Deep Neural Networks with Trainable Activations and Controlled Lipschitz Constant

EE5180 Project Midterm Presentation

Group 21

EE21D002, EE21D005, EE21M010, EE21S056
Department of Electrical Engineering
IIT Madras

Overview

Introduction

- A **variational framework** to learn the activation functions of deep neural networks is introduced.
- The aim of the paper is to increase the **capacity** of network while controlling the **Lipschitz bound** of the network.
- The **capacity** of neural networks is given by $\log_2(|A|)$ where $A(n_1, n_2, \ldots, n_l)$ is a feedforward, layered, fully connected network.
- The goal of **supervised learning** is to approximate an unknown mapping from a set of noisy samples.

Introduction

- In supervised learning, we find the function $f : \mathbb{R}^n \to \mathbb{R}^d$ which gives $y_m \approx f(x_m)$ where (x_m, y_m) are training samples for $m = 1, 2, \ldots, M$.
- In the scalar case where $d = 1$ a classical formulation of the problem

$$\min_{f \in \mathcal{H}(\mathbb{R}^d)} \left(\sum_{m=1}^{M} \mathbf{E}(y_m, f(x_m)) + \lambda \|f\|_{\mathcal{H}}^2 \right) \tag{1}$$

- Although the problem (1) is **infinite dimensional**, the kernel representer theorem states that the the solution is unique and has the form

$$f(\mathbf{x}) = \sum_{m=1}^{M} a_m \mathrm{k}(\mathbf{x}, \mathbf{x}_m) \tag{2}$$

where $\mathrm{k}(\cdot, \cdot)$ is the unique reproducing kernel.

Introduction

- Recently **Deep Learning** has been outperforming the kernel methods with applications such as image classification, inverse problems and segmentation.
- A deep neural network is a repeated composition of affine mappings and pointwise **non-linearities** (Activation functions).
- The classical choice to an activation function is **sigmoid** but it suffers from vanishing gradients.
- The currently preferred activation functions are **ReLU**$=max(x,0)$ and its variants such as **Leaky ReLU** $=max(x,ax)$ where $a \in (0,1)$ and **PReLU**.

Introduction

- A ReLU can be interpreted as **Linear spline** with one knot. It has been shown that Linear spline are *maximally regularized*.
- Although ReLU networks are favourable, one may want to learn the activation functions.
- The closest attempt to that is the PReLU where we learn '**a**', a parameter in this particular activation function.
- The **Lipschitz regularity** is of great importance for the stability of deep neural networks.

Notion of TV norm and $\mathrm{BV}^2(\mathbb{R})$

- The space of functions with **second-order bounded variations** is $\mathrm{BV}^{(2)}(\mathbb{R})$ and is defined as

$$\mathrm{BV}^{(2)}(\mathbb{R}) = \{f \in \mathcal{S}'(\mathbb{R}) : \|\mathrm{D}^2 f\|_{\mathcal{M}} < \infty\} \tag{3}$$

  where
  $\mathcal{S}'(\mathbb{R})$ is the space of tempered distributions,
  $\mathrm{D} : \mathcal{S}'(\mathbb{R}) \to \mathcal{S}'(\mathbb{R})$ is the generalized derivative operator and,
  $\mathrm{TV}^{(2)}(f) \triangleq \|\mathrm{D}^2 f\|_{\mathcal{M}}$ is the second-order total variation norm.
- However, $\mathrm{TV}^{(2)}(f)$ is a semi-norm which makes $\mathrm{BV}^{(2)}(\mathbb{R})$ ineligible to be a *Banach space*.

