Safeguarded Dynamic Label Regression for Noisy Supervision

Jiangchao Yao[†,‡], Hao Wu[†], Ya Zhang[†]
Ivor W. Tsang[‡], Jun Sun[†]
[†]Shanghai Jiao Tong University
[‡]University of Technology Sydney

November 14, 2018

Outline

Background

Low Expensive Noisy Data Meets Deep Learning

- Inexhaustible social images with annotations on websites.
- Fine-grained annotations from crowdsourcing platforms.
- Rich medical diagnosis by numerous levels of doctors.
- Large amount of unreliable stock labels for revenue.

Learning with Noisy Supervision Brings Robustness

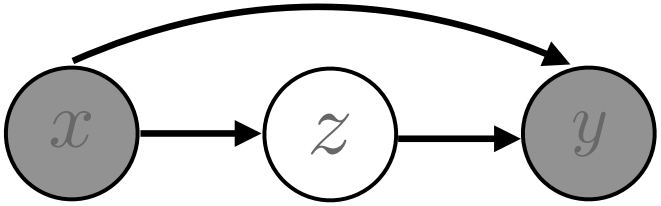
Existing deep learning based methods,

- Learning with Noise Transition
- Learning with Sample Re-weighting
- Learning with Model Regularization

Outline

Probabilistic Modeling

Learning with Noise Transition



Learning with Noise Transition

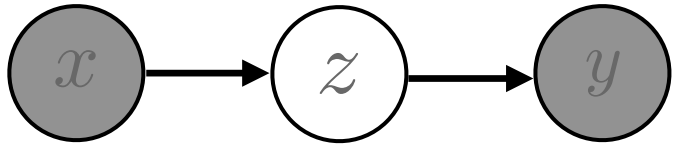
$$\ln P(y|x) = \ln \sum_{z} \underbrace{P(y|z,x)}_{\text{Noise transition}} \underbrace{P(z|x)}_{\text{Classifier}} \tag{1}$$

Classification Risk:

$$\mathbb{E}_{x,z}\left[-\ln P(z|x)\right] \neq \mathbb{E}_{x,y}\left[-\ln P(y|x)\right] \text{ if } z \not\equiv y.$$

Probabilistic Modeling

Learning with Noise Transition



Learning with Noise Transition

If it is given the **class-conditional** noise, *i.e.*,

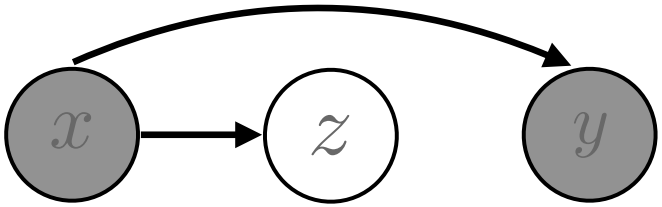
$$P(y|z,x) = P(y|z) = T_{zy} \textbf{ (given and invertible)},$$

we have the following theorem for parameter estimation,

$$\arg\min_{\theta} \mathbb{E}_{x,z}\left[-\ln P(z|x)\right] = \arg\min_{\theta} \mathbb{E}_{x,y}\left[-\ln P(y|x)\right].$$

Probabilistic Modeling

Learning with Sample Re-weighting



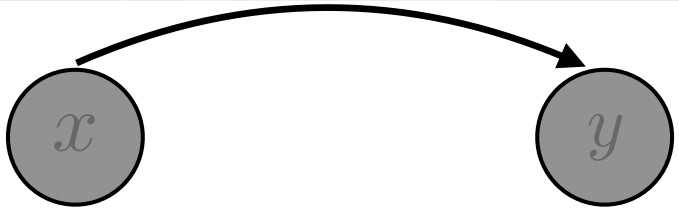
Learning with Sample Re-weighting

$$\mathbb{E}_{x,z}\left[-\ln \underbrace{P(z|x)}_{\text{classifier}}\right] = \mathbb{E}_{x,y}\left[-\underbrace{\frac{P_c(x,z)}{P_n(x,y)}\bigg|_{z=y}}_{\text{weight}} \ln \underbrace{P(y|x)}_{\text{classifier}}\right] \tag{2}$$

$$= \mathbb{E}_{x,y}\left[-\underbrace{\beta(x,y)}_{\text{weight}} \ln \underbrace{P(y|x)}_{\text{classifier}}\right]$$

