



HAL
open science

Dichotomize and Generalize: PAC-Bayesian Binary Activated Deep Neural Networks

Gaël Letarte, Pascal Germain, Benjamin Guedj, François Laviolette

► To cite this version:

Gaël Letarte, Pascal Germain, Benjamin Guedj, François Laviolette. Dichotomize and Generalize: PAC-Bayesian Binary Activated Deep Neural Networks. ML with guarantees – NeurIPS 2019 workshop, Dec 2019, Vancouver, Canada. ⟨hal-02482354⟩

HAL Id: hal-02482354

<https://inria.hal.science/hal-02482354v1>

Submitted on 18 Feb 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



HAL Authorization

Dichotomize and Generalize: PAC-Bayesian Binary Activated Deep Neural Networks

Gaël Letarte¹, Pascal Germain², Benjamin Guedj^{2,3}, François Laviolette¹

¹ Département d'informatique et de génie logiciel, Université Laval, Québec, Canada

² Équipe-projet Modal, Inria Lille - Nord Europe, Villeneuve d'Ascq, France

³ UCL Centre for Artificial Intelligence, University College London, London, England



Introduction

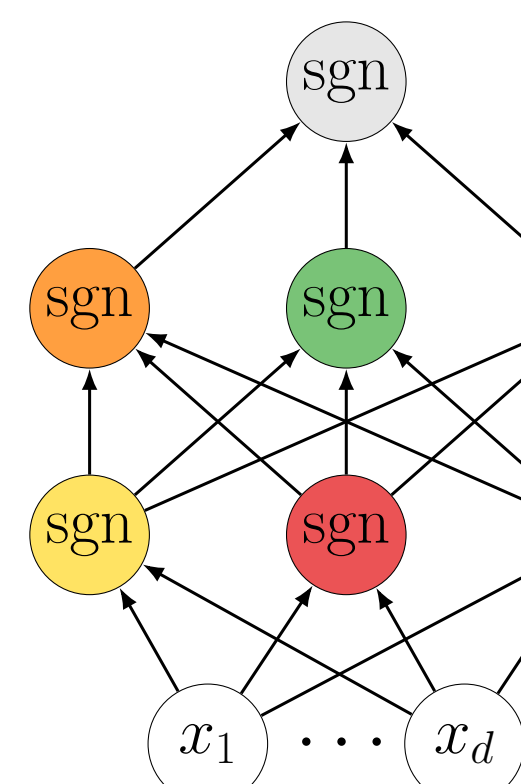
We present a comprehensive study of multilayer neural networks with binary activation, relying on the PAC-Bayesian theory.

Contributions

- ▶ An end-to-end framework to train a binary activated deep neural network (DNN).
- ▶ Nonvacuous PAC-Bayesian generalization bounds for binary activated DNNs.

Binary Activated Neural Networks

- ▶ L fully connected layers
- ▶ d_k denotes the number of neurons of the k^{th} layer
- ▶ $\text{sgn}(a) = 1$ if $a > 0$ and $\text{sgn}(a) = -1$ otherwise
- ▶ Weights matrices : $\mathbf{W}_k \in \mathbb{R}^{d_k \times d_{k-1}}$, $\theta = \text{vec}(\{\mathbf{W}_k\}_{k=1}^L) \in \mathbb{R}^D$



$$f_\theta(\mathbf{x}) = \text{sgn}(\mathbf{w}_L \text{sgn}(\mathbf{W}_{L-1} \text{sgn}(\dots \text{sgn}(\mathbf{W}_1 \mathbf{x}))))$$

PAC-Bayesian Theory

Given a data distribution \mathcal{D} , a training set $S = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n)\} \sim \mathcal{D}^n$, with $\mathbf{x}_i \in \mathbb{R}^{d_0}$ and $y_i \in \{-1, 1\}$, a loss $\ell : [-1, 1]^2 \rightarrow [0, 1]$, a predictor $f \in \mathcal{F}$:

$$\mathcal{L}_{\mathcal{D}}(f) = \mathbb{E}_{(\mathbf{x}, y) \sim \mathcal{D}} \ell(f(\mathbf{x}), y) \quad \leftarrow \text{generalization loss}$$

$$\widehat{\mathcal{L}}_S(f) = \frac{1}{n} \sum_{i=1}^n \ell(f(\mathbf{x}_i), y_i) \quad \leftarrow \text{empirical loss}$$

