Deep Neural Networks with Trainable Activations and Controlled Lipschitz Constant

EE5180 Project Final 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 **functional capacity** of neural networks is defined as the class of functions it can compute as its weights are varied.
- The classical formulation of the **supervised** learning problem is

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

Introduction

- 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.
- 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).

Introduction

- The activation function can be chosen as **sigmoid** function $h(x) = \frac{1}{1+e^{-x}}$, **ReLU**=max(x,0),**Leaky ReLU**=max(x,ax) where a$\in$(0,1) and **PReLU**
- 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 and the closest attempt is **PReLU**
- The **trainable activation** functions proposed can be used to replace classical activations such as **ReLU**

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

- 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(does not grow too fast),
  $\mathrm{D} : \mathcal{S}'(\mathbb{R}) \rightarrow \mathcal{S}'(\mathbb{R})$ is the generalized derivative operator and,
  $\mathrm{TV}^{(2)}(f) \triangleq \|\mathrm{D}^2 f\|_{\mathcal{M}}$ is the second-order total variation norm which is a semi norm.
- So we define $\mathrm{BV}^{(2)}$ norm to make it a *Banach space*.

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

Lipschitz continuity

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

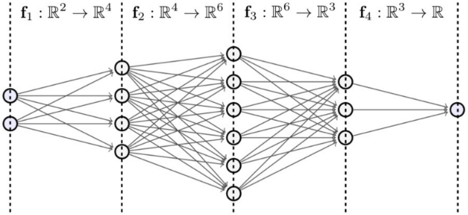
$$\|f(x_1) - f(x_2)\|_{\mathcal{Y}} \leq C\|x_1 - x_2\|_{\mathcal{X}}, \forall x_1, x_2 \in \mathcal{X} \tag{5}$$

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

Input-output relation for a DNN

- An L-layer neural network with layer descriptor$(N_0, N_1, \ldots, N_L)$ can be characterized as

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



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

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

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

  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 $\mathbf{W}_l = \begin{bmatrix} \mathbf{w}_{1,l} & \mathbf{w}_{2,l} & \dots & \mathbf{w}_{N_l,l} \end{bmatrix}$ 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 from the **space of second order bounded variation functions** $\mathrm{BV}^{(2)}(\mathbb{R})$ are selected.
- **Key Feature** : Lipschitz Continuity
- This feature ensures that the activation function is *continuous* and *differentiable* everywhere.
- Moreover, it is shown that any function in $\mathrm{BV}^{(2)}(\mathbb{R})$ has well defined left and right derivatives at any point which is an important requirement for an activation function for using the *back-propagation* scheme in the training step.

Second Order Bounded Variation Activations

- In the paper, it is shown that any neural network with activations from $\mathrm{BV}^{(2)}(\mathbb{R})$ specifies a Lipschitz-continuous input-output relation along with an upper-bound for its Lipschitz constant. The relationship and upper-bound are shown below.

$$\| f_{deep}(x_1) - f_{deep}(x_2) \|_p \leq C \| x_1 - x_2 \|_p, \forall x_1, x_2 \in \mathbb{R}^{N_0} \tag{9}$$

where,

$$C = \left(\prod_{l=1}^{L} \| W_l \|_{q,\infty} \right) \left(\prod_{l=1}^{L} \| \sigma_l \|_{\mathrm{BV}^{(2)},p} \right) \tag{10}$$

$$q \in [1, \infty]; 1/p + 1/q = 1, \| W_l \|_{q,\infty} = max_n \| W_{n,l} \|_q \tag{11}$$

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

Learning Activations

- Now, a variational framework to learn Lipschitz activations in a deep neural network is formulated. The search space is limited to $\mathrm{BV}^{(2)}(\mathbb{R})$ to ensure the Lipschitz continuity of the input-output relation of the global network.
- 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**

$$\mathcal{J}(f_{deep}; X, Y) = \sum_{m=1}^{M} E(y_m, f_{deep}(x_m)) + \sum_{l=1}^{L} \mu_l R_l(\mathbf{W}_l) + \sum_{l=1}^{L} \lambda_l ||\boldsymbol{\sigma}_l||_{\mathrm{BV}^{(2)},1} \qquad (12)$$

  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** $R(\mathbf{W}) = ||\mathbf{W}||_F^2$.

