Machine Learning Homework 2

ID: 112652011

> Reading and Explaining Lemmas
>
> Your task is to read the following paper:
>
> Ryck et al., On the approximation of functions by tanh neural networks
>
> Focus on Lemma 3.1 and Lemma 3.2.

Lemma 3.1

Let $k \in \mathrm{N}_0$ and $s \in 2\mathrm{N}-1$. Then it holds that for all $\epsilon > 0$ there exists a shallow tanh neural network $\Psi_{s,\epsilon} : [-M, M] \to \mathrm{R}^{\frac{s+1}{2}}$ of width $\frac{s+1}{2}$ such that

$$\max_{p \leq s, p \text{ odd}} \left\| f_p - (\Psi_{s,\epsilon})_{\frac{p+1}{2}} \right\|_{W^{k,\infty}} \leq \epsilon$$

Moreover, the weights of $\Psi_{s,\epsilon}$ scale as $O\left(\epsilon^{-s/2}(2(s+2)\sqrt{2M})^{s(s+3)}\right)$ for small ϵ and large s .

Explanation

This lemma states that for any odd power monomials (i.e. $y^1, y^3, y^5, \ldots$), we are able to construct a tanh neural network with single hidden layer (**shallow**) to approximate the function y^p with an error less than ϵ in the $W^{k,\infty}$ norm, with the width of $\frac{s+1}{2}$ (the # of neurons in the hidden layer).

The reason why this works is because $tanh()$ is an odd function. Its Taylor series expansion around zero contains only odd-power terms. The paper use some method to isolate and scale these derivatives, eventually creating an approximation of the monomial y^p.

Meaning of symbols

- p : The power of the monomial. In this lemma, p is an odd integer.
- s : The highest power of the monomial we want to approximate. ◦
- $f_p(y) := y^p$: The object function.
- $\Psi_{s,\epsilon}$: The shallow N.N. we constructed. It is a multi-output network designed to approximate all odd-degree monomials from power 1 to s
- $(\Psi_{s,\epsilon})_{\frac{p+1}{2}}$: The $\frac{p+1}{2}$-th output of the neural network $\Psi_{s,\epsilon}$. This index maps the sequence of odd powers (1, 3, 5, ...) to a sequence of consecutive integer indices (1, 2, 3, ...). For example, the 1st output approximates y^1, the 2nd output approximates y^3, and so on.
- $\|\cdot\|_{W^{k,\infty}}$: **Sobolev norm**. It measures the maximum error not only for the function itself but also for all its derivatives up to order k.
- ϵ : Maximum acceptable error
- $\max_{p \leq s,\, p \text{ odd}}$: It takes the maximum error across all the approximations for the odd monomials from y^1 up to y^s.

Lemma 3.2

Let $k \in \mathrm{N}_0$, $s \in 2\mathrm{N}-1$ and $M > 0$. For every $\epsilon > 0$ there exists a shallow tanh neural network $\Psi_{s,\epsilon} : [-M, M] \to \mathrm{R}^s$ of width $\frac{3(s+1)}{2}$ such that

$$\max_{p \leq s} \left\| f_p - (\Psi_{s,\epsilon})_p \right\|_{W^{k,\infty}} \leq \epsilon$$

Explanation

This lemma extends the result of Lemma 3.1 to cover all integer-power monomials (both even and odd) up to a degree s. It proves that a shallow `tanh` network can still approximate any monomial y^p (for $p \leq s$) to an arbitrary precision ϵ in the strong $W^{k,\infty}$ norm.

The main challenge is that `tanh` is an odd function, so it cannot directly represent even functions like y^2 or y^4. The authors overcome this with a clever **recursive construction**. The core idea is based on an algebraic identity (Eq. 25 in the paper) which expresses an even-degree monomial y^{2n} in terms of:

1. Two **odd-degree** monomials: $(y+\alpha)^{2n+1}$ and $(y-\alpha)^{2n+1}$.
2. A series of **lower-order even-degree** monomials: y^{2k} for $k < n$.

This creates a recursive path.

For example,

- If we want to approximate y^4, we need to approximate $(y+\alpha)^5, (y-\alpha)^5$ and y^0, y^2
- If we want to approximate y^2, we need to approximate $(y+\alpha)^3, (y-\alpha)^3$ and y^0

Since we are already able to approximate the odd-degree function, so we can approximate the even-degree function, too.