Probabilistic Modeling

Learning with Sample Re-weighting



Learning with Sample Re-weighting

Typically, the weight can be estimated with small **clean data**,

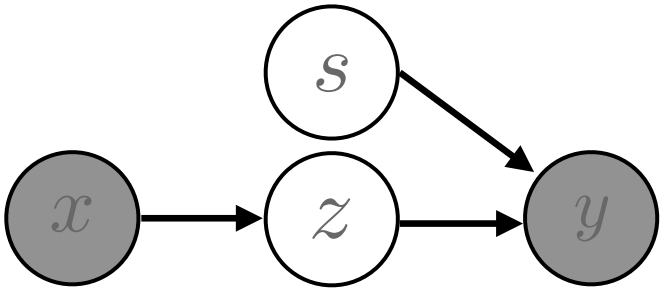
$$\beta(x, y) = \frac{P_c(x, y)}{P_n(x, y)} = \frac{P_c(y|x)}{P_n(y|x)}.$$

A simpler way is based on the classifier itself motivated by the **perceptual consistency**, i.e., bootstrapping,

$$\beta(x, y) = \alpha + (1 - \alpha)\frac{P(y|x)}{P_n(y|x)}, \text{ where } \alpha \in [0, 1].$$

Probabilistic Modeling

Learning with Model Regularization



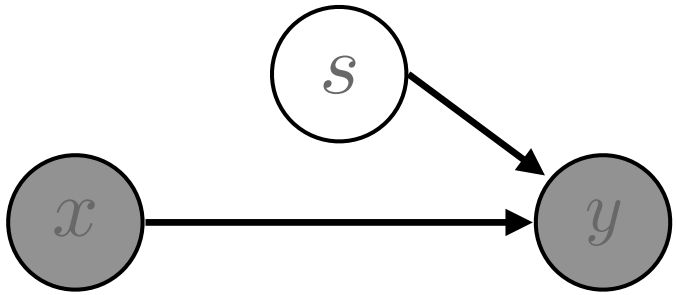
Learning with Model Regularization

$$\ln P(y|x) = \ln \sum_{z} \underbrace{P(y|z,s)}_{\text{Noise transition}} \underbrace{P(z|x)}_{\text{Classifier}} \tag{3}$$

Explicitly introduce a **noise source** to apportion the reasoning from z to y, which alleviates the **uncertain effect** on classifier

Probabilistic Modeling

Learning with Model Regularization



Learning with Model Regularization

Roughly treat $z{=}y$ and the following conjecture is quite useful, i.e., deep neural networks memorize physic patterns ♠ in order.

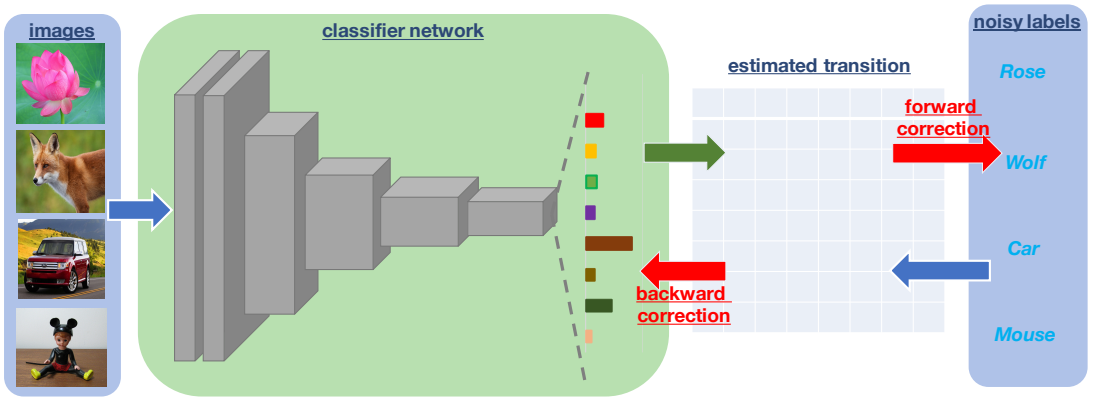
$$\textbf{Conjecture: } \text{simple } ♠ \xrightarrow[\text{memorizing order}]{\text{simple } (x,y) \mid \text{hard}(x,y)} \text{hard } ♠$$

