna Menon, Richard Nock, and Lizhen Qu. Making deep neural networks robust to label noise: A loss correction approach. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1944–1952, 2017.

[25] Mansheej Paul, Surya Ganguli, and Gintare Karolina Dziugaite. Deep learning on a data diet: Finding important examples early in training. *Advances in neural information processing systems*, 34:20596–20607, 2021.

[26] Stephan Rabanser, Anvith Thudi, Kimia Hamidieh, Adam Dziedzic, Israfil Bahceci, Akram Bin Sediq, Hamza Sokun, and Nicolas Papernot. Selective prediction via training dynamics. *arXiv preprint arXiv:2205.13532*, 2022.

[27] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PmLR, 2021.

[28] Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. Learning to reweight examples for robust deep learning. In *International conference on machine learning*, pp. 4334–4343. PMLR, 2018.

[29] Nabeel Seedat, Jonathan Crabbé, Ioana Bica, and Mihaela van der Schaar. Data-iq: Characterizing subgroups with heterogeneous outcomes in tabular data, 2022.

[30] Nabeel Seedat, Fergus Imrie, and Mihaela van der Schaar. Dissecting sample hardness: A fine-grained analysis of hardness characterization methods for data-centric ai. In *International Conference on Learning Representations*, volume 2024, pp. 9495–9564, 2024.

[31] Jun Shu, Qi Xie, Lixuan Yi, Qian Zhao, Sanping Zhou, Zongben Xu, and Deyu Meng. Meta-weight-net: Learning an explicit mapping for sample weighting. *Advances in neural information processing systems*, 32, 2019.

[32] Swabha Swayamdipta, Roy Schwartz, Nicholas Lourie, Yizhong Wang, Hannaneh Hajishirzi, Noah A Smith, and Yejin Choi. Dataset cartography: Mapping and diagnosing datasets with training dynamics. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 9275–9293, 2020.

[33] Daiki Tanaka, Daiki Ikami, Toshihiko Yamasaki, and Kiyoharu Aizawa. Joint optimization framework for learning with noisy labels. In *2018 IEEE/CVF conference on computer vision and pattern recognition*, pp. 5552–5560. IEEE, 2018.

[34] Sunil Thulasidasan, Tanmoy Bhattacharya, Jeff Bilmes, Gopinath Chennupati, and Jamal Mohd-Yusof. Combating label noise in deep learning using abstention. *arXiv preprint arXiv:1905.10964*, 2019.

[35] Mariya Toneva, Alessandro Sordoni, Remi Tachet des Combes, Adam Trischler, Yoshua Bengio, and Geoffrey J Gordon. An empirical study of example forgetting during deep neural network learning, 2018.

[36] Yisen Wang, Xingjun Ma, Zaiyi Chen, Yuan Luo, Jinfeng Yi, and James Bailey. Symmetric cross entropy for robust learning with noisy labels. In *2019 IEEE/CVF international conference on computer vision (ICCV)*, pp. 322–330. IEEE, 2019.

[37] Jiaheng Wei, Zhaowei Zhu, Hao Cheng, Tongliang Liu, Gang Niu, and Yang Liu. Learning with noisy labels revisited: A study using real-world human annotations. *arXiv preprint arXiv:2110.12088*, 2021.

[38] Yinjun Wu, Adam Stein, Jacob Gardner, and Mayur Naik. Learning to select pivotal samples for meta re-weighting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 6128–6136, 2023.

[39] Tong Xiao, Tian Xia, Yi Yang, Chang Huang, and Xiaogang Wang. Learning from massive noisy labeled data for image classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2691–2699, 2015.

[40] Da Xu, Yuting Ye, and Chuanwei Ruan. Understanding the role of importance weighting for deep learning, 2021.

[41] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

[42] Zhilu Zhang and Mert Sabuncu. Generalized cross entropy loss for training deep neural networks with noisy labels. *Advances in neural information processing systems*, 31, 2018.

[43] Songzhu Zheng, Pengxiang Wu, Aman Goswami, Mayank Goswami, Dimitris Metaxas, and Chao Chen. Error-bounded correction of noisy labels. In *International conference on machine learning*, pp. 11447–11457. PMLR, 2020.

