Holonomic gradient method (HGM) の Neural tangent kernel (NTK) への応用

迫田章裕, 高山信毅

- [st-2024] A.Sakoda, N.Takayama, An Application of the Holonomic Gradient Method to the Neural Tangent Kernel, http://arxiv.org/abs/2410.23626

Neural Network (NN), Kernel method とは

$$\mathrm{R}^{d_0} \ni x \mapsto W^{(1)}x + b^{(1)} =: g^{(1)} \mapsto \sigma^{(1)}(g^{(1)}) \in \mathrm{R}^{d_1}$$

なる形の写像を $L+1$ 個合成した写像 $f(\theta, x) \in \mathrm{R}^{d_{L+1}}$. $\sigma^{(1)}$ を activation function と呼ぶ. $W^{(1)}$: weight matrix, $b^{(1)} \in \mathrm{R}^{d_2}$: bias vector.
θ: パラメータ weight matrix, bias vector の全部.
$d_{L+1} = 1$, $\sigma^{(L+1)} = \mathrm{id}$, $\sigma^{(i)} = \sigma$ とする. Neural tangent kernel:

$$K(x, x') = \left\langle \frac{\partial f(x, \theta)}{\partial \theta}, \frac{\partial f(x', \theta)}{\partial \theta} \right\rangle \tag{1}$$

x_i 入力, y_i 出力, トレーニングデータ:
$\{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\}$.
kernel matrix: $H^* = (K(x_i, x_j)) \in \mathrm{R}^{N \times N}$. Kernel method

$$f(x) \sim (K(x, x_1), K(x, x_2), \ldots, K(x, x_N))(H^*)^{-1}(y_1, y_2, \ldots, y_N)^T. \tag{2}$$

定理: Jacot et al 2018[a]. NN の幅が $d_i \to \infty$ ならほとんどすべての θ について $K(x,x') \sim \Theta(x,x')$ となる.

[a] https://arxiv.org/abs/1806.07572

ネットの記事など [1]. Θ (NTK) の定義.

$$\Sigma^{(0)}(x,x') = x^T x', \tag{3}$$

$$\Lambda^{(h)}(x,x') = \begin{pmatrix} \Sigma^{(h-1)}(x,x) & \Sigma^{(h-1)}(x,x') \\ \Sigma^{(h-1)}(x',x) & \Sigma^{(h-1)}(x',x') \end{pmatrix} \tag{4}$$

$$\Sigma^{(h)}(x,x') = c_\sigma E_{(u,v)\sim N(0,\Lambda^{(h)})}[\sigma(u)\sigma(v)] \tag{5}$$

$$\dot{\Sigma}^{(h)}(x,x') = c_\sigma E_{(u,v)\sim N(0,\Lambda^{(h)})}[\dot{\sigma}(u)\dot{\sigma}(v)] \tag{6}$$

$$\Theta(x,x') = \Theta^{(L)}(x,x') = \sum_{h=1}^{L+1} \left(\Sigma^{(h-1)}(x,x') \prod_{h'=h}^{L+1} \dot{\Sigma}(x,x') \right) \tag{7}$$

[1] https://oumpy.github.io/blog/2020/04/neural_tangents.html

Dual activation

$E_{(u,v)\sim N(0,\Lambda^{(h)})}[\sigma(u)\sigma(v)]$ (dual activation of σ) は積分で書くと

$$\hat{E}[\sigma(u)\sigma(v)] = \int_{\mathbb{R}^2} \sigma(u)\sigma(v)\exp(x_{11}u^2 + 2x_{12}uv + x_{22}v^2)dudv \quad (8)$$

$$E_{(u,v)\sim N(0,\Lambda^{(h)})}[\sigma(u)\sigma(v)] = \hat{E}[\sigma(u)\sigma(v)]\frac{\sqrt{\det(x)}}{\pi}, \quad \Lambda^{(h)} = -\frac{1}{2}x^{-1}. \quad (9)$$

- ReLU (rectified linear unit)[2]: $\sigma(u) = uY(u)$, $Y(u)$ Heaviside 関数.
- Sigmoid: $\sigma(u) = \frac{1}{1+\exp(-u)}$
- GeLU (Gaussian error linear unit): $\sigma(u) = \frac{1}{2}\left(1 + \mathrm{erf}\left(\frac{u}{\sqrt{2}}\right)\right)$

dual activation が計算できれば NTK が計算できる

[2] https://en.wikipedia.org/wiki/Activation_function

［st-2024］アルゴリズムの流れ (ReLU の場合)

HGM はパラメータ付き積分の値を計算

- $(u\partial_u - 1) \bullet uY(u) = 0.$
- 小山-竹村 2012[3] より

$$\int_{\mathrm{R}^2} uY(u)vY(v)\exp(x_{11}u^2 + 2x_{12}uv + x_{22}v^2 + y_1u + y_2v)dudv$$

は

$$\partial_1(-y_1 - 2(x_{11}\partial_1 + x_{12}\partial_2)) - 1, \quad (10)$$
$$\partial_2(-y_2 - 2(x_{12}\partial_1 + x_{22}\partial_2)) - 1, \quad (11)$$
$$\partial_{12} - 2\partial_1\partial_2, \quad (12)$$
$$\partial_{11} - \partial_1^2, \quad (13)$$
$$\partial_{22} - \partial_2^2, \quad (14)$$

を満たす ($\partial_{ij} = \partial/\partial x_{ij}$, $\partial_i = \partial/\partial y_i$). 生成する左イデアルを I_1.