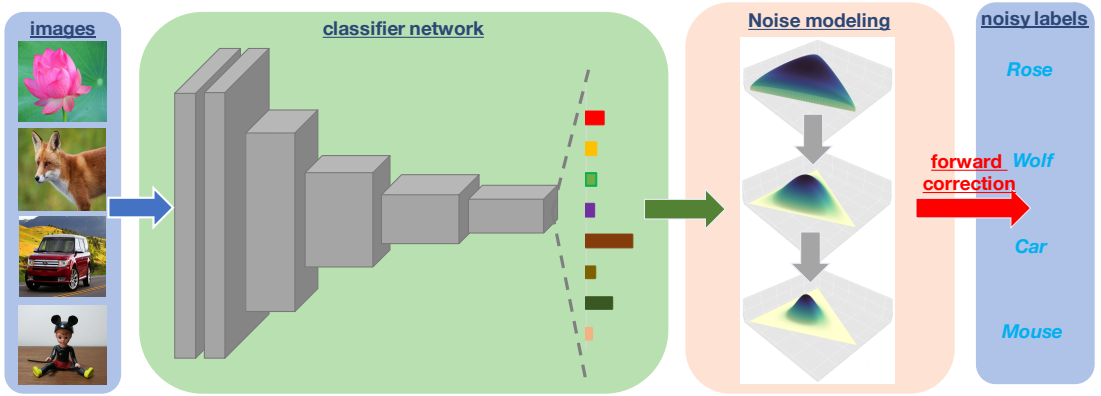
Since most simple clean data belongs to simple (x, y), we can detain memorizing in the early phase by dropout regularization.

Outline

Motivation

In this work, we focus on learning with noise transition.





- Backward correction

$$\min \underbrace{T^{-1}}_{fixed} * \ell\left(\tilde{y}, P(y|x)\right)$$

- Forward correction

$$\min -\ln\left(\underbrace{T}_{fixed} * P(y|x)\right)$$

- Noise adaptation

$$\min -\ln\left(\underbrace{\phi}_{tunable} * P(y|x)\right)$$

Motivation

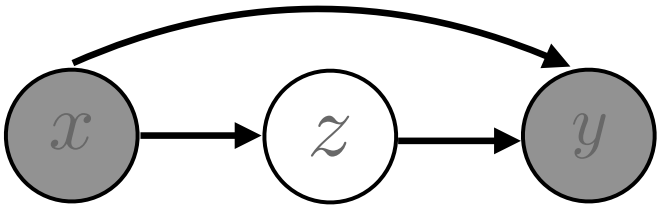
Problems

- Forward and backward corrections critically depends on the accurate estimation of noise transition, which is impractical via a finite anchor set in real-world scenarios.
- Stochastic approximation to EM learning between the classifier and the noise model via a noise adaptation layer is not rigorous and suffers from instability in tuning.

Solutions

- Learning on posterior labels along with learning the noise transition is more reliable than that on noisy labels.
- Gibbs sampling for Bayesian class-conditional noise model can reduce computational costs and avoid tweaking issues.