[44] Xiaoling Zhou and Ou Wu. Which samples should be learned first: Easy or hard? *IEEE Transactions on Neural Networks and Learning Systems*, 35(11):15264–15278, 2023.

[45] Bohan Zhuang, Lingqiao Liu, Yao Li, Chunhua Shen, and Ian Reid. Attend in groups: a weakly-supervised deep learning framework for learning from web data. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2915–2924. IEEE, 2017.

A IMPLEMENTATION AND TRAINING PROTOCOL

We summarize the training settings in Table 3 and the pretrained encoders in Table 4. Settings follow the implementation defaults.

Table 3: **Training settings.** Settings used to train the models.

Setting	CIFAR-N	Clothing1M
Trainable classifier	Linear head; encoder frozen	Full ResNet-50
Initialization	Pretrained features; random head	ImageNet-1K weights
Model optimizer	Adam	SGD, momentum 0.9
Learning rate	10^{-4}	0.002
Weight decay	0	0.001
Batch size	128	32
Epoch budget	At most 100	80; 1,000 batches/epoch
CE warm-up	2 epochs	1 epoch
Scalar optimizer	Adam, 0.005	Adam, 0.005
Scalar initialization	$(10, 2, 6)$	$(10, 2, 6)$
Head update	Adam	SGD
Weight-update data	15% observed-label holdout	1% noisy-training holdout
Checkpoint criterion	Observed-label accuracy	Official clean-validation accuracy

Table 4: **Frozen CIFAR-N encoders.** Features are extracted using each pretrained model's evaluation transform. Input size is the square crop size in pixels.

Encoder	Model identifier	Dimension	Input
ResNet-18	torchvision `ResNet18_Weights.IMAGENET1K_V1`	512	224
ResNet-50	torchvision `ResNet50_Weights.IMAGENET1K_V1`	2048	224
CLIP ViT-B/32	timm `vit_base_patch32_clip_224.openai`	512	224
DINOv2 ViT-S/14	timm `vit_small_patch14_dinov2.lvd142m`	384	518

Note that Algorithm 1 uses separate optimizers for the classifier and the three weighting parameters as shown in Table 3.

For CIFAR-N, we extract features once for each image and encoder, without training-time augmentation. Each method selects its checkpoint by the highest observed-label validation accuracy. Early stopping has patience of five epochs. The `ReduceLROnPlateau` scheduler also uses accuracy, patience of five, and a minimum learning rate of 10^{-5}. Test evaluation uses the restored checkpoint.

The Clothing1M implementation trains all layers of an ImageNet-initialized ResNet-50. It reserves a stratified 1% of the noisy training pool for adaptation and samples the remaining images evenly across observed classes. Training images are resized, randomly cropped to 224×224, horizontally flipped, and normalized. Evaluation uses a deterministic center crop. The learning rate is reduced by a factor of ten after 40 epochs. The official clean validation set is used to select the checkpoint.

B SELECTIVE CIFAR-N RESULTS

Table 1 in the main paper contains all 24 full-coverage comparisons. Tables 5, 6, 7, 8, 9, and 10 give LILAW accuracy at every reported lower coverage. These results use the same selected checkpoints as the full-coverage comparison.

Table 5: **CIFAR-10N Aggregate: LiLAW selective accuracy.** Mean ± sample SD over five seeds, in percent. Columns give coverage in percent. Table 1 reports full coverage.

Encoder	50%	60%	70%	80%	90%	95%
ResNet-18	98.54±0.07	97.69±0.12	95.97±0.10	93.62±0.09	89.88±0.06	87.64±0.10
ResNet-50	99.16±0.04	98.51±0.05	97.41±0.07	95.49±0.07	92.22±0.17	90.19±0.13
CLIP ViT-B/32	99.84±0.00	99.81±0.05	99.46±0.03	98.76±0.05	97.14±0.12	95.50±0.13
DINOv2 ViT-S/14	99.95±0.01	99.93±0.01	99.93±0.01	99.88±0.02	99.53±0.02	98.89±0.03

Table 6: **CIFAR-10N Random 1: LiLAW selective accuracy.** Mean ± sample SD over five seeds, in percent. Columns give coverage in percent. Table 1 reports full coverage.