PAC-Bayesian Theorem

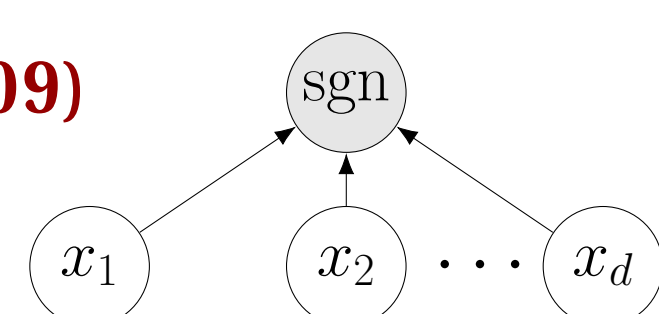
For any prior P on \mathcal{F} , with probability $1 - \delta$ on the choice of $S \sim \mathcal{D}^n$, we have for all $C > 0$, and any posterior distribution Q on \mathcal{F} :

$$\mathbb{E}_{f \sim Q} \mathcal{L}_{\mathcal{D}}(f) \leq \frac{1}{1 - e^{-C}} \left(1 - \exp \left(-C \mathbb{E}_{f \sim Q} \widehat{\mathcal{L}}_S(f) - \frac{1}{n} [\text{KL}(Q \| P) + \ln \frac{2\sqrt{n}}{\delta}] \right) \right)$$

Linear Classifier

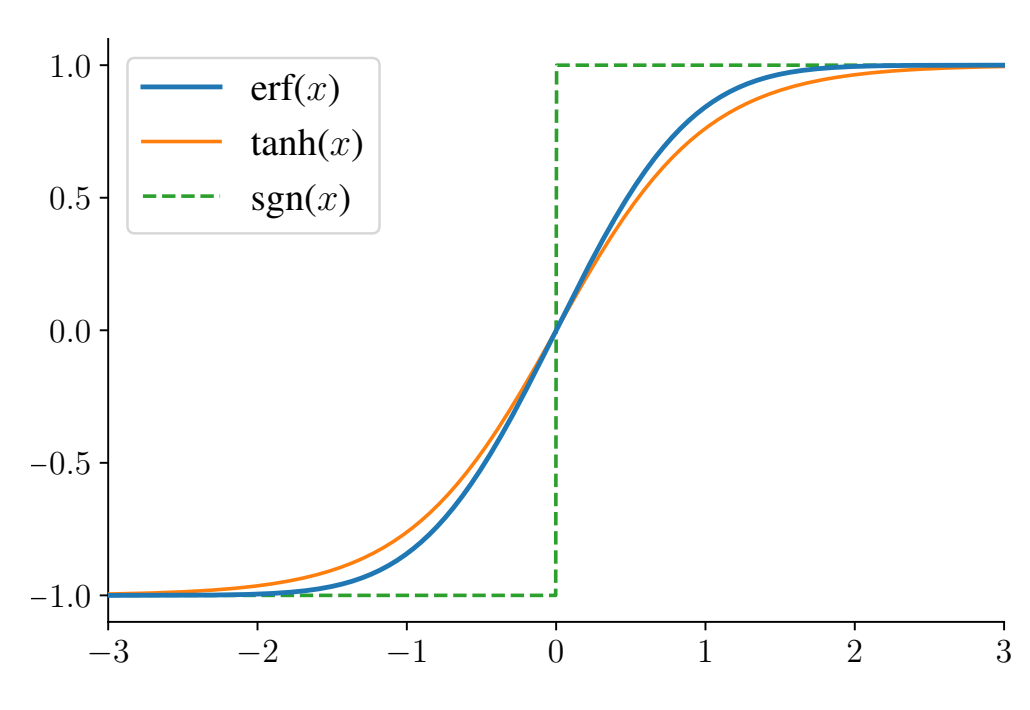
PAC-Bayesian Learning of Linear Classifiers (Germain et al., 2009)

$$f_{\mathbf{w}}(\mathbf{x}) := \text{sgn}(\mathbf{w} \cdot \mathbf{x}), \text{ with } \mathbf{w} \in \mathbb{R}^d$$