Latent Class-Conditional Noise Model



Learning on Posterior Labels

$$\tilde{z}_m \sim \underbrace{P(\tilde{z}_m|x_m, y_m)}_{\text{Posterior}} \propto \underbrace{P(\tilde{z}_m|x_m)}_{\text{Classifier}} \underbrace{P(y_m|x_m, \tilde{z}_m)}_{\text{Noise transition}} \tag{4}$$

Explicitly Decoupled Minimization:

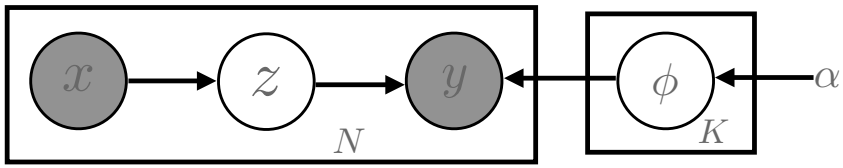
$$\min \mathbb{E}_{x,\tilde{z}} \left[-\ln P(\tilde{z}|x) \right] \text{ and } \min \mathbb{E}_{x,\tilde{z},y} \left[-\ln P(y|x,\tilde{z}) \right] \tag{5}$$

Classification Risk:

$$\mathbb{E}_{x,z} \left[-\ln P(z|x) \right] \leftarrow \mathbb{E}_{x,\tilde{z}} \left[-\ln P(\tilde{z}|x) \right] \text{ if } P(x,z) \leftarrow P(x,\tilde{z})$$

Latent Class-Conditional Noise Model

Dynamic Label Regression



Dynamic Label Regression

If it is the **latent class-conditional noise** (**LCCN**), *i.e.*,

$$P(y|z,x) = P(y|z) = \phi_{zy},$$

the noise model can degrade to a nonparametric counting n_{zy}. We then simplify the computation with the Gibbs sampling,

$$\tilde{z}_m|Z^{\neg(m)} \sim \underbrace{P(\tilde{z}_m|x_m)}_{\text{Classifier}} \underbrace{\frac{\alpha_{y_m} + n^{\neg}_{\tilde{z}_m y_m}}{\sum_{k'}(\alpha_{k'}) + n^{\neg}_{\tilde{z}_m k'}}}_{\text{Noise transition}}. \quad (6)$$

Latent Class-Conditional Noise Model

Safeguarded Transition Update

Theorem

Suppose α_i is a positive smoothing scalar, N_i is the current sample number of the ith category (i=1,...,K), M_i is the sum of the sample numbers newly allocated into (positive) and removed from (negative) the ith category after a batch of training samples, and $\widehat{M}_i$ is its absolute sum of such two cases. Then, for the transition vector ϕ_i of the ith category, its variation via a training batch is characterized by the following equation,

$$\left|\phi_i^{new} - \phi_i^{old}\right| \leq \frac{|r_i| + \widehat{r}_i}{1 + r_i}$$

where $r_i = \frac{M_i}{N_i + \sum_{j=1}^{K} \alpha_j}$ *and* $\widehat{r}_i = \frac{\widehat{M}_i}{N_i + \sum_{j=1}^{K} \alpha_j}$. *According to the definition, we have* $r_i > -1$, $\widehat{r}_i \geq 0$ *and* $\widehat{r}_i \geq |r_i|$.

Latent Class-Conditional Noise Model

Safeguarded Dynamic Label Regression

Algorithm 1 Dynamic Label Regression for LCCN

Require: A noisy dataset $\mathcal{D} = \{(x_n, y_n)\}_{n=1}^{N}$, a classifier $P(\cdot|x)$ modeled by DNN f_θ, warming-up steps δ, the running epoch number L and the batch-size M.

Directly pretrain the classifier f_θ on the noisy dataset $\mathcal{D}$.
Compute the warming-up noise transition matrix ϕ'.
for epoch $i = 1$ to L **do**
    for batch $j = 1$ to $\lceil N/M \rceil$ **do**
        Let step=i$\times\lceil N/M \rceil$+j and hook a batch of samples.
        if step $< \delta$ **then**
            Substitute the transition in Equation (6) with ϕ', and sample z_n.
        else
            Sample z_n with Equation (6) for the batch.
        end if
        Update the confusion matrix $N_{(\cdot)(\cdot)}$ based on observations $\{(z_n, y_n)\}$.
        Optimize Equation (5) to learn f_θ and estimate ϕ.
    end for
end for
Output the classifier f_θ and the noise transition ϕ.

