Some Papers about Reweighting

Narsil Zhang

Oct. 2019

Outline

Reweight is a method, not our goal.
Goal: class imbalance, noisy data(label)

Problems

- imbalance of number of data across each class
- the gradient information is dominant by numerous ”easy” examples

Focal Loss [4]

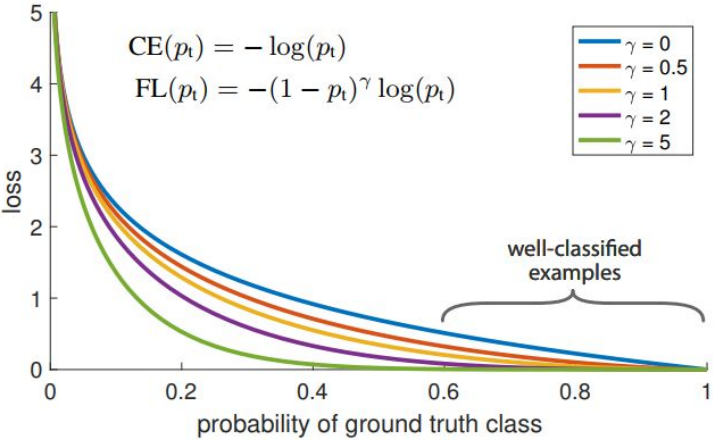


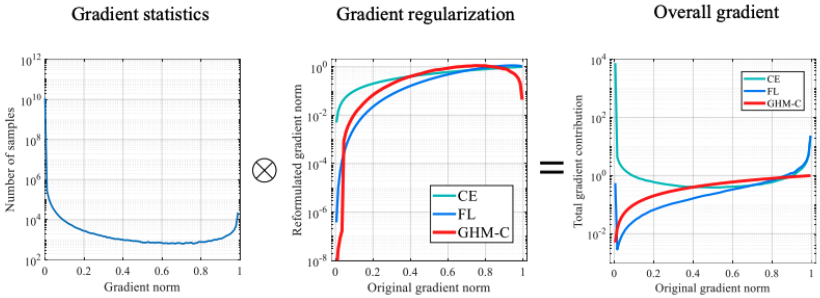
Figure: Focal Loss, t denotes class

Gradient Harmonized Mechanism [3]

$$L_{GHM} = \sum_{i=1}^{N} \frac{L_{CE}(p_i)}{GD(g_i)}$$

where g_i is the gradient norm relevant with i-th data and GD denotes some "gradient density" estimation

Effects



GHM can suppress those extremely hard examples.

Effective number of samples

Denote the effective number of samples E_n is the expected volume of n samples.

Proposition

$$E_n = \frac{1-\beta^n}{1-\beta}, \quad \beta = \frac{n-1}{n}$$

Effective number of samples

Denote the effective number of samples E_n is the expected volume of n samples.

Proposition

$$E_n = \frac{1-\beta^n}{1-\beta}, \quad \beta = \frac{n-1}{n}$$

Proof.

Induction. Denote $p = \frac{E_{n-1}}{N}$, then

$$E_n = pE_{n-1} + (1-p)\left(E_{n-1}+1\right) = 1 + \frac{N-1}{N}E_{n-1}$$

□

Class-balanced loss[1]

Suppose class y has n_y training samples, the class-balanced (CB) softmax cross-entropy loss is:

$$\mathrm{CB}_{\text{softmax}}(\mathbf{z}, y) = -\frac{1-\beta}{1-\beta^{n_y}} \log\left(\frac{\exp(z_y)}{\sum_{j=1}^{C} \exp(z_j)}\right)$$

where class y has n_y samples; In practice β is hyperparamenter.

Class-balanced loss[1]

Suppose class y has n_y training samples, the class-balanced (CB) softmax cross-entropy loss is:

$$\mathrm{CB}_{\text{softmax}}(\mathbf{z}, y) = -\frac{1-\beta}{1-\beta^{n_y}} \log\left(\frac{\exp(z_y)}{\sum_{j=1}^{C} \exp(z_j)}\right)$$

where class y has n_y samples; In practice β is hyperparamenter.

Class-balanced loss can be combined with focal loss.

Bilevel Optimization

$$\theta^*(w) = \arg\min_{\theta} \sum_{i=1}^{N} w_i f_i(\theta)$$

$$w^* = \arg\min_{w, w \geq 0} \frac{1}{M} \sum_{i=1}^{M} f_i^v \left(\theta^*(w)\right)$$

Influence analysis

$$f_{i,\epsilon}(\theta) = \epsilon_i f_i(\theta)$$

$$\hat{\theta}_{t+1}(\epsilon) = \theta_t - \alpha \nabla \sum_{i=1}^{n} f_{i,\epsilon}(\theta)\Big|_{\theta=\theta_t}$$

$$\epsilon_t^* = \arg\min_{\epsilon} \frac{1}{M} \sum_{i=1}^{M} f_i^v\left(\theta_{t+1}(\epsilon)\right)$$

v denotes validation loss.