PAC-Bayesian analysis

- ▶ Space of all linear classifiers $\mathcal{F}_d := \{f_{\mathbf{v}} | \mathbf{v} \in \mathbb{R}^d\}$
- ▶ Gaussian prior $P_{\mathbf{u}} := \mathcal{N}(\mathbf{u}, I_d)$ over \mathcal{F}_d
- ▶ Gaussian posterior $Q_{\mathbf{w}} := \mathcal{N}(\mathbf{w}, I_d)$ over \mathcal{F}_d
- ▶ Predictor $F_{\mathbf{w}}(\mathbf{x}) := \mathbb{E}_{\mathbf{v} \sim Q_{\mathbf{w}}} f_{\mathbf{v}}(\mathbf{x}) = \text{erf} \left(\frac{\mathbf{w} \cdot \mathbf{x}}{\sqrt{d} \|\mathbf{x}\|} \right)$
- ▶ Linear loss $\ell(f_{\mathbf{v}}(\mathbf{x}), y) := \frac{1}{2} - \frac{1}{2} y f_{\mathbf{v}}(\mathbf{x})$



Bound minimization

$$C n \widehat{\mathcal{L}}_S(F_{\mathbf{w}}) + \text{KL}(Q_{\mathbf{w}} \| P_{\mathbf{u}}) = C \frac{1}{2} \sum_{i=1}^n \text{erf} \left(-y_i \frac{\mathbf{w} \cdot \mathbf{x}_i}{\sqrt{d} \|\mathbf{x}_i\|} \right) + \frac{1}{2} \|\mathbf{w} - \mathbf{u}\|^2$$

Shallow Learning

Posterior $Q_\theta = \mathcal{N}(\theta, I_D)$, over the family of all networks $\mathcal{F}_D = \{f_\theta | \theta \in \mathbb{R}^D\}$, where

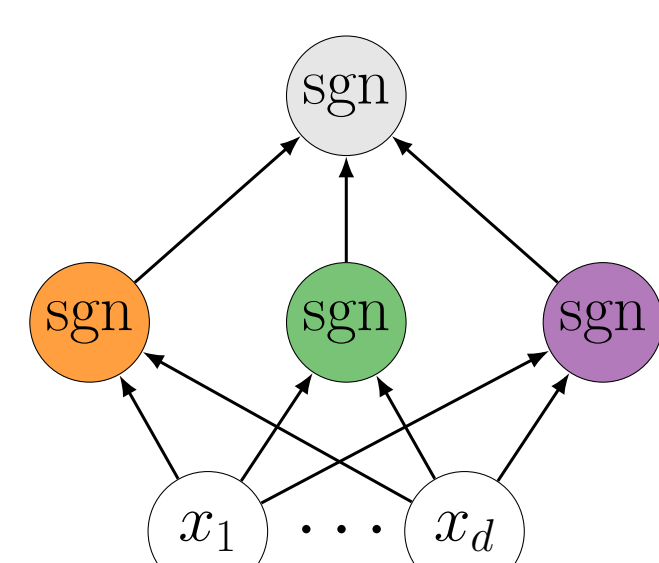
$$f_\theta(\mathbf{x}) = \text{sgn}(\mathbf{w}_2 \cdot \text{sgn}(\mathbf{W}_1 \mathbf{x}))$$

$$F_\theta(\mathbf{x}) = \mathbb{E}_{\theta \sim Q_\theta} f_\theta(\mathbf{x})$$

$$= \int_{\mathbb{R}^{d_1 \times d_0}} Q_1(\mathbf{V}_1) \int_{\mathbb{R}^{d_1}} Q_2(\mathbf{v}_2) \text{sgn}(\mathbf{v}_2 \cdot \text{sgn}(\mathbf{V}_1 \mathbf{x})) d\mathbf{v}_2 d\mathbf{V}_1$$

