SGD Resample

September 30, 2019

1 Discrete Algorithm

let $\mathbf{x} \in \mathbb{R}^d$ to denote the input feature and we wish to do regression task, i.e. to approximate respose variable $y = g(x) \in \mathbb{R}$ using a three-layer neural network $\hat{y} = f(x)$. The architecture of the neural network is defined as follows:

$$\begin{aligned} \mathbf{h}_1 &= \sigma(\mathbf{W}^{(1)}\mathbf{x}) \\ \mathbf{h}_2 &= \sigma(\mathbf{W}^{(2)}\mathbf{h}_1) \\ \hat{y} &= \mathbf{W}^{(3)}\mathbf{h}_2 \end{aligned} \tag{1}$$

, where $\mathbf{W}^{(i)} \in \mathbb{R}^{m_i \times m_{i-1}}$ is the weight matrix in i-th layer, and m_i is the hidden size in layer i, here the input layer is layer 0 and $m_0 = d$.

The SGD resample algorithm is doing as follows

1. minimize empirical loss w.r.t. last layer, the loss function is

$$\mathcal{L}(\mathbf{W}^{(3)}) = \frac{1}{n}\sum_{i=1}^{n}(f(x) - y)^2 + \frac{\lambda_3}{1}\|\mathbf{W}^{(3)}\|_F$$

2. resample weights defined in layer 2: minimize the regularization terms w.r.t. weights connected to layer 2 and fix function value in layer 3 invariant:

$$\mathcal{L}(\mathbf{W}^{(2)}, \mathbf{W}^{(3)}) = \frac{\gamma}{n}\sum_{i=1}^{n}(\hat{y} - \hat{y}^{old})^2 + \frac{\lambda_3}{1}\|\mathbf{W}^{(3)}\|_F + \frac{\lambda_2}{m_2}\|\mathbf{W}^{(2)}\|_F$$

, where γ are hyperparameters that control the constraint of invariance of function value, x^{old} denote the respective value x in original neural network.

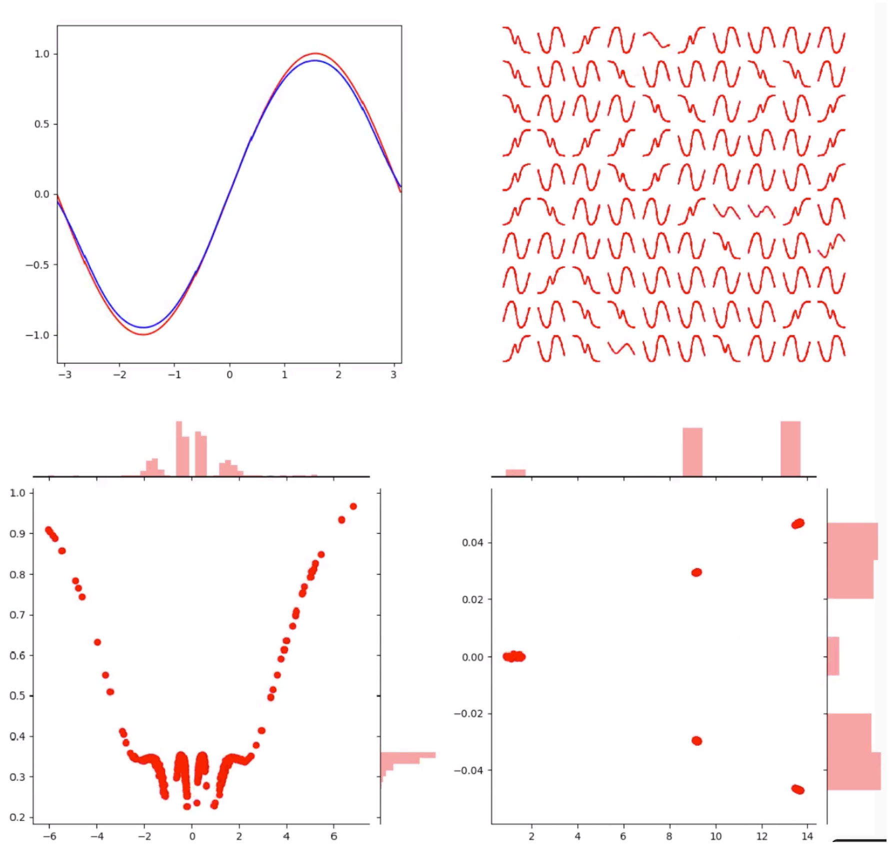
3. resample weights defined in layer 1: minimize the regularization terms w.r.t. weights connected to layer 1 and fix function value in layer 2 invariant:

$$\mathcal{L}(\mathbf{W}^{(1)}, \mathbf{W}^{(2)}) = \frac{\gamma}{n}\sum_{i=1}^{n}\|\mathbf{h}_2 - \mathbf{h}_2^{old}\|_2^2 + \frac{\lambda_2}{m_2}\|\mathbf{W}^{(2)}\|_F + \frac{\lambda_1}{m_1}\|\mathbf{W}^{(1)}\|_F$$

.

1.1 Joint Training Results

The stationary distributions of parameters learned by standard sgd are as follows:



prediction function x: input feature, y: output red: groundtruth, blue: prediction	*Samples of θ_2* x: input feature, y: response of the neuron θ_2 in layer 2 each subplot: a sample of θ_2
dist of θ_1 and $\omega(\theta_1)$ x: θ_1, y: $\omega(\theta_1) = \sqrt{\int \lvert\theta_2(\theta_1)\rvert^2 \rho(d\theta_2)}$ top and right: marginal histogram	*dist of $\lVert\theta_2\rVert_2$ and $w(\theta_2)$* x: $\lVert\theta_2\rVert_2$, y: $w(\theta_2)$ top and right: marginal histogram

final regression loss 0.3e-3, regularization loss: 3.0e-3.

1.2 Steps towards alternative training

1. use well trained $\mathbf{W}^{(1)}$, and fix them during training: only alternating between step [1] and step [2] in discrete algorithm (video fixedtheta1.mp4)
   - final regression loss: 0.3e-3, regularization loss: 2.7e-3
   - could share similar distribution w.r.t. the conventional sgd algorithm.
   - attained similar regression loss and smaller regularization loss (and regression loss is more stable)
2. use well trained $\mathbf{W}^{(1)}$ as initialization and alternating between step [1], [2], [3] (video welltheta1.mp4)
   - final regression loss: 1.4e-3, regularization loss 2.4e-3.
   - it seems that the distribution of θ_2 converges quickly but the regression loss converge very slowly.
3. use random $\mathbf{W}^{(1)}$ as initialization, do the original algorithm
   - final regression loss: 1e-2
   - could not converge (both regression loss and distribution)