Lipschitz Continuity

- To define **Lipschitz continuity**, the space defined in equation (3) has to be a *normed* space. To that end, define the $\mathrm{BV}^{(2)}$ norm

$$\|f\|_{\mathrm{BV}^{(2)}} \triangleq \mathrm{TV}^{(2)}(f) + |f(0)| + |f(1)| \tag{4}$$

Lipschitz Continuity (for generic Banach spaces)

Given generic Banach spaces $(\mathcal{X}, \|\cdot\|_{\mathcal{X}})$ and $(\mathcal{Y}, \|\cdot\|_{\mathcal{Y}})$, a function $f : \mathcal{X} \to \mathcal{Y}$ is said to be *Lipschitz-continuous* if there exists a finite constant $C > 0$ such that

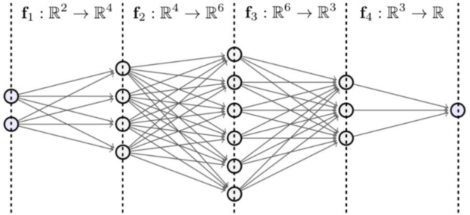
$$\|f(x_1) - f(x_2)\|_{\mathcal{Y}} \leq C\|x_1 - x_2\|_{\mathcal{X}}, \forall x_1, x_2 \in \mathcal{X} \tag{5}$$

The minimal value of C is called the **Lipschitz constant** of f.

Input-output relation for a DNN

An L-layer neural network can be characterized by the function

$$f_{deep} : \mathbb{R}^{N_0} \to \mathbb{R}^{N_L} : \mathbf{x} \mapsto f_L \circ \cdots f_1(\mathbf{x}) \tag{6}$$



$$\mathbf{f}_{\text{deep}} = \mathbf{f}_4 \circ \mathbf{f}_3 \circ \mathbf{f}_2 \circ \mathbf{f}_1 : \mathbb{R}^2 \to \mathbb{R}$$

Figure: Schematic view of an example neural network with layer descriptor (2,4,6,3,1)

Formulating f_{deep} and $(\mathrm{BV}^{(2)}, p)$-norm

- f_l represents the l^{th} layer of the neural network f_{deep} and can be formulated as

$$f_l(\mathbf{x}) = \left(\sigma_{1,l}(\mathbf{w}_{1,l}^T\mathbf{x}), \sigma_{2,l}(\mathbf{w}_{2,l}^T\mathbf{x}), \ldots, \sigma_{N_l,l}(\mathbf{w}_{N_l,l}^T\mathbf{x})\right) \tag{7}$$

  where $\mathbf{w}_{n,l} \in \mathbb{R}^{N_{l-1}}$ are the **weight** vectors and $\sigma_{n,l} : \mathbb{R} \to \mathbb{R}$ are **activation** functions for $n = 1, 2, ..., N_l$.
- Alternatively, we can have matrix $\mathbf{W}_l = \begin{bmatrix}\mathbf{w}_{1,l} & \mathbf{w}_{2,l} & \ldots & \mathbf{w}_{N_l,l}\end{bmatrix}$ and vector $\boldsymbol{\sigma}_l : \mathbb{R}^{N_l} \to \mathbb{R}^{N_l}$ such that $f_l = \boldsymbol{\sigma}_l \circ \mathbf{W}_l$.
- We can finally define $(\mathrm{BV}^{(2)}, p)$-norm $\forall p \in [1, +\infty)$ of the nonlinear layer $\boldsymbol{\sigma}_l$ as

$$\|\boldsymbol{\sigma}_l\|_{(\mathrm{BV}^{(2)},p)} = \left(\sum_{n=1}^{N_l} \|\sigma_{n,l}\|_{\mathrm{BV}^{(2)}}^p\right)^{\frac{1}{p}} \tag{8}$$

Second Order Bounded Variation Activations

- Activation functions are selected from $\mathrm{BV}^{(2)}(\mathbb{R})$
- **Key feature** - Lipschitz continuity

Proposition 1

Any function with second order bounded variations is Lipschitz - continuous. For any function $\sigma \in \mathrm{BV}^{(2)}(\mathbb{R})$ and $x_1, x_2 \in \mathbb{R}$

$$|\sigma(x_1) - \sigma(x_2)| \leq \| \sigma \|_{\mathrm{BV}^{(2)}} |x_1 - x_2| \tag{9}$$

Proposition 2

For any function $\sigma \in \mathrm{BV}^{(2)}(\mathbb{R})$ and $x_0 \in \mathbb{R}$, the left and right derivatives of σ at the point $x = x_0$ exist and are finite.