[3] https://arxiv.org/abs/1211.6822

- 積分を $y_1 = y_2 = 0$ に制限した関数の満たす方程式は

$$I_2 = (I_1 + y_1 D + y_2 D) \cap \mathbb{C}\langle x_{11}, x_{12}, x_{22}, \partial_{11}, \partial_{12}, \partial_{22} \rangle$$

ここで $D = \mathbb{C}\langle y_1, y_2, x_{11}, x_{12}, x_{22}, \partial_1, \partial_2, \partial_{11}, \partial_{12}, \partial_{22} \rangle$.

- 素手の制限計算, たとえば

$$\begin{aligned} & -y_1\partial_1 - 1 - 2(x_{11}\partial_1^2 + x_{12}\partial_1\partial_2) - 1 \\ \to \quad & -y_1\partial_1 - 2(x_{11}\partial_{11} + (1/2)x_{12}\partial_{12}) - 1 - 1, \qquad \text{by (12) and (13).} \end{aligned}$$

- $g(x) = \hat{E}[\sigma(u)\sigma(v)](x)$, $\sigma = \mathrm{ReLU}$ が満たすのは

$$\begin{aligned} & 2x_{11}\partial_{11} + x_{12}\partial_{12} + 1, \\ & x_{12}\partial_{12} + 2x_{22}\partial_{22} + 1, \\ & 4\partial_{11}\partial_{22} - \partial_{12}^2 \end{aligned}$$

$$F = (1, \partial_{12})^T \bullet g.$$

$$\partial_{x_{11}} \bullet F - P_{11} F = 0, \partial_{x_{12}} \bullet F - P_{12} F = 0, \partial_{x_{22}} \bullet F - P_{22} F = 0$$

$$P_{11} = \begin{pmatrix} \frac{-1}{x_{11}} & \frac{-\frac{1}{2} x_{12}}{x_{11}} \\ \frac{2x_{12}}{x_{11}(x_{12}^2 - x_{22}x_{11})} & \frac{\frac{1}{2}(2x_{12}^2 + 3x_{22}x_{11})}{x_{11}(x_{12}^2 - x_{22}x_{11})} \end{pmatrix},$$

$$P_{12} = \begin{pmatrix} 0 & 1 \\ \frac{-4}{x_{12}^2 - x_{22}x_{11}} & \frac{-5x_{12}}{(x_{12}^2 - x_{22}x_{11})} \end{pmatrix},$$

$$P_{22} = \begin{pmatrix} \frac{-1}{x_{22}} & \frac{-\frac{1}{2} x_{12}}{x_{22}} \\ \frac{2x_{12}}{x_{22}(x_{12}^2 - x_{22}x_{11})} & \frac{\frac{1}{2}(2x_{12}^2 + 3x_{22}x_{11})}{x_{22}(x_{12}^2 - x_{22}x_{11})} \end{pmatrix}.$$

$F(-1, 0, -1) = (1/4, \pi/8)^T$. `scipy`, `solve_ivp`.

```
import("nk_restriction.rr");;
V=[y1,y2,x11,x12,x22]; DV=poly_dvar(V);
P1=poly_dmul(dy1,-y1-2*x11*dy1-2*x12*dy2,V)-1;
P2=poly_dmul(dy2,-y2-2*x12*dy1-2*x22*dy2,V)-1;
I=[P1,P2,dx11-dy1^2,dx22-dy2^2,dx12-2*dy1*dy2];
dp_gr_print(1);
Iprime=nk_restriction.restriction_ideal(I,V,DV,[1,1,0,0,0]

import("yang.rr");;
VV=[x11,x12,x22]; DVV=poly_dvar(VV);
yang.define_ring(["partial",VV]);
RII=map(dp_ptod,Iprime,DVV);
yang.verbose();
RG=yang.buchberger(RII);;
Std=[1,dx12];
Pf=yang.pfaffian(map(dp_ptod,Std,DVV),RG);
```

素手で計算してみると定理に気がつく

定理 [st-2024] $\hat{E}[u^m v^n Y(u) Y(v)](x_{11}, x_{12}, x_{22}) =$

$$\frac{1}{4}\Gamma(\alpha)\Gamma(\beta)\varphi_1 + \frac{1}{2}\Gamma\left(\alpha + \frac{1}{2}\right)\Gamma\left(\beta + \frac{1}{2}\right)\varphi_2$$

$$\varphi_1 := (-x_{11})^{-\alpha}(-x_{22})^{-\beta}\,{}_2F_1\left(\alpha, \beta, \frac{1}{2}; z\right)$$

$$\varphi_2 := (-x_{11})^{-\alpha}(-x_{22})^{-\beta}\sqrt{z}\,\mathrm{sign}\,(x_{12})\,{}_2F_1\left(\alpha + \frac{1}{2}, \beta + \frac{1}{2}, \frac{3}{2}; z\right)$$

$$\alpha = \frac{1+m}{2}, \beta = \frac{1+n}{2}, z = \frac{x_{12}^2}{x_{11}x_{22}}.$$

$${}_2F_1(1, 1, 1/2; z) = \left(1 + \frac{\sqrt{z}\arcsin(\sqrt{z})}{\sqrt{1-z}}\right)(1-z)^{-1}.$$

- Han et al 2022[4] は closed form に詳しい.

[4] https://arxiv.org/abs/2209.04121

ReSin (rectified sin) $\sigma(u) = Y(u) \sin u$

Rank 8 system. 13.1s (Risa/Asir, AMD EPYC 7552 48-Core, 1.5GHz)

Method	Training time (s)	Inference time (s)
closed	NA	NA
GaussHerm	3.916	4.949
hgm	289.5	1005
hgm all-at-once	21.07	23.39

	Kernel error
gh − hgm	0.0019103
gh − aao	0.0016427
hgm − aao	0.00062839

	Inference error
GaussHerm	0.97328
hgm	0.96874
hgm all-at-once	0.95745