Why High-rank Neural Networks Generalize?: An Algebraic Framework with RKHSs

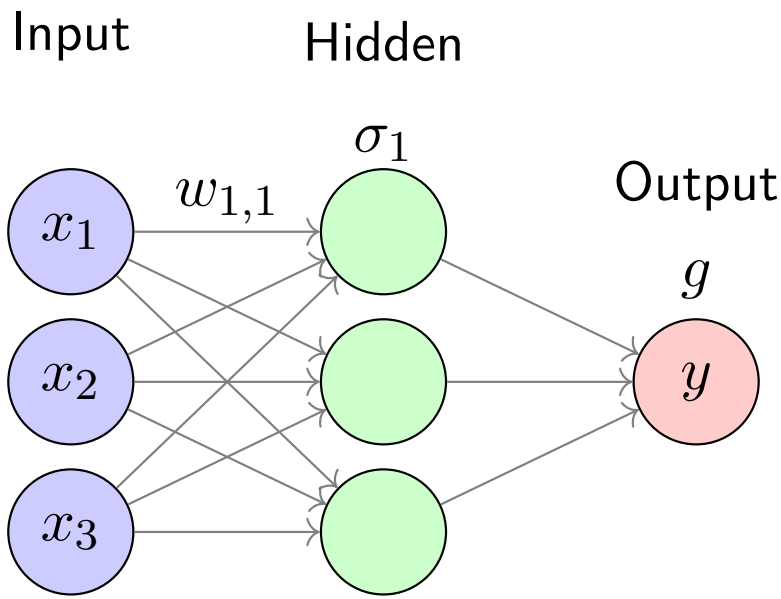
Yuka Hashimoto (NTT / RIKEN AIP)
Sho Sonoda (RIKEN AIP / CyberAgent, Inc.)
Isao Ishikawa (Kyoto University / RIKEN AIP)
Masahiro Ikeda (The University of Osaka / RIKEN AIP)

Neural Networks

Consider a neural network :

$$f = g \circ b_L \circ W_L \circ \cdots \circ \sigma_1 \circ b_1 \circ W_1 \quad (1)$$

$W_j : \mathbb{R}^{d_{j-1}} \to \mathbb{R}^{d_j}$ $(j = 1, \ldots, L)$
: weight (linear)
$b_j : \mathbb{R}^{d_j} \to \mathbb{R}^{d_j}$
: bias $(b_j(x) = x + \tilde{b}_j)$
$\sigma_j : \mathbb{R}^{d_j} \to \mathbb{R}^{d_j}$
: activation function (nonlinear)
$g : \mathbb{R}^{d_L} \to \mathbb{C}$
: nonlinear function



Generalization and Rademacher complexity

Generalization : How much does the model predict well on unseen data?

Rademacher complexity : complexity of the model
Ω : Probability space e.w. a probability measure P,
$X : \Omega \to \mathcal{X}$: random variable for data
$x_1, \ldots, x_n$: i.i.d. samples from $X_* P$, $\mathcal{F}$: $\mathbb{R}$-valued function class on $\mathcal{X}$
$s_1, \ldots, s_n$: i.i.d. Rademacher variables ($P(s_i{=}1){=}1/2$, $P(s_i{=}{-}1){=}1/2$)

$$\hat{R}_n(\mathbf{x}, \mathcal{F}) = \frac{1}{n} \int_\Omega \sup_{f \in \mathcal{F}} \sum_{i=1}^n f(x_i) s_i(\omega) \mathrm{d}P(\omega) = \frac{1}{n} \mathrm{E}\left[\sup_{f \in \mathcal{F}} \sum_{i=1}^n f(x_i) s_i \right] \quad (2)$$

Theorem 1

Let $g : \mathbb{R} \mapsto [0, 1]$ be Lipshitz continuous (error function) and $f \in \mathcal{F}$. For any $\delta \in (0, 1)$, with probability at least $1 - \delta$, we have

$$\mathrm{E}[g(f(X))] \leq \frac{1}{n} \sum_{i=1}^n g(f(x_i)) + C \hat{R}_n(\mathbf{x}, \mathcal{F}) + 3 \sqrt{\frac{\log 1/\delta}{n}} \quad (3)$$