Learning to Reweight Examples for Robust Deep Learning[5]

$$u_{i,t} = -\eta \frac{\partial}{\partial \epsilon_{i,t}} \frac{1}{m} \sum_{j=1}^{m} f_j^v \left(\theta_{t+1}(\epsilon)\right)\Big|_{\epsilon_{i,t}=0}$$

$$w_{i,t} = \frac{|u_{i,t}|}{\sum_j |u_{j,t}|}$$

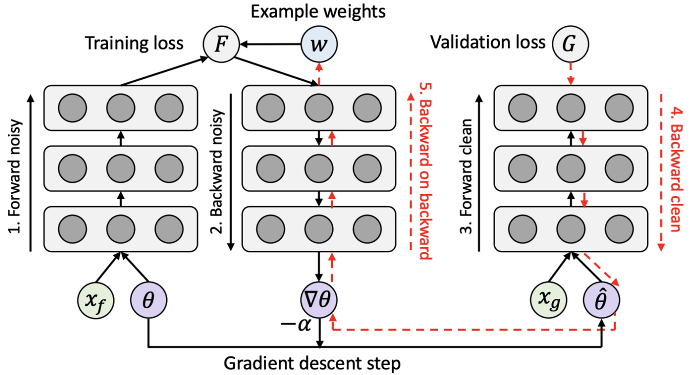


Figure 1. Computation graph of our algorithm in a deep neural network, which can be efficiently implemented using second order automatic differentiation.

Meta-NN

Use a meta-NN $\mathcal{V}(L;\Theta)$ to give the reweight coefficient for loss term L.

$$\mathbf{w}^*(\Theta) = \arg\min_{\mathbf{w}} \mathcal{L}^{\text{train}}(\mathbf{w};\Theta) \triangleq \frac{1}{N}\sum_{i=1}^{N} \mathcal{V}\left(L_i^{\text{train}}(\mathbf{w});\Theta\right) \cdot L_i^{\text{train}}(\mathbf{w})$$

Meta-NN

Use a meta-NN $\mathcal{V}(L;\Theta)$ to give the reweight coefficient for loss term L.

$$\mathbf{w}^*(\Theta) = \arg\min_{\mathbf{w}} \mathcal{L}^{\text{train}}(\mathbf{w};\Theta) \triangleq \frac{1}{N}\sum_{i=1}^{N} \mathcal{V}\left(L_i^{\text{train}}(\mathbf{w});\Theta\right) \cdot L_i^{\text{train}}(\mathbf{w})$$

Alternatively update $\mathbf{w}$ and Θ to minimize loss

$$\Theta^* = \arg\min_{\Theta} \mathcal{L}^{\text{meta}}\left(\mathbf{w}^*(\Theta)\right) \triangleq \frac{1}{M}\sum_{i=1}^{M} L_i^{\text{meta}}\left(\mathbf{w}^*(\Theta)\right)$$

Meta-weight-net[6]

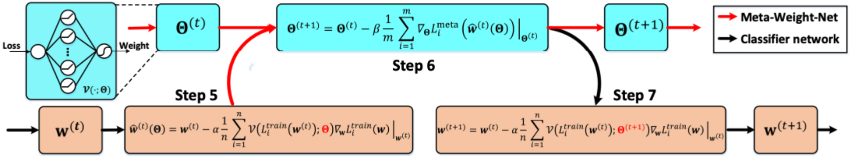


Figure 2: Main flowchart of the proposed MW-Net Learning algorithm (steps 5-7 in Algorithm 1).

Curriculum

$$\min_{\mathbf{w},\mathbf{v}} \mathbb{F}(\mathbf{w},\mathbf{v}) = \frac{1}{n}\sum_{i=1}^{n} \mathbf{v}_i^T \mathbf{L}\left(\mathbf{y}_i, g_s\left(\mathbf{x}_i,\mathbf{w}\right)\right) + G(\mathbf{v};\lambda) + \theta\|\mathbf{w}\|_2^2$$

Each G gives a curriculum.

MentorNet[2]

Use an nn (MentorNet $g(;\Theta)$) to learn data-driven curriculum :

$$g_m\left(\mathbf{z}_i;\Theta^*\right)=\arg\min_{v_i\in[0,1]}\mathbb{F}(\mathbf{w},\mathbf{v}),\forall i\in[1,n]$$

or

$$\Theta^*=\arg\min_{\Theta}\sum_{(\mathbf{x}_i,y_i)\in\mathcal{D}} g_m\left(\mathbf{z}_i;\Theta\right)\ell_i+G\left(g_m\left(\mathbf{z}_i;\Theta\right);\lambda\right)$$

References I

Y. Cui, M. Jia, T.-Y. Lin, Y. Song, and S. Belongie, *Class-balanced loss based on effective number of samples*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 9268–9277.

L. Jiang, Z. Zhou, T. Leung, L. Li, and L. Fei-Fei, *Mentornet: Regularizing very deep neural networks on corrupted labels*, CoRR, abs/1712.05055 (2017).

B. Li, Y. Liu, and X. Wang, *Gradient harmonized single-stage detector*, CoRR, abs/1811.05181 (2018).

T. Lin, P. Goyal, R. B. Girshick, K. He, and P. Dollár, *Focal loss for dense object detection*, CoRR, abs/1708.02002 (2017).

References II

M. Ren, W. Zeng, B. Yang, and R. Urtasun, *Learning to reweight examples for robust deep learning*, arXiv preprint arXiv:1803.09050, (2018).

J. Shu, Q. Xie, L. Yi, Q. Zhao, S. Zhou, Z. Xu, and D. Meng, *Meta-weight-net: Learning an explicit mapping for sample weighting*, arXiv preprint arXiv:1902.07379, (2019).