Encoder	50%	60%	70%	80%	90%	95%
ResNet-18	98.44±0.06	97.56±0.06	95.96±0.12	93.31±0.07	89.52±0.15	87.34±0.17
ResNet-50	99.20±0.06	98.42±0.06	97.34±0.09	95.36±0.08	91.99±0.09	89.88±0.10
CLIP ViT-B/32	99.84±0.00	99.78±0.02	99.46±0.05	98.75±0.06	97.08±0.10	95.48±0.14
DINOv2 ViT-S/14	99.95±0.02	99.94±0.01	99.91±0.01	99.86±0.01	99.49±0.03	98.85±0.05

Table 7: **CIFAR-10N Random 2: LiLAW selective accuracy.** Mean ± sample SD over five seeds, in percent. Columns give coverage in percent. Table 1 reports full coverage.

Encoder	50%	60%	70%	80%	90%	95%
ResNet-18	98.40±0.09	97.55±0.09	95.77±0.11	93.25±0.09	89.64±0.08	87.43±0.11
ResNet-50	99.08±0.08	98.32±0.11	97.24±0.10	95.25±0.12	92.03±0.26	89.98±0.30
CLIP ViT-B/32	99.84±0.03	99.70±0.03	99.43±0.02	98.67±0.17	96.92±0.26	95.24±0.37
DINOv2 ViT-S/14	99.95±0.01	99.93±0.01	99.92±0.01	99.84±0.02	99.41±0.03	98.72±0.03

Table 8: **CIFAR-10N Random 3: LiLAW selective accuracy.** Mean ± sample SD over five seeds, in percent. Columns give coverage in percent. Table 1 reports full coverage.

Encoder	50%	60%	70%	80%	90%	95%
ResNet-18	98.33±0.04	97.43±0.07	95.74±0.04	93.11±0.17	89.50±0.15	87.32±0.09
ResNet-50	99.09±0.12	98.42±0.06	97.27±0.07	95.37±0.09	92.14±0.11	90.00±0.14
CLIP ViT-B/32	99.86±0.01	99.77±0.02	99.44±0.04	98.79±0.07	97.01±0.04	95.33±0.12
DINOv2 ViT-S/14	99.96±0.01	99.95±0.01	99.93±0.01	99.85±0.02	99.43±0.02	98.81±0.04

Table 9: **CIFAR-10N Worst: LiLAW selective accuracy.** Mean ± sample SD over five seeds, in percent. Columns give coverage in percent. Table 1 reports full coverage.

Encoder	50%	60%	70%	80%	90%	95%
ResNet-18	97.25±0.12	95.85±0.12	93.64±0.15	90.53±0.15	86.84±0.27	84.79±0.27
ResNet-50	98.19±0.20	96.95±0.22	95.09±0.12	92.53±0.19	89.07±0.20	86.99±0.21
CLIP ViT-B/32	99.46±0.04	99.13±0.05	98.37±0.12	96.88±0.23	94.18±0.20	92.30±0.31
DINOv2 ViT-S/14	99.86±0.02	99.76±0.01	99.51±0.06	98.83±0.18	97.18±0.37	95.63±0.35

Table 10: **CIFAR-100N: LiLAW selective accuracy.** Mean ± sample SD over five seeds, in percent. Columns give coverage in percent. Table 1 reports full coverage.

Encoder	50%	60%	70%	80%	90%	95%
ResNet-18	82.11 ± 0.38	77.34 ± 0.28	72.65 ± 0.17	68.02 ± 0.13	63.55 ± 0.17	61.32 ± 0.25
ResNet-50	85.50 ± 0.42	80.48 ± 0.29	75.84 ± 0.22	71.36 ± 0.15	66.69 ± 0.18	64.40 ± 0.19
CLIP ViT-B/32	92.16 ± 0.17	88.10 ± 0.07	83.69 ± 0.18	78.81 ± 0.13	73.93 ± 0.21	71.45 ± 0.19
DINOv2 ViT-S/14	96.97 ± 0.11	94.83 ± 0.15	91.95 ± 0.11	88.03 ± 0.30	83.66 ± 0.12	81.35 ± 0.16