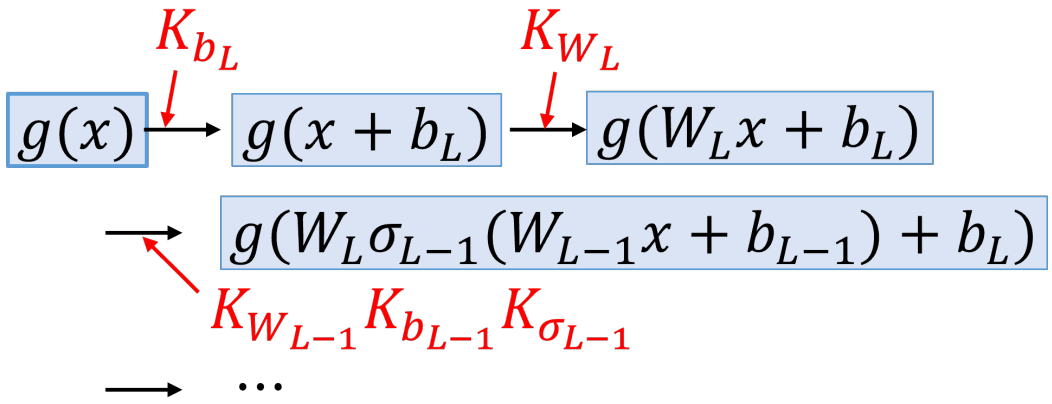
Representing a neural network using Koopman operators

H_j : Function space
For $h : \mathbb{R}^{j-1} \to \mathbb{R}^j$, Koopman operator K_h is the linear operator from H_j to H_{j-1} defined as

$$K_h v := v \circ h \quad (v \in H_j) \tag{4}$$

$$f(x) = K_{W_1} K_{b_1} K_{\sigma_1} \cdots K_{W_L} K_{b_L} g(x) \tag{5}$$

Goal : Analyze the composition structure of neural networks with Koopman operators.



Existing Koopman-based generalization bound

Question: Do the networks with high-rank weight matrices generalize well?
$\rightarrow$ Emperically, yes! But was not fully understood theoretically.

F : Class of all functions represented by the neural network (in Sobolev space).
$\mathcal{W}_j(C,D) = \{W \in \mathbb{R}^{d_{j-1}\times d_j} \mid \|W\| \le C,\ |\det W^*W|^{-1/4} \le D\}$
$F_{\mathrm{inj}}(C,D) = \{f \in F \mid W_j \in \mathcal{W}_j(C,D)\}$

Theorem 2 (Hashimoto et al. ICLR 2024)

The Rademacher complexity of the class of neural networks with injective weights is bounded as

$$\hat{R}_n(\mathbf{x}, F_{\mathrm{inj}}(C,D))) \le O\left(\sup_{W_j \in \mathcal{W}_j(C,D)} \frac{\|g\| \prod_{j=1}^{L-1} \|K_{\sigma_j}\|_{H^s} \|W_j\|^{s_{j-1}}}{\sqrt{n} \prod_{j=1}^{L} |\det W_j^* W_j|^{1/4}} \right). \quad (6)$$

The factor $\|W_j\|^{s_{j-1}}/\det(W_j^*W_j)^{1/4}$ comes from the Koopman operator with respect to W_j.

New Koopman-based generalization bound

Challenges of the existing Koopman-based bounds:

- Activation functions should be smooth and unbounded (excludes Leaky ReLU, sigmoid, tanh,...)
- The dependency of the bound on the activation function is not clear.

$\rightarrow$ We introduce a regularized model parametorized by c that goes to the practical model as $c \to \infty$. We use L^2 function spaces.

$\mathcal{W}_j(D) := \{W \in \mathbb{R}^{d_{j-1} \times d_j} \mid |\det W_j^* W_j|^{-1/4} \leq D\}$,
$F_c(D) := \{f : \text{regularized model with parameter } c \mid W_j \in \mathcal{W}_j(D)\}$.

Theorem 3

The Rademacher complexity of the class of neural networks with injective weights is bounded as

$$\hat{R}(\mathbf{x}, F_c(D)) \leq O\left(\sup_{W_j \in \mathcal{W}_j(D)} \frac{\|g\| \prod_{j=1}^{L-1} \|K_{\sigma_j}\|_{L^2} (2c/\pi)^{d_0/2}}{\sqrt{n} \prod_{j=1}^{L} |\det W_j^* W_j|^{1/4}} \right). \quad (7)$$

Advantages of the new analysis

- We have the same determinant term in the denominator.
- We can apply the bounds to wider range of neural networks (e.g. Reaky ReLU, sigmoid, tanh networks).
- We can understand the effect of activation functions by calculating the upper bound of $\|K_{\sigma_l}\|$.
  - Leaky ReLU: independent of $\|W_j\|$
  - sigmoid, tanh: $\mathcal{X}_j$ depends on $\|W_j\|$

    $\rightarrow \|K_{\sigma_j}\|$ grows as $\|W_j\|$ becomes large.