Meta Soft Label Generation for Noisy Labels

Görkem Algan*^, Ilkay Ulusoy*
*Middle East Technical University, Ankara
^ASELSAN, Ankara

Abstract

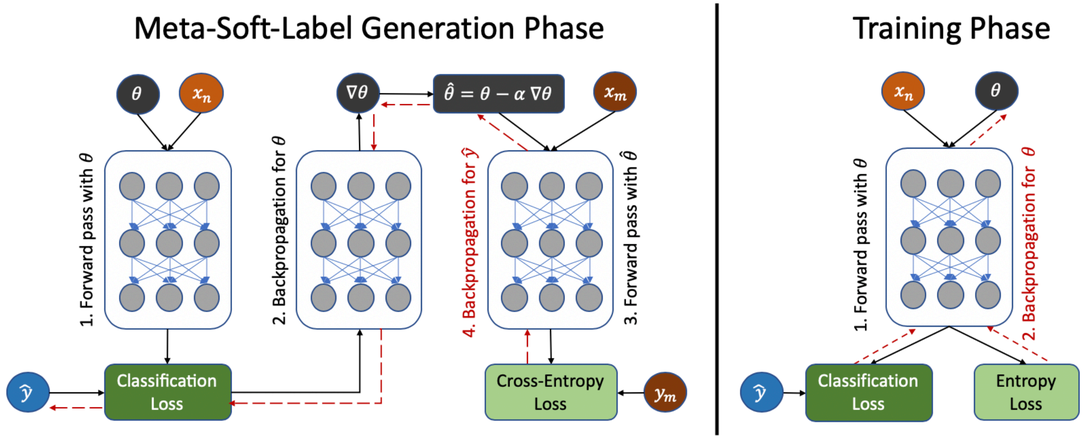
The existence of noisy labels in the dataset causes significant performance degradation for deep neural networks (DNNs). To address this problem, we propose a Meta Soft Label Generation algorithm called MSLG, which can jointly generate soft labels using meta-learning techniques and learn DNN parameters in an end-to-end fashion. Our approach adapts the meta-learning paradigm to estimate optimal label distribution by checking gradient directions on both noisy training data and noise-free meta-data. In order to iteratively update soft labels, meta-gradient descent step is performed on estimated labels, which would minimize the loss of noise-free meta samples. In each iteration, the base classifier is trained on estimated meta labels. MSLG is model-agnostic and can be added on top of any existing model at hand with ease. We performed extensive experiments on CIFAR10, Clothing1M and Food101N datasets. Results show that our approach outperforms other state-of-the-art methods by a large margin.

Contributions

- We propose an end-to-end framework called MSLG, which can simultaneously generate meta-soft-labels and learn base classifier parameters. It seeks for label distribution that would maximize gradient steps on meta dataset by checking the consistency of gradient directions.
- Our algorithm needs a clean subset of data as meta-data. However, required clean subset of data is very small compared to whole dataset (less than 2% of training data), which can easily be obtained in many cases. Our algorithm can effectively eliminate noise even in the extreme scenarios such as 80% noise.
- Our proposed solution is model agnostic and can easily be adopted by any classification architecture at hand. It requires minimal hyper-parameter tuning and can easily adapt to various datasets from various domains. We conduct extensive experiments to show the effectiveness of the proposed method under both synthetic and real-world noise scenarios.

The Proposed Algorithm

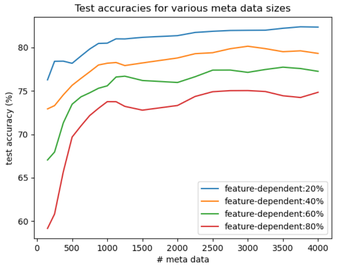
The overall architecture of MSLG is illustrated in Figure below, which consists of two consecutive stages: meta-soft-label generation and training. For each batch of data, first posterior label distribution is updated according to meta-objective, and then the base classifier is trained on these predicted labels.



Training consists of two consecutive stages. In the first stage, θ is updated on noisy training data x_n and its predicted label $\hat{y}$. Then forward pass is done with updated parameters $\hat{\theta}$ on meta data x_m with clean labels y_m. Afterward, gradients are backpropagated for optimal prediction of $\hat{y}$. In the second stage, network is trained on noisy data batch with predicted labels and additional entropy loss.

Results

Dataset	CIFAR10								Clothing1M	Food101N
Noise type	Uniform				Feature-dependent				---	---
Noise ratio (%)	20	40	60	80	20	40	60	80	---	---
Symmetric CE	82.72	79.79	74.09	54.56	76.21	67.76	fail	fail	71.02	fail
Generalized CE	84.62	**81.98**	**74.48**	44.36	81.21	71.80	66.56	fail	67.85	71.60
Bootstrap	82.51	76.97	66.13	38.41	81.24	71.63	69.74	23.25	69.35	78.03
Co-teaching	85.96	80.24	70.38	41.22	81.19	72.47	67.67	18.66	69.63	78.95
Forward Loss	83.31	80.25	71.34	28.77	77.60	69.21	39.23	fail	70.94	fail
Joint Opt.	83.74	78.75	68.17	39.22	81.61	74.03	72,15	44.15	72.16	76.12
PENCIL	**83.34**	79.27	71.41	46.57	81.62	75.08	69.24	fail	73.49	78.26
MLNT	83.20	78.14	66.34	40.80	82.46	72.52	70.12	fail	73.47	fail
Meta-Weight	84.12	80.68	71.78	46.71	81.06	71.50	67.50	22.28	73.72	76.12
MSLG	83.48	78.82	72.92	**56.26**	**82.62**	**79.30**	**77.33**	**74.87**	**76.02**	**79.06**



Loss Functions

Classification loss: We defined our meta loss as KL-divergence loss with a slight trick. KL-divergence loss is not symmetric, meaning that $\mathrm{KL}(\mathrm{P}||Q) \neq \mathrm{KL}(\mathrm{Q}||P)$. Therefore, we adopted KL-divergence loss but with $f_\theta(x_i)$ and y_i in reverse order than traditional use, as follows:

$$L_c(f_\theta(x), \hat{y}) = \mathrm{KL}(f_\theta(x_i)||y_i), \text{ where}$$

$$\mathrm{KL}(f_\theta(x_i)||y_i) = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C} f_\theta^j(x_i)\log\left(\frac{f_\theta^j(x_i)}{y_i^j}\right)$$

Entropy loss: This property is useful to prevent learning curve to halt since network predictions are forced to be different than estimated soft labels $\hat{y}$.

$$L_e(f_\theta(x)) = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C} f_\theta^j(x_i)\log(f_\theta^j(x_i))$$

Formulation of $\hat{y}$

In MSLG, label distribution y^d is maintained for all training samples x_i. We initialized y^d using noisy labels $\tilde{y}$ with the following formula

$$y^d = K\,\tilde{y}$$

where K is a large constant. Then softmax is applied to get normalized soft labels

$$\hat{y} = softmax(y^d)$$

This setup provides unconstrained learning for y^d while producing valid soft labels $\hat{y}$ all the time.

Overall Training

We propose a two-stage framework for our algorithm. In the first stage we train the base classifier with traditional SGD on noisy labels as warm-up training, and in the second stage we employ MSLG algorithm. Warm-up training is advantageous for two reasons. Firstly, works show that in the presence of noisy labels, deep networks initially learn useful representations and overfit the noise only in the later stages. Therefore, we can still leverage useful information in initial epochs. Secondly, we update $\hat{y}$ according to predictions coming from the base network. Without any pre-training, random feedback coming from the base network would cause $\hat{y}$ to lead in the wrong direction.