Latent Class-Conditional Noise Model

Complexity Analysis

Stochastic training a DNN model involves two steps, the forward and backward computations. In each mini-batch update, its time complexity is $\mathcal{O}(M\Lambda)$, where M is the mini-batch size and Λ is the parameter size. Here, in Algorithm 1, we additionally add a sampling operation via Equation (6) whose complexity is $\mathcal{O}(M + K^2)$ (K is the class size). Note that, the first term in the RHS of Equation (6) has been computed in the forward procedure. So the extra cost for the sampling is negligible compared to $\mathcal{O}(M\Lambda)$. An optimization for noise modeling is also negligible, which involves the normalization of a confusion matrix only and the complexity is $\mathcal{O}(K^2)$. Since the big-O complexity of each mini-batch remains the same, our method is scalable to big data.

Outline

Experiments

Asymmetric Noise & Wild Noise

(1) Inject asymmetric noise manually

CIFAR-10: $trk \xrightarrow{r} atm, brd \xrightarrow{r} apl, deer \xrightarrow{r} horse, cat \xrightarrow{r} dog$

CIFAR-100: one $\xrightarrow{r}$ the next circularly within the superclass

(2) Totally agnostic wild noise
Clothing1M: 14 predefined clothing classes.
WebVision17: 1000 classes same to those of ImageNet.

Experiments

Experimental Setup

CIFAR-10 and CIFAR-100:

- Resnet-32 and data augmentation
- mo=0.9, weight-decay=1e-4, batch-size=128
- 40 (lr=0.5), 80 (lr=0.1), 120 (lr=0.01) epochs

Clothing1M and WebVision17:

- Resnet-50 (ImageNet pretrained) and data augmentation
- lr=0.01, mo=0.9, weight-decay=1e-3, batch-size=32
- 10 epochs (lr is divided by 10 after each 5 epochs)

Baselines:

CE, Forward, S-adaptation, Bootstrapping, Joint Optimization.

Experiments

CIFAR-10 and CIFAR-100

Dataset		CIFAR-10				
#	Method \ Noise Ratio	0.1	0.3	0.5	0.7	0.9
1	CE	90.10	88.12	76.93	59.01	56.85
2	Bootstrapping	90.73	88.12	76.29	57.04	56.79
3	Forward	90.86	89.03	82.47	67.11	57.29
4	S-adaptation	91.02	88.83	86.79	72.74	60.92
5	LCCN	**91.35**	**89.33**	**88.41**	**79.48**	**64.82**
6	CE with the clean data	91.63				

Table: The average accuracy(%) over 5 trials on noisy CIFAR-10.

Experiments

CIFAR-10 and CIFAR-100

Dataset		CIFAR-100				
#	Method \ Noise Ratio	0.1	0.2	0.3	0.4	0.5
1	CE	66.15	64.31	60.11	51.68	33.37
2	Bootstrapping	66.48	64.61	63.01	55.27	**34.52**
3	Forward	65.43	62.72	61.28	52.64	33.82
4	S-adaptation	65.52	64.11	62.39	52.74	30.07
5	LCCN	**67.83**	**67.63**	**66.86**	**65.52**	33.71
6	CE with the clean data	69.41				

Table: The average accuracy(%) over 5 trials on noisy CIFAR-100.

Experiments

CIFAR-10 and CIFAR-100

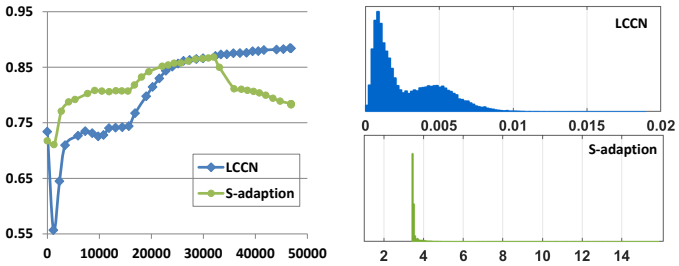
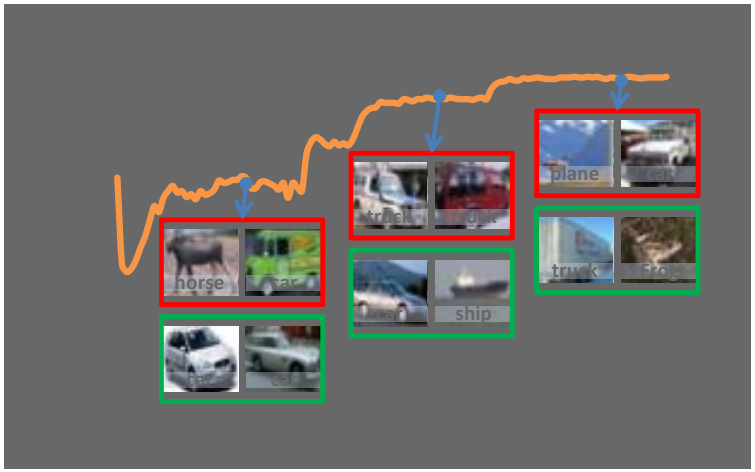