C REPRESENTATIVE TRAINING EXAMPLES

Figures 3–5 show examples selected for large values of each component. The displayed W_α examples have correct labels and confident correct predictions. The W_β examples are also recognized correctly, although their observed labels are wrong. The W_δ examples receive low-confidence predictions, including both correct and incorrect classes. These patterns explain the descriptions easy, easy to moderate, and hard or ambiguous, respectively. Here, difficulty concerns recognizing the true class rather than fitting the observed annotation. Each tile shows the observed label, the true class, and a prediction with its maximum class probability $q = \max_j s_j$. Green predictions match the true class; red predictions do not. These examples are from using ResNet-18 with LILAW on CIFAR-100 with 20% synthetic uniform label noise.

C.1 EASY EXAMPLES: LARGE W_α

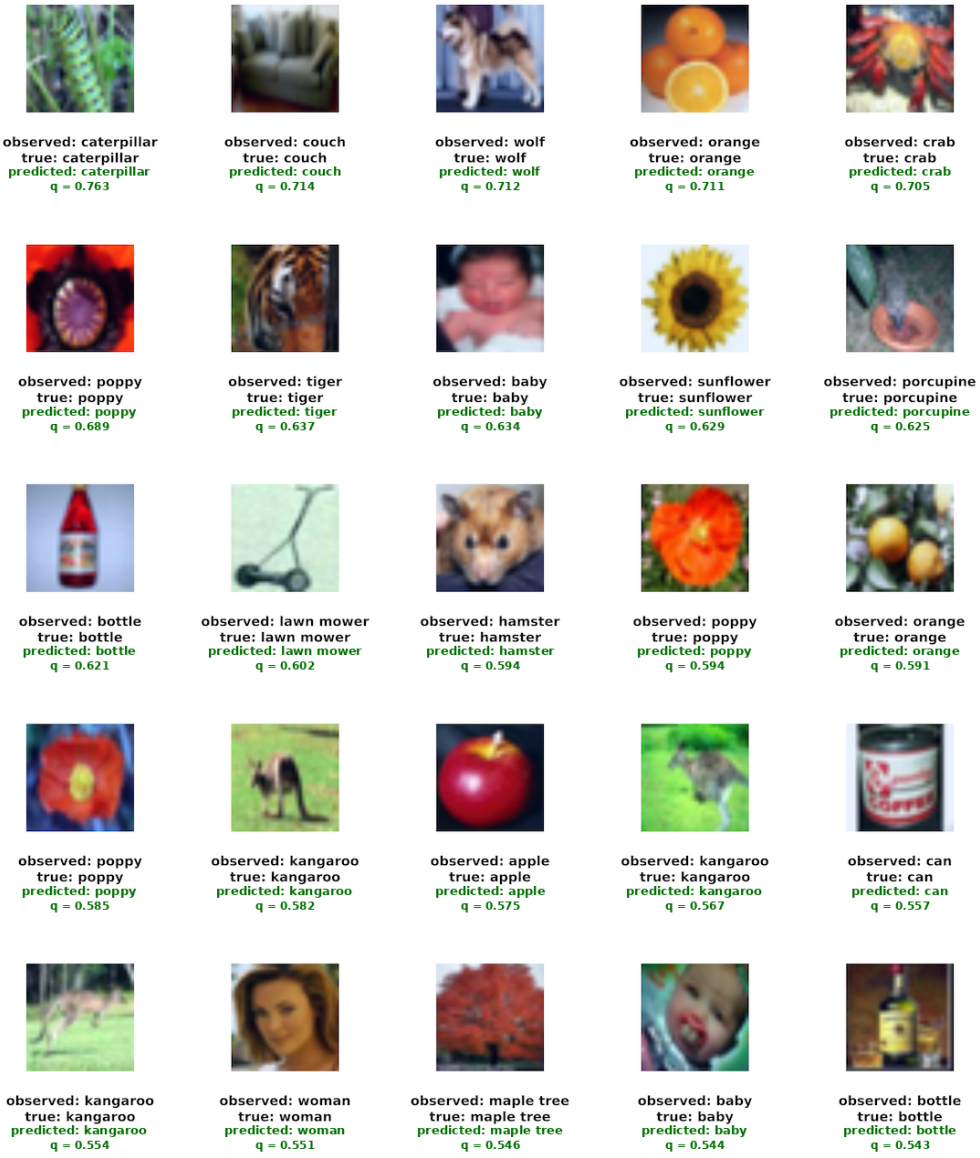

Figure 3: Examples selected for large W_α. All 25 have correct observed labels and correct predictions, with q ranging from 0.543 to 0.763.

