確率的勾配降下法の平滑化効果を利用した段階的最適化手法によるディープニューラルネットワークの大域的最適化

05001736　明治大学　*佐藤 尚樹　SATO Naoki
01016200　明治大学　飯塚 秀明　IIDUKA Hideaki

1. はじめに

段階的最適化手法は非凸関数の大域的最適解を求めるための手法であり、画像処理や機械学習分野でよく利用される一方で、その理論的な解析は少ない。本発表は、段階的最適化アルゴリズムが大域的最適解に収束するための十分条件を示し、その条件の下での収束解析を提供する。また、確率的勾配降下法には目的関数を平滑化する効果があり、その度合いは学習率とバッチサイズが決定することを示す。最後に、確率的勾配降下法の平滑化効果を利用した段階的最適化アルゴリズムを提案し、その収束解析を提供する。

2. 段階的最適化

段階的最適化手法は非凸最適化問題の大域的最適解を探索する大域的最適化手法の一つである。まず徐々に小さくなるノイズ $(\delta_m)_{m\in[M]}$ による平滑化演算によって、徐々に元の目的関数 $f\colon \mathbb{R}^d \to \mathbb{R}$ に近づくような平滑化された M 個の関数の列 $(\hat{f}_{\delta_m})_{m\in[M]}$ を用意する。そして、最も平滑化された関数 $\hat{f}_{\delta_1}$ を最初に最適化し、その近似解を初期点として 2 番目に大きく平滑化された関数 $\hat{f}_{\delta_2}$ を最適化し、次に 2 番目の近似解を初期点として 3 番目に大きく平滑化された関数 $\hat{f}_{\delta_3}$ を最適化する、という手順を繰り返すことで、元の目的関数 f の局所的最適解を避けて大域的最適解を探索する。一般に関数の平滑化は、正規分布や一様分布に従う確率変数で関数を畳み込むことで実現される。

定義 1 (関数の平滑化) *L_f-Lipschitz* 関数 f を平滑化して得られる関数 $\hat{f}_\delta\colon \mathbb{R}^d \to \mathbb{R}$ は、

$$\hat{f}_\delta(\boldsymbol{x}) := \mathbb{E}_{\boldsymbol{u}\sim B(\boldsymbol{0};1)}\left[f(\boldsymbol{x}-\delta\boldsymbol{u})\right]$$

と表される。ここで、$\delta \in \mathbb{R}$ は平滑化の度合いを表し、$\boldsymbol{u} \in \mathbb{R}^d$ は閉球 $B(\boldsymbol{0};1)$ から一様にサンプリングされたベクトルである。また、

$$\boldsymbol{x}^\star := \operatorname*{argmin}_{\boldsymbol{x}\in\mathbb{R}^d} f(\boldsymbol{x}),\quad \boldsymbol{x}^\star_\delta := \operatorname*{argmin}_{\boldsymbol{x}\in\mathbb{R}^d} \hat{f}_\delta(\boldsymbol{x})$$

とする。

Hazan ら [1] は、ある条件を満たす非凸関数の族である σ-nice 関数を定義し、段階的最適化アルゴリズムが σ-nice 関数の大域的最適解に収束することを示したが、σ-nice 関数がどの程度特別な関数なのかは不明であった。そこで、σ-nice 関数を拡張した new σ-nice 関数を定義し、任意の非凸関数が new σ-nice 関数であるための十分条件を示す。

定義 2 (new σ-nice 関数) 任意の関数 $f\colon \mathbb{R}^d \to \mathbb{R}$ に対して次の 2 つの条件が満たされるとき、関数 f は *new σ-nice* 関数であるという。

(i) 任意の $\delta_m \in \mathbb{R}$ $(m \in [M])$ と $\boldsymbol{x}^\star_{\delta_m}$ に対して、

$$\left\|\boldsymbol{x}^\star_{\delta_m} - \boldsymbol{x}^\star_{\delta_{m+1}}\right\| \le |\delta_m| - |\delta_{m+1}|,$$

が成り立つ。ただし、$|\delta_{m+1}| := \gamma_m|\delta_m|, \gamma_m \in (0,1)$ とする。

(ii) 任意の $\delta_m \in \mathbb{R}$ $(m \in [M])$ に対して、関数 $\hat{f}_{\delta_m}(\boldsymbol{x})$ は近傍 $N(\boldsymbol{x}^\star; d_m\delta_m)$ で σ-強凸である。