Figure: Test accuracy of LCCN and S-adaptation on CIFAR-10 with r=0.5 (left), and the corresponding histogram (right) for the variation of ϕ via a batch of samples.



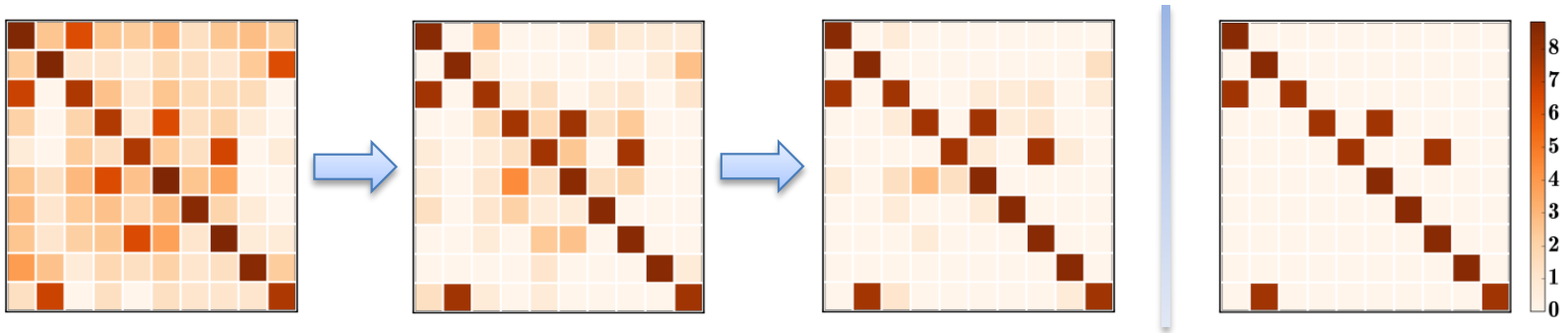


Figure: The colormap for the confusion matrix on CIFAR-10 with r=0.5. We use the log-scale of each entry in the confusion matrix for fine-grained visualization. The left three maps are gradually learned by LCCN and the right one is the groudtruth.

Experiments

#	Method	Accuracy
1	CE	68.94
2	Bootstrapping	69.12
3	Forward	69.84
4	S-adaptation	70.36
5	Joint Optimization	72.23
6	LCCN	**73.07**
7	CE with the clean data	75.28

#	Method	Accuracy@1	Accuracy@5
1	CE	63.11	83.69
2	Bootstrapping	63.20	83.81
3	Forward	63.10	83.78
4	S-adaptation	62.54	81.73
5	LCCN	**63.52**	**84.27**

Table: Results on Clothing1M (top) and WebVision (bottom).

Outline

Conclusion

Summary

- We present a Latent Class-Conditional Noise model to solve the issues when training the classifier and modeling the noise transition together in previous models .
- A dynamic label regression method is deduced for LCCN. Theoretical analysis guarantees the safeguarded transition update and introduces the negligible computational cost.
- Experiments on CIFAR-10, CIFAR-100 datasets and the real-world noisy datasets, confirm the superiority of our model compared with some state-of-the-art methods.
- More future works based on LCCN can be extended to deal with general multiple sources of noise in practise.

Q&A
Thank you!