C.2 EASY-TO-MODERATE EXAMPLES: LARGE W_β

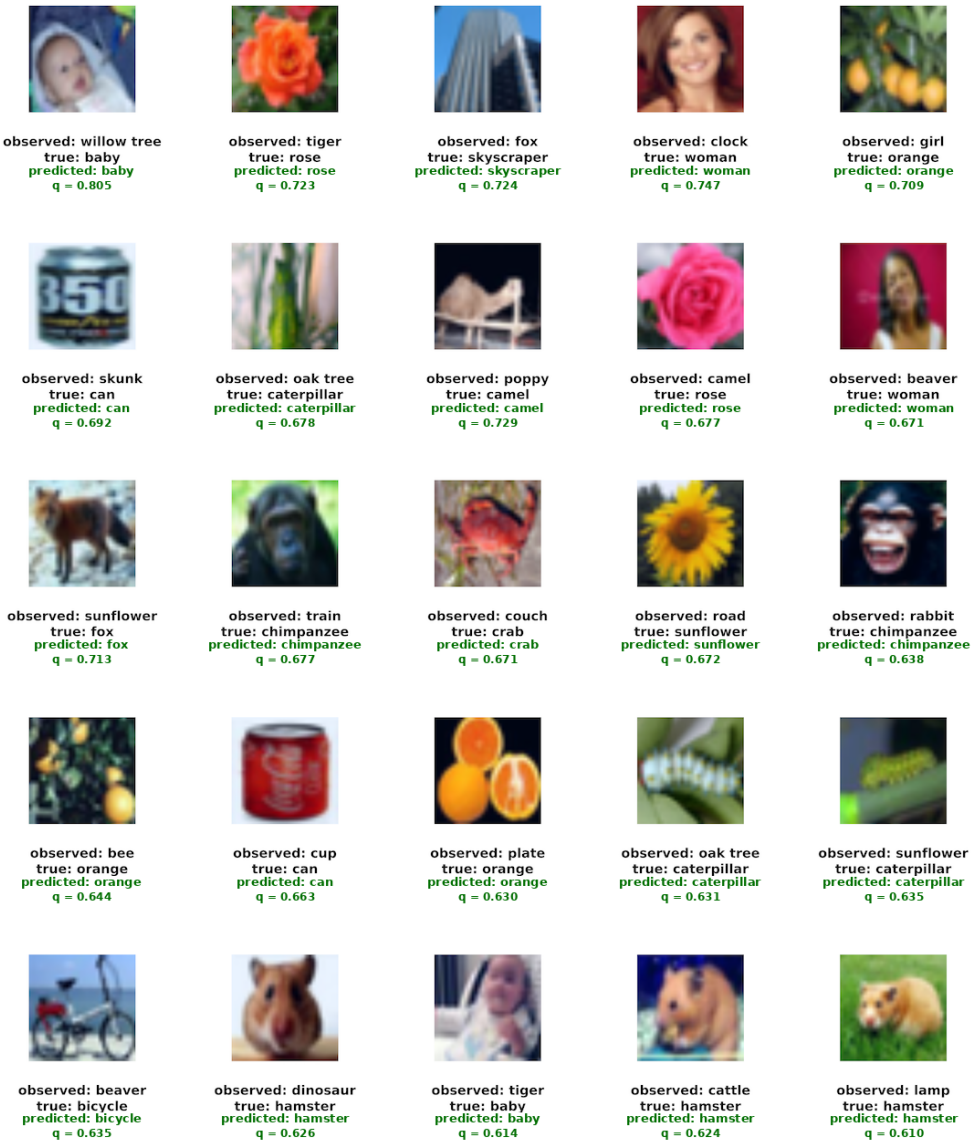

Figure 4: Examples selected for large W_β. All 25 have incorrect observed labels, but every displayed prediction recognizes the true class, with q between 0.610 and 0.805.