Second Order Bounded Variation Activations

Theorem (1)

Any feedforward fully connected deep neural network $f_{deep} : \mathbb{R}^{N_0} \longrightarrow \mathbb{R}^{N_L}$ *with second order bounded variation activations* $\sigma_{n,l} \in BV^{(2)}(\mathbb{R})$ *is Lipschitz continuous. If we consider* ℓ_p *for* $p \in [1, \infty]$ *topology in the input and output spaces, the neural network satisfies the global Lipschitz bound,*

$$\| f_{deep}(x_1) - f_{deep}(x_2) \|_p \leq C \| x_1 - x_2 \|_p, \forall x_1, x_2 \in \mathbb{R}^{N_0} \tag{10}$$

where,

$$C = \left(\prod_{l=1}^{L} \| W_l \|_{q,\infty} \right) \left(\prod_{l=1}^{L} \| \sigma_l \|_{BV^{(2)},p} \right) \tag{11}$$

$$q \in [1, \infty]; 1/p + 1/q = 1, \| W_l \|_{q,\infty} = max_n \| W_{n,l} \|_q \tag{12}$$

is the mixed norm $\ell_q - \ell_\infty$ *of the* l^{th} *linear layer*

Second Order Bounded Variation Activations

- An alternative bound for the Lipschitz constant of the neural network is obtained when the *standard Euclidean topology* is assumed for the input and output spaces.

Proposition 3

Let $f_{deep} : \mathbb{R}^{N_0} \longrightarrow \mathbb{R}^{N_L}$ be a fully connected feedforward neural network with activations selected from $\mathrm{BV}^{(2)}(\mathbb{R})$. For all $x_1, x_2 \in \mathbb{R}^{N_0}$ we have that

$$\| f_{deep}(x_1) - f_{deep}(x_2) \|_2 \leq C_E \| x_1 - x_2 \|_2, \forall x_1, x_2 \in \mathbb{R}^{N_0} \tag{13}$$

where,

$$C_E = \left(\prod_{l=1}^{L} \| W_l \|_F \right) \left(\prod_{l=1}^{L} \| \sigma_l \|_{\mathrm{BV}^{(2)},1} \right) \tag{14}$$

Learning Activations

- The (**weak***) **continuity** of the sampling functional is needed to guarantee the well-posedness of the learning problem.

Theorem (2)

For any $x_0 \in \mathbb{R}^{N_0}$, the sampling functional δ_{x_0}: $f_{deep} \mapsto f_{deep}(x_0)$ is weak*-continuous in the space of neural networks with second-order bounded-variation activations.

- For a dual pair $(\mathcal{X}, \mathcal{X}')$ of Banach spaces, the sequence $\{\omega_n\}_{n\in\mathbb{N}} \in \mathcal{X}'$ converges in the weak*-topology to $\omega_{lim} \in \mathcal{X}'$ if, for any element $\varphi \in \mathcal{X}$, we have that

$$\langle \omega_n, \varphi \rangle \to \langle \omega_{lim}, \varphi \rangle, \; n \to +\infty \tag{15}$$

- Consequently, a functional $\nu : \mathcal{X}' \to \mathbb{R}$ is weak*-continuous if $\nu(\omega_n) \to \nu(\omega_{lim})$ for any sequence $\{\omega_n\}_{n\in\mathbb{N}} \in \mathcal{X}'$ that converges in the weak*-topology to ω_{lim}.