$$= \sum_{\mathbf{s} \in \{-1, 1\}^{d_1}} \text{erf} \left(\frac{\mathbf{w}_2 \cdot \mathbf{s}}{\sqrt{2d_1}} \right) \int_{\mathbb{R}^{d_1 \times d_0}} \mathbb{1}[\mathbf{s} = \text{sgn}(\mathbf{V}_1 \mathbf{x})] Q_1(\mathbf{V}_1) d\mathbf{V}_1$$

$$= \sum_{\mathbf{s} \in \{-1, 1\}^{d_1}} \underbrace{\text{erf} \left(\frac{\mathbf{w}_2 \cdot \mathbf{s}}{\sqrt{2d_1}} \right)}_{F_{\mathbf{w}_2}(\mathbf{s})} \prod_{i=1}^{d_1} \left[\frac{1}{2} + \frac{s_i}{2} \text{erf} \left(\frac{\mathbf{w}_1^i \cdot \mathbf{x}}{\sqrt{2} \|\mathbf{x}\|} \right) \right] \Pr(\mathbf{s} | \mathbf{x}, \mathbf{W}_1)$$



PAC-Bayesian bound ingredients

- ▶ Empirical loss : $\widehat{\mathcal{L}}_S(F_\theta) = \mathbb{E}_{\theta \sim Q_\theta} \widehat{\mathcal{L}}_S(f_\theta) = \frac{1}{n} \sum_{i=1}^n \left[\frac{1}{2} - \frac{1}{2} y_i F_\theta(\mathbf{x}_i) \right]$
- ▶ Complexity term : $\text{KL}(Q_\theta \| P_\mu) = \frac{1}{2} \|\theta - \mu\|^2$, with $\mu \in \mathbb{R}^D$
- ▶ Generalization bound : $\frac{1}{1 - e^{-C}} \left(1 - \exp \left(-C \widehat{\mathcal{L}}_S(F_\theta) - \frac{1}{n} [\text{KL}(Q_\theta \| P_\mu) + \ln \frac{2\sqrt{n}}{\delta}] \right) \right)$

Visualization

The proposed method can be interpreted as a majority vote of hidden layer representations.