Learning Activations

- Training of the neural network is done via the minimization

$$\min_{\substack{\mathbf{w}_{n,l} \in \mathbb{R}^{N_{l-1}},\\ \sigma_{n,l} \in BV^{(2)}(\mathbb{R})}} \mathcal{J}(f_{deep}; X, Y) \tag{13}$$

- It is then shown that there always exists a solution f^*_{deep} of (13) with activations $\sigma_{n,l}$ of the form

$$\sigma_{n,l}(x) = \sum_{k=1}^{K_{n,l}} a_{n,l,k} ReLU(x - \tau_{n,l,k}) + b_{1,n,l}x + b_{2,n,l}, \tag{14}$$

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

- In (14), an optimal ReLU-based parametric is shown to learn activations. Thus, the **original infinite-dimensional problem** (13) reduces to a **finite-dimensional parametric optimization** where only finite number of parameters are required.
- In practice, **regularization constant** λ_l restrains $b_{1,n,l}$ and $b_{2,n,l}$ from taking larger values, this in turn helps us control the Lipschitz regularity of the network.
- $\mathrm{BV}^{(2)}(\mathbb{R})$-regularization imposes an ℓ_1 penalty on the ReLU weights thus **promoting sparsity** reducing the number of knots significantly as compared to its upper bound.

Experimental Setup

- The goal of the experiment is to classify points that are inside a circle of area 2 centred at the origin.
- Thus, the target function is

$$\mathbb{1}_{\mathrm{Circle}}(x_1, x_2) = \begin{cases} 1, & x_1^2 + x_2^2 \leq \frac{2}{\pi} \\ 0, & \text{otherwise.} \end{cases} \tag{15}$$

- The training dataset is generated with $M = 1000$ random points with uniform distribution in $[-1, 1]^2$.
- Fully connected 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**.

Experimental Setup

- For the activations, we consider the simple piecewise-linear functions **absolute value** and **soft thresholding**, defined as

$$f_{abs}(x) = \begin{cases} x, & x \geq 0 \\ -x, & x < 0, \end{cases} \tag{16}$$

$$f_{soft}(x) = \begin{cases} x - \frac{1}{2}, & x \geq \frac{1}{2} \\ 0, & x \in (-\frac{1}{2}, \frac{1}{2}) \\ x + \frac{1}{2}, & x \leq -\frac{1}{2}. \end{cases} \tag{17}$$

- Initializing half of the activations with f_{abs} and the other half with f_{soft} allows the network to be flexible to both even and odd functions.

Comparison with ReLU-Based Activations

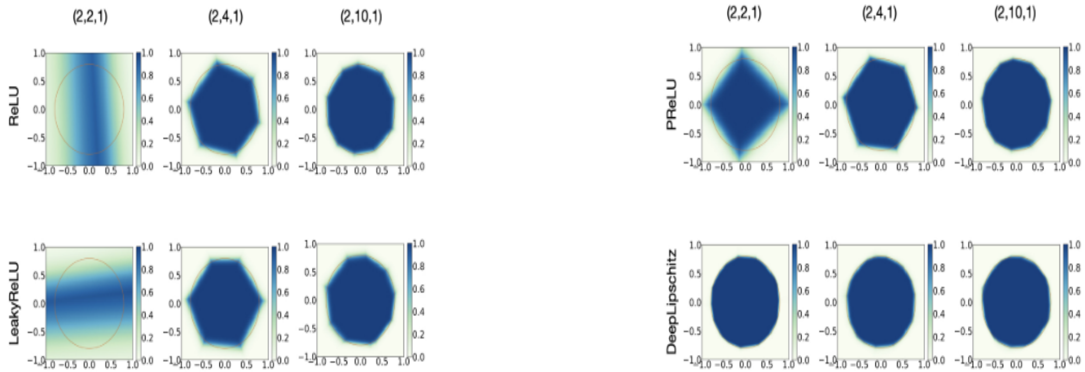


Figure: Comparison of performance of different activations. In each case, we consider $W = 1, 2, 5$ hidden neurons.

Comparison with ReLU-Based Activations

	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

- **Deep Lipschitz**, already in the simplest configuration with layer descriptor (2,2,1), outperforms all other methods, even when they are deployed over the richer architecture (2,10,1).
- Secondly, there are fewer number of parameters for this scheme.

Comparison with ReLU-Based Activations

- In the minimal case $W = 1$, the network is expected to learn parabola-type activations since the target function can also be represented as

$$\mathbb{1}_{\mathrm{Circle}}(x_1, x_2) = \mathbb{1}_{[0,2\pi]}(x_1^2 + x_2^2), \tag{18}$$

which is the composition of the sum of two parabolas and a threshold function.

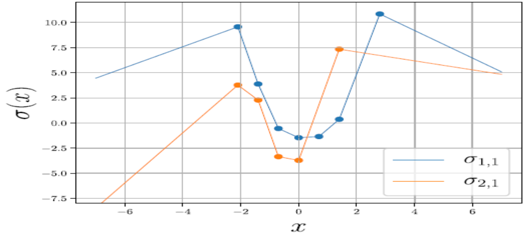


Figure: Learned activations for a simple network with layer descriptor (2,2,1).