Learning Activations

- Given the data-set (X,Y) of size M that consists in the pairs $(x_m, y_m) \in \mathbb{R}^{N_0} \times \mathbb{R}^{N_L}$ for $m = 1, 2, \ldots, M$, we then consider the following **cost functional**

$$\mathcal{J}(f_{deep}; X, Y) = \sum_{m=1}^{M} E(y_m, f_{deep}(x_m)) + \sum_{l=1}^{L} \mu_l R_l(\mathbf{W}_l) + \sum_{l=1}^{L} \lambda_l ||\boldsymbol{\sigma}_l||_{\mathrm{BV}^{(2)},1} \quad (16)$$

  where f_{deep}, $\mathbf{W}_l$, $\boldsymbol{\sigma}_l = (\sigma_{1,l}, \ldots, \sigma_{N_l,l})$, and $E(\cdot,\cdot)$ have their usual meanings, and $R_l : \mathbb{R}^{N_l} \times \mathbb{R}^{N_{l-1}} \to \mathbb{R}$ is a **regularization functional** for the linear weights of the lth layer. The positive constants $\mu_l, \lambda_l > 0$ balance the regularization effect in the training step.
- Standard choice for weight regularization is the **Frobenius norm** $R(\mathbf{W}) = ||\mathbf{W}||_F^2$.
- Under some natural conditions, there always exists a solution of (16) with **continuous piecewise-linear** activation functions, which we refer to as a **deep-spline** neural network.

Learning Activations

Theorem (3)

Consider the training of a deep neural network via the minimization

$$\min_{\substack{\mathbf{w}_{n,l}\in\mathbb{R}^{N_{l-1}},\\ \sigma_{n,l}\in BV^{(2)}(\mathbb{R})}} \mathcal{J}(f_{deep}; X, Y) \tag{17}$$

If we assume our loss function $E(\cdot,\cdot)$ to be proper, lower semi-continuous, and coercive and the regularization functionals R_l to be continuous and coercive, then there always exists a solution f^*_{deep} of (16) with activations $\sigma_{n,l}$ of the form

$$\sigma_{n,l}(x) = \sum_{k=1}^{K_{n,l}} a_{n,l,k} ReLU(x - \tau_{n,l,k}) + b_{1,n,l}x + b_{2,n,l}, \tag{18}$$

where $K_{n,l} \leq M$ and $a_{n,l,k}$, $\tau_{n,l,k}$, $b_{\cdot,n,l} \in \mathbb{R}$ are adaptive parameters.

Learning Activations

- Theorem 3 suggests an optimal **ReLU-based parametric** to learn activations.
- This property translates the original **infinite-dimensional** problem (17) into a **finite-dimensional** parametric optimization, where one only needs to determine the ReLU weights $\mathrm{a}_{n,l,k}$, the positions $\tau_{n,l,k}$, and the affine terms $\mathrm{b}_{1,n,l}$, $\mathrm{b}_{2,n,l}$.
- One of the key differences between this work and *"A representer theorem for deep neural networks"* of M.Unser lies in the choice of **Regularization**.
- It is the **BV$^{(2)}$-regularization** that allows us to obtain the **global bound** for the **Lipschitz constant** of the network, unlike the framework of the other work, which relies on the **TV$^{(2)}$-regularization** (semi-norm).
- But the catch here is that, in Unser's work, the activations have at most (M-2) **knots**, as opposed to our case where $K_{n,l} \leq M$.

Learning Activations

- Another interesting property governs the **energy relationship** between the consecutive linear and nonlinear layers.

Theorem (4)

For any local minima of the minimization problem (17) with linear weights $\mathbf{W}_l$ and nonlinear layers $\boldsymbol{\sigma}_l$, we have that

$$\lambda_l \|\boldsymbol{\sigma}_l\|_{BV^{(2)},1} = 2\mu_{l+1}\|\mathbf{W}_{l+1}\|_F^2, \quad l = 1, 2, \ldots, L-1 \tag{19}$$

- This clearly shows that the regularization constants μ_l and λ_l provide a **balance** between the linear and nonlinear layers.
- The authors have used the above relation to determine the value of λ_l, thereby reducing the number of hyper-parameters to be tuned and resulting in a faster training scheme.