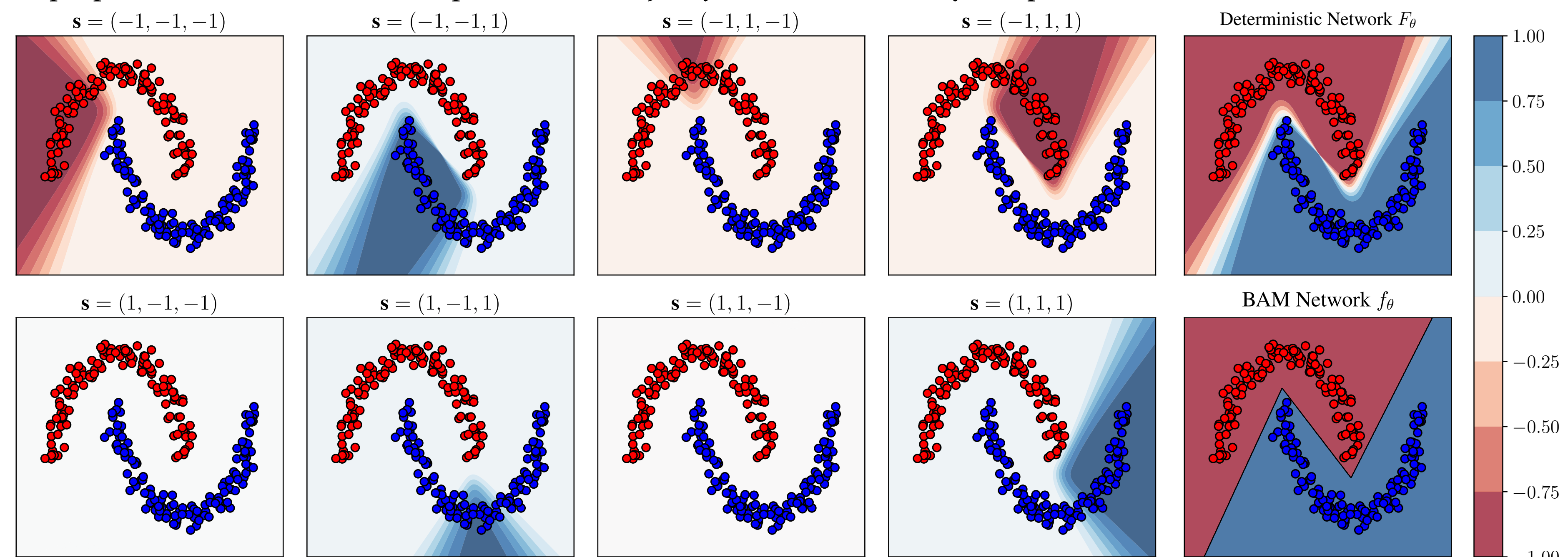


FIGURE 1: Illustration of the proposed method for a one hidden layer network of size $d_1=3$, interpreted as a majority vote over 8 binary representations $\mathbf{s} \in \{-1, 1\}^3$. For each \mathbf{s} , a plot shows the values of $F_{\mathbf{w}_2}(\mathbf{s}) \Pr(\mathbf{s} | \mathbf{x}, \mathbf{W}_1)$.

Stochastic Approximation

Prediction.

$$F_\theta(\mathbf{x}) = \sum_{\mathbf{s} \in \{-1, 1\}^{d_1}} F_{\mathbf{w}_2}(\mathbf{s}) \Pr(\mathbf{s} | \mathbf{x}, \mathbf{W}_1)$$

Hidden layer partial derivatives, with $\text{erf}'(x) := \frac{2}{\sqrt{\pi}} e^{-x^2}$.

$$\frac{\partial}{\partial \mathbf{w}_1^k} F_\theta(\mathbf{x}) = \frac{\mathbf{x}}{2^{\frac{3}{2}} \|\mathbf{x}\|} \text{erf}' \left(\frac{\mathbf{w}_1^k \cdot \mathbf{x}}{\sqrt{2} \|\mathbf{x}\|} \right) \sum_{\mathbf{s} \in \{-1, 1\}^{d_1}} s_k F_{\mathbf{w}_2}(\mathbf{s}) \left[\frac{\Pr(\mathbf{s} | \mathbf{x}, \mathbf{W}_1)}{\Pr(s_k | \mathbf{x}, \mathbf{w}_1^k)} \right]$$

Monte Carlo sampling

We generate T random binary vectors $\{\mathbf{s}^t\}_{t=1}^T$ according to $\Pr(\mathbf{s} | \mathbf{x}, \mathbf{W}_1)$.

$$F_\theta(\mathbf{x}) \approx \frac{1}{T} \sum_{t=1}^T F_{\mathbf{w}_2}(\mathbf{s}^t) \quad \frac{\partial}{\partial \mathbf{w}_1^k} F_\theta(\mathbf{x}) \approx \frac{\mathbf{x}}{2^{\frac{3}{2}} \|\mathbf{x}\|} \text{erf}' \left(\frac{\mathbf{w}_1^k \cdot \mathbf{x}}{\sqrt{2} \|\mathbf{x}\|} \right) \frac{1}{T} \sum_{t=1}^T \frac{s_k^t}{\Pr(s_k^t | \mathbf{x}, \mathbf{w}_1^k)} F_{\mathbf{w}_2}(\mathbf{s}^t)$$

This turns out to be a variant of the REINFORCE algorithm (Williams, 1992).