Sparsity-Promoting Effect of $BV^{(2)}$-Regularization

- Irrespective of the large number of ReLUs in the actual expansion of the activations, the learned activations have only a **sparse expansion** in the ReLU basis.
- The $BV^{(2)}$-regularization imposes an ℓ_1 **penalty** on the ReLU weights in the expansion, thus promoting **sparsity**!

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

where $\mathrm{K}_{n,l} \leq \mathrm{M}$ and $\mathrm{a}_{n,l,k}$, $\tau_{n,l,k}$, $\mathrm{b}_{\cdot,n,l} \in \mathbb{R}$ are adaptive parameters.
- The $BV^{(2)}$ norm of the activation is given by

$$\|\sigma_{n,l}\|_{BV^{(2)}} = \|\mathbf{a}_{n,l}\|_1 + |\sigma(1)| + |\sigma(0)|,$$

where $\mathbf{a}_{n,l} = (\mathrm{a}_{n,l,1}, \ldots, \mathrm{a}_{n,l,K_{n,l}})$ is the vector of ReLU coefficients.

Effect of the Parameter λ

- Decay parameter μ is set to 10^{-4} with layer descriptor (2,2,1)
- For error rate, a transition occurs as λ varies
- We can have a proper range of λ values, less than 10^{-3}

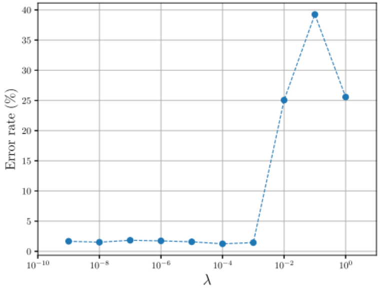
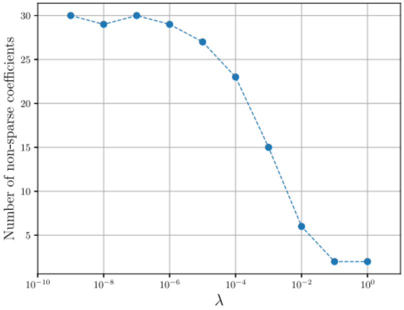


Figure: Effect of λ on Error Rate

Effect of the Parameter λ

- The sparsity and Lipschitz regularity of the network increases with λ
- Best choice - $\lambda = 10^{-3}$
- With $\mu = 10^{-4}$, $\lambda = \frac{16}{11(2W+1)^{\mu}} = 0.5*10^{-5}$



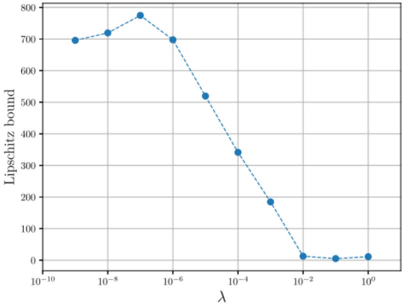


Figure: Effect of λ on Lipschitz bound and Number of nonzero ReLU coefficients

ℓ_1 Versus ℓ_2 Outer Norms

- We have

$$C_E = \left(\prod_{l=1}^{L} \| W_l \|_F \right) \left(\prod_{l=1}^{L} \| \sigma_l \|_{\mathrm{BV}^{(2)},1} \right) \tag{19}$$

- (2,10,1) Network is trained with two outer norms.
- ℓ_1 outer norm results fewer parameters, due to its global sparsifying effect.

Outer norm	N_{param}	**Performance**
ℓ_1	66	98.61
ℓ_2	89	98.39

Table: Effect of the ℓ_1 Vs ℓ_2 Outer Norms

Effect of the Parameter K

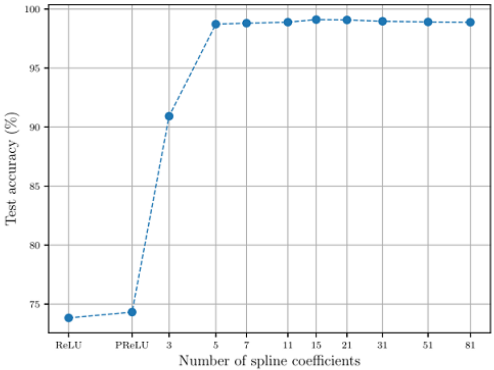


Figure: Performance versus the number K of spline knots of each activation functions.

Conclusion

- A variational framework is introduced to learn the activations of a deep neural network while controlling its global Lipschitz regularity.
- Provided a global bound for Lipschitz constants of neural networks with second-order bounded-variation activations.
- Solution for variational problem exists and is in the form of a deep-spline network with continuous piecewise linear activation functions.
- In this paper, a more *complex* activation function is deployed to simplify the architecture of the neural network to a great extent.

The End