Final Optimization Problem

- Optimization problem (17) can now be expanded using equations (4), (16) and definition of **Frobenius norm** as follows

$$\min_{\substack{\mathbf{w}_{n,l}\in\mathbb{R}^{N_{l-1}},\\ \mathbf{a}_{n,l}\in\mathbb{R}^{K_{n,l}},\\ b_{i,n,l}\in\mathbb{R}}} \sum_{m=1}^{M} \mathbf{E}(\mathbf{y}_m, f_{deep}(\mathbf{x}_m)) + \sum_{l=1}^{L} \mu_l \sum_{n=1}^{N_l} \|\mathbf{w}_{n,l}\|_2^2$$
$$+ \sum_{l=1}^{L} \lambda_l \sum_{n=1}^{N_l} \left(\|\mathbf{a}_{n,l}\|_1 + |\sigma_{n,l}(1)| + |\sigma_{n,l}(0)|\right) \quad (20)$$

- (20) is the final optimization problem that is to be solved, however the parameter $K_{n,l}$ still needs to be fixed before initialization. The authors of this paper have set it to $K = 21$ for the experimental setup that is explained shortly after.

Experimental Setup

- To verify the validity of the framework described precedingly, an experiment is setup as follows —classify points inside a circle with area 2 centered at the origin in a two dimensional space.
- Thus, the target function is

$$\mathbb{1}_{\text{Circle}}(x_1, x_2) = \begin{cases} 1, & x_1^2 + x_2^2 \leq \frac{2}{\pi} \\ 0, & \text{otherwise.} \end{cases} \tag{21}$$

- The training dataset is generated with $M = 1000$ random points with uniform distribution in $[-1, 1]^2$.
- Neural networks of the form $(2, 2W, 1)$, where W is the width parameter of the hidden layer are used. Specifically, for the last layer **sigmoid** activation is used with **binary cross-entropy loss**.

Results

Subsequently after the experimental setup, number of experiments are performed to test the claims described in the paper and are verified quantifiably. These experiments are listed below —

1. *Comparison with ReLU-based Activations*
2. *Sparsity-Promoting Effect of* $\mathrm{BV}^{(2)}$*-Regularization*
3. *Effect of the Parameter* λ
4. ℓ_1 *versus* ℓ_2 *Outer-Norms*
5. *Effect of the Parameter K*

	Architecture	N_{param}	**Performance**
ReLU	(2,10,1)	41	98.15
LeakyReLU	(2,10,1)	41	98.12
PReLU	(2,10,1)	51	98.19
Deep Lipschitz	(2,2,1)	**23**	**98.54**

Table: Number of parameters and Performance in the Area Classification Experiment

Plans for the final presentation

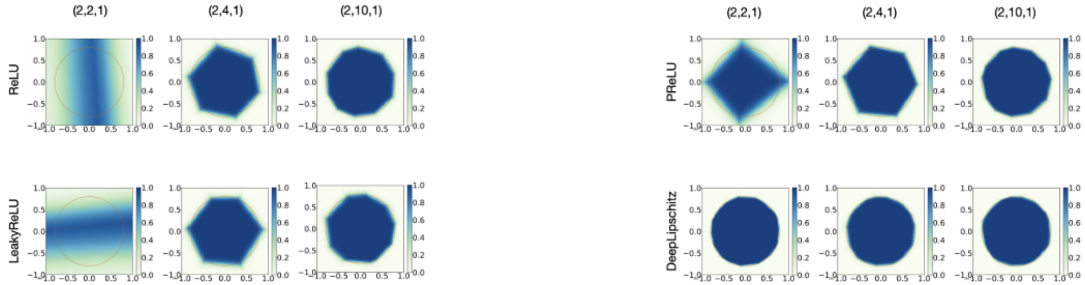


Figure: Visual Comparison of performance of different activation functions

For the final presentation, we plan to simulate all of the above experiments using **Python** and validate the results shown in the paper.

The End