命題 1 関数 $f\colon \mathbb{R}^d \to \mathbb{R}$ が、十分小さな正数 $r > 0$ に対して閉球 $B(\boldsymbol{x}^\star; r)$ で σ-強凸で、任意の $m \in [M]$ に対して、$\boldsymbol{x}^\star_{\delta_m}$ は近傍 $N(\boldsymbol{x}^\star; d_m|\delta_m|)$ 内に含まれると仮定する。このとき、関数 f が *new σ-nice* 関数であるための十分条件は、任意の $m \in [M]$ に対して、ノイズの大きさ $|\delta_m|$ が次の条件を満たすことである。

$$\frac{2\max\left\{\left\|\nabla\hat{f}_{\delta_m}(\boldsymbol{x}^\star)\right\|, \left\|\nabla\hat{f}_{\delta_{m+1}}(\boldsymbol{x}^\star)\right\|\right\}}{\sigma\left(1-\gamma_m\right)} \le |\delta_m| = \left|\delta^-\right| \quad (1)$$

ただし、$\boldsymbol{x} \in N(\boldsymbol{x}^\star; a_m r), \boldsymbol{u}_m \sim B(\boldsymbol{0};1)$ に対して

$$|\delta^-| := \Big|\|\boldsymbol{x}^\star - \boldsymbol{x}\|\|\boldsymbol{u}_m\|\cos\theta - \sqrt{\|\boldsymbol{x}^\star - \boldsymbol{x}\|^2\|\boldsymbol{u}_m\|^2\cos^2\theta - r^2(a_m^2-1)}\Big|$$

とし、θ は $\boldsymbol{u}_m$ と $\boldsymbol{x}^\star - \boldsymbol{x}$ のなす角とする。

命題 1 より、式 (1) を満たすようなノイズ δ_m で関数を平滑化すれば、任意の関数 f は new σ-nice 関数となる。new σ-nice 関数を段階的最適化アルゴリズムで最適化すると、$\mathcal{O}\left(1/\epsilon^{\frac{1}{p}+2}\right)\,(p\in(0,1))$ 回の反復で大域的最適解 $\boldsymbol{x}^\star$ の ϵ 近傍に到達する [1, Theorem 3.2]。しかし、一般にディープニューラルネットワーク (DNN) を含む非凸関数の畳み込み演算を計算することは困難であり、このアルゴリズムは適用できない。

3. 確率的勾配降下法の平滑化特性

確率的勾配降下法は式 (3) のように点列を更新する。ただし、η は学習率で、$\nabla f_{\mathcal{S}_t}$ はミニバッチ $\mathcal{S}_t$ で全勾配 ∇f を推定したミニバッチ確率的勾配である。最急降下方向と確率的勾配降下法の探索方向の間には各反復で $\boldsymbol{\omega}_t := \nabla f_{\mathcal{S}_t}(\boldsymbol{x}_t) - \nabla f(\boldsymbol{x}_t)$ だけノイズが生じる。時刻 t において、最急降下法で点列を更新した先を $\boldsymbol{y}_t$ とし、確率的勾配降下法で点列を更新した先を $\boldsymbol{x}_{t+1}$ とする。すなわち、

$$\boldsymbol{y}_t := \boldsymbol{x}_t - \eta\nabla f(\boldsymbol{x}_t), \tag{2}$$

$$\boldsymbol{x}_{t+1} := \boldsymbol{x}_t - \eta\nabla f_{\mathcal{S}_t}(\boldsymbol{x}_t) \tag{3}$$

とすると、

$$\begin{aligned}\mathbb{E}_{\boldsymbol{\omega}_t}\left[\boldsymbol{y}_{t+1}\right] &= \boldsymbol{y}_t - \eta\nabla\mathbb{E}_{\boldsymbol{\omega}_t}\left[f(\boldsymbol{y}_t - \eta\boldsymbol{\omega}_t)\right]\\ &= \boldsymbol{y}_t - \eta\nabla\mathbb{E}_{\boldsymbol{u}_t\sim B(\mathbf{0};1)}\left[f\left(\boldsymbol{y}_t - \frac{\eta C}{\sqrt{b}}\boldsymbol{u}_t\right)\right]\\ &= \boldsymbol{y}_t - \eta\nabla\hat{f}_{\frac{\eta C}{\sqrt{b}}}(\boldsymbol{y}_t)\end{aligned}$$