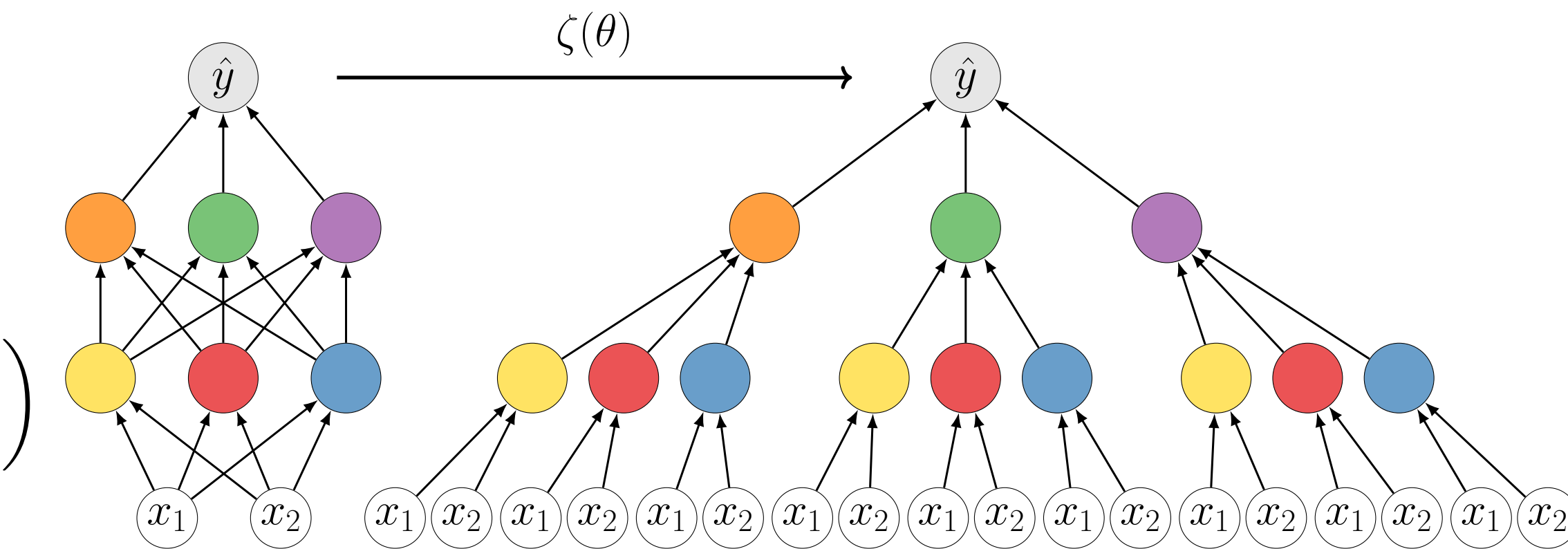
Deep Learning (PBGNet)

To enable a layer-by-layer computation of the prediction function, we want the neurons of a given layer to be independent of each other. This is achieved with the tree architecture mapping function $\zeta(\theta)$ applied on a multilayer network.

Recursive definition. $F_k^{(j)}$ denotes the output of the j^{th} neuron of the k^{th} hidden layer :

$$F_1^{(j)}(\mathbf{x}) = \text{erf} \left(\frac{\mathbf{w}_1^j \cdot \mathbf{x}}{\sqrt{2} \|\mathbf{x}\|} \right) \quad \leftarrow \text{first hidden layer}$$

$$F_{k+1}^{(j)}(\mathbf{x}) = \sum_{\mathbf{s} \in \{-1, 1\}^{d_k}} \text{erf} \left(\frac{\mathbf{w}_{k+1}^j \cdot \mathbf{s}}{\sqrt{2d_k}} \right) \prod_{i=1}^{d_k} \left(\frac{1}{2} + \frac{1}{2} s_i \times F_k^{(i)}(\mathbf{x}) \right)$$



Kullback-Leibler regularization

$$\text{KL}(Q_{\zeta(\theta)} \| P_{\zeta(\mu)}) = \frac{1}{2} \left(\|\mathbf{w}_L - \mathbf{u}_L\|^2 + \sum_{i=1}^{L-1} d_{k+1}^i \|\mathbf{W}_i - \mathbf{U}_i\|^2 \right) \text{ for } d_k^i := \prod_{i=k}^L d_i.$$

Experiment

We perform model selection using a validation set for a MLP with tanh activations, and using the PAC-Bayes bound for our PBGnet and PBGnet_{pre} algorithms. PBGnet_t and PBGnet_{t-bnd} are intermediate variants.