C.3 HARD EXAMPLES: LARGE W_δ

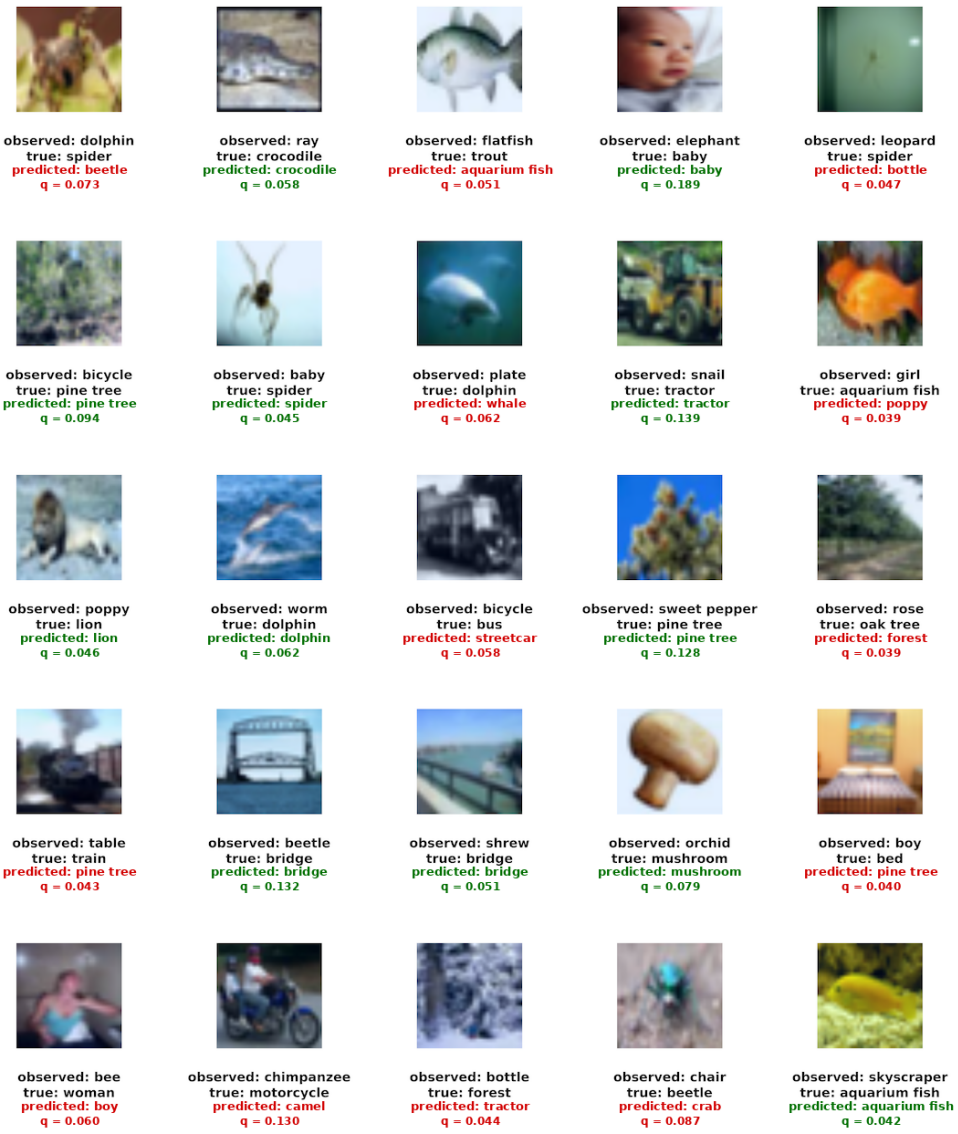

Figure 5: Examples selected for large W_δ. All 25 have incorrect observed labels, and their maximum predicted probabilities are low: q ranges from 0.039 to 0.189. The predictions include 12 correct classes and 13 errors, which are all unconfident.