が成り立つ。ただし、b はバッチサイズ、C は確率的勾配の分散を表している。よって、確率的勾配降下法で関数 f を最適化することは、期待値の意味ではノイズ $\frac{\eta C}{\sqrt{b}}$ で f を平滑化した関数 $\hat{f}_{\frac{\eta C}{\sqrt{b}}}$ を最急降下法で最適化することと等価であると言える。

4. 提案手法

確率的勾配降下法を使用するというだけで、目的関数 f は暗黙的にある程度平滑化されていて、その度合いは学習率とバッチサイズによって定まることが分かった。そこで、この特性を利用して段階的最適化アルゴリズムを構成する。すなわち、学習の初期段階では、十分大きな学習率と十分小さなバッチサイズを使用し、徐々に学習率は小さく、バッチサイズは大きくしていくことで、暗黙的な段階的最適化を実現する。new σ-nice 関数をアルゴリズム 1 で最適化すると、$\mathcal{O}\left(1/\epsilon^{\frac{1}{p}}\right)\,(p\in(0,1))$ 回の反復で大域的最適解 $\boldsymbol{x}^\star$ の ϵ 近傍に到達する [2, Theorem 3.4]。したがって、フルバッチサイズでも実行できる計算機が用意できれば、DNN を含む非凸関数の大域的最適化が可能となる。

アルゴリズム 1 暗黙的な段階的最適化

Require: $\epsilon>0, p\in(0,1), \bar{d}>0, \boldsymbol{x}_1, \eta_1, b_1$

$\delta_1 := \frac{\eta_1 C}{\sqrt{b_1}}$

$\alpha_0 := \min\left\{\frac{\sqrt{b_1}}{4L_f\eta_1 C(1+\bar{d})}, \frac{\sqrt{b_1}}{\sqrt{2}\sigma\eta_1 C}\right\}, M^p := \frac{1}{\alpha_0\epsilon}$

for $m=1$ to $M+1$ **do**

  if $m\neq M+1$ **then**

    $\epsilon_m := \sigma^2\delta_m^2,\ T_F := H_4/\left(\epsilon_m - H_3\eta_m\right)$

    $\gamma_m := \frac{(M-m)^p}{\{M-(m-1)\}^p}$

    $\kappa_m/\sqrt{\lambda_m} = \gamma_m\ (\kappa_m\in(0,1], \lambda_m\geq 1)$

  end if

  $\boldsymbol{x}_{m+1} := \mathrm{SGD}(T_F, \boldsymbol{x}_m, \hat{f}_{\delta_m}, \eta_m, b_m)$

  $\eta_{m+1} := \kappa_m\eta_m, b_{m+1} := \lambda_m b_m$

  $\delta_{m+1} := \frac{\eta_{m+1}C}{\sqrt{b_{m+1}}}$

end for

  return $\boldsymbol{x}_{M+2}$

アルゴリズム 2 確率的勾配降下法 (SGD)

Require: $T_F, \hat{\boldsymbol{x}}_1, F, \eta, b$

for $t=1$ to T_F **do**

  $\hat{\boldsymbol{x}}_{t+1} := \hat{\boldsymbol{x}}_t - \eta\nabla F_{\mathcal{S}_t}(\boldsymbol{x}_t)$

end for

  return $\hat{\boldsymbol{x}}_{T_F+1} = \mathrm{SGD}(T_F, \hat{\boldsymbol{x}}_1, F, \eta, b)$

参考文献

[1] E. Hazan, K. Yehuda and S. Shalev-Shwartz, On Graduated Optimization for Stochastic Non-Convex Problems, Proceedings of The 33rd International Conference on Machine Learning, 48, 1833-1841, 2016.

[2] N. Sato and H. Iiiduka, Using Stochastic Gradient Descent to Smooth Nonconcex Functions: Analysis of Implicit Graduated Optimization with Optimal noise Scheduling. arXiv, https://arxiv.org/abs/2311.08745, 2023.