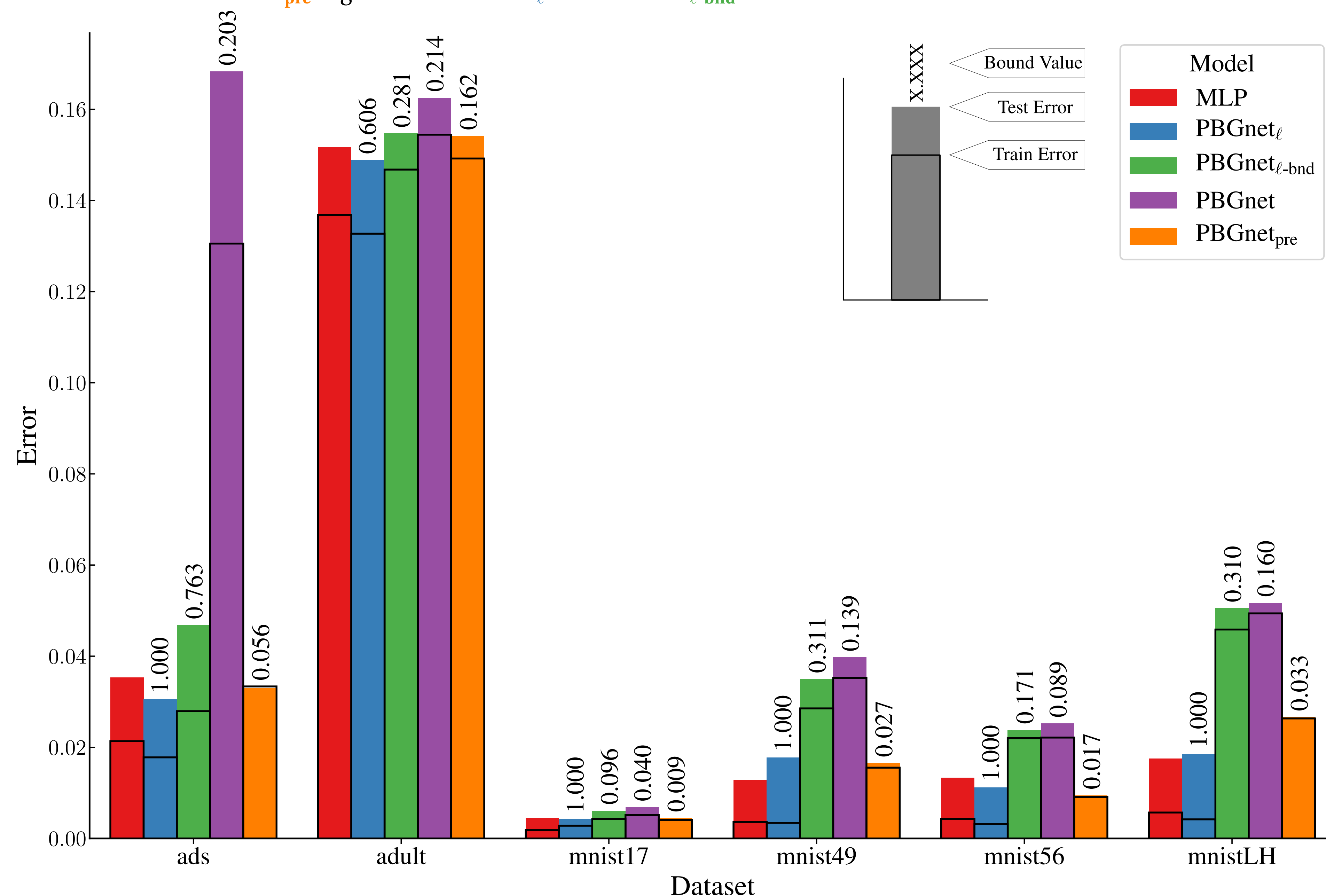


FIGURE 2: Experiment results for the considered models on the binary classification datasets. PAC-Bayesian bounds hold with probability 0.95.