Regularization by Misclassification in ReLU Neural Networks

Elisabetta Cornacchia Jan Hazla
Ido Nachum Amir Yehudayoff

June 7, 2021

Outline

Main Question

How does a neural network evolves when trained over a complex dataset?

We add label noise during training to model such datasets.

Highlights

- Implicit L2 regularization
- Modes in which a network can die during training (for example, dying ReLUs)
- Conjecture a new mode of self regularization
- Practical suggestion for training neural networks

SGD with Hinge Loss for a Single Neuron

Dataset: $x_i \in \mathbb{R}^d$, $y_i \in \{\pm 1\}$

Hinge Loss: $H(x) = \max\{0, x\} = \text{ReLU}(x)$

The perceptron algorithm $\triangleq$ SGD with mini-batch of size 1

over the loss: $\sum_{i=1}^{n} H(y_i V^{(t)} \cdot x_i)$

SGD update with a learning rate h:
$V^{(t+1)} = V^{(t)} + h y_i x_i$

Why Hinge Loss?

- Convergence guarantee for a single neuron (Novikoff, 1963)
- A simple update rule for ReLU networks
- Cross entropy is a smoothed version

SGD with Hinge Loss for a Two Layer Neural Network

Neural Network: $N(x) = V \cdot \text{ReLU}(M \cdot x)$
$V = (v_1, ..., v_k)$ for k neurons in the hidden layer

SGD update with a learning rate h:

if $y \cdot N(x) \leq 0$ (if the network missclasifies x)

$$V^{(t+1)} = V^{(t)} + hy \cdot \text{ReLU}(M^{(t)} \cdot x)$$

if $M_i^{(t)} \cdot x \geq 0$ (if neuron i in the hidden layer fires)

$$M_i^{(t+1)} = M_i^{(t)} + hyv_i^{(t)} \cdot x$$

Implicit L2 Regularization

Theorem

Let $N : \mathbb{R}^d \to \mathbb{R}$ *be a* ReLU *neural network of any architecture and a sample* (x, y) *such that* $y \cdot N(x) < 0$.
There exists a learning rate $h_0 > 0$ *such that for all* $0 < h < h_0$ *and any layer* M *of* N *it holds that*

$$\left\| M^{(t+1)} \right\|_F < \left\| M^{(t)} \right\|_F$$

A Proof for a Single Neuron

$$\left\| V^{(t+1)} \right\|^2 = (V^{(t)} + hyx) \cdot (V^{(t)} + hyx) =$$
$$\left\| V^{(t)} \right\|^2 + 2hyV^{(t)} \cdot x + h^2 \left\| x \right\|^2$$

A Proof for a Two Layer Network

$$\left\| M_i^{(t+1)} \right\|^2 = (M_i^{(t)} + hyv_i^{(t)} x) \cdot (M_i^{(t)} + hyv_i^{(t)} x) =$$
$$\left\| M_i^{(t)} \right\|^2 + 2hyv_i^{(t)} M_i^{(t)} \cdot x + h^2 (v_i^{(t)})^2 \left\| x \right\|^2$$

$$\left\| M_i^{(t+1)} \right\|^2 = \left\| M_i^{(t)} \right\|^2 + 2hyv_i^{(t)} \operatorname{ReLU}(M_i^{(t)} \cdot x) + h^2 (v_i^{(t)})^2 \left\| x \right\|^2$$

$$\left\| M^{(t+1)} \right\|^2 = \left\| M^{(t)} \right\|^2 + 2hyV^{(t)} \cdot \operatorname{ReLU}(M^{(t)} \cdot x) + h^2 \left\| V^{(t)} \right\|^2 \left\| x \right\|^2$$

Corollary

In gradient flow training, the weight matrix norm of every layer monotonically decreases as a function of t.

Questions

Is it possible for the weights to decay to zero, effectively cutting off all connections in the network?

Or relatedly, what are the implications of adding noise to the labels during training?

Learning Pure Noise

Definition

A neural network learns pure noise for t steps if at each step (x, y) is chosen independently according to $x \sim \mathcal{D}$ and $y \sim U\{\pm 1\}$.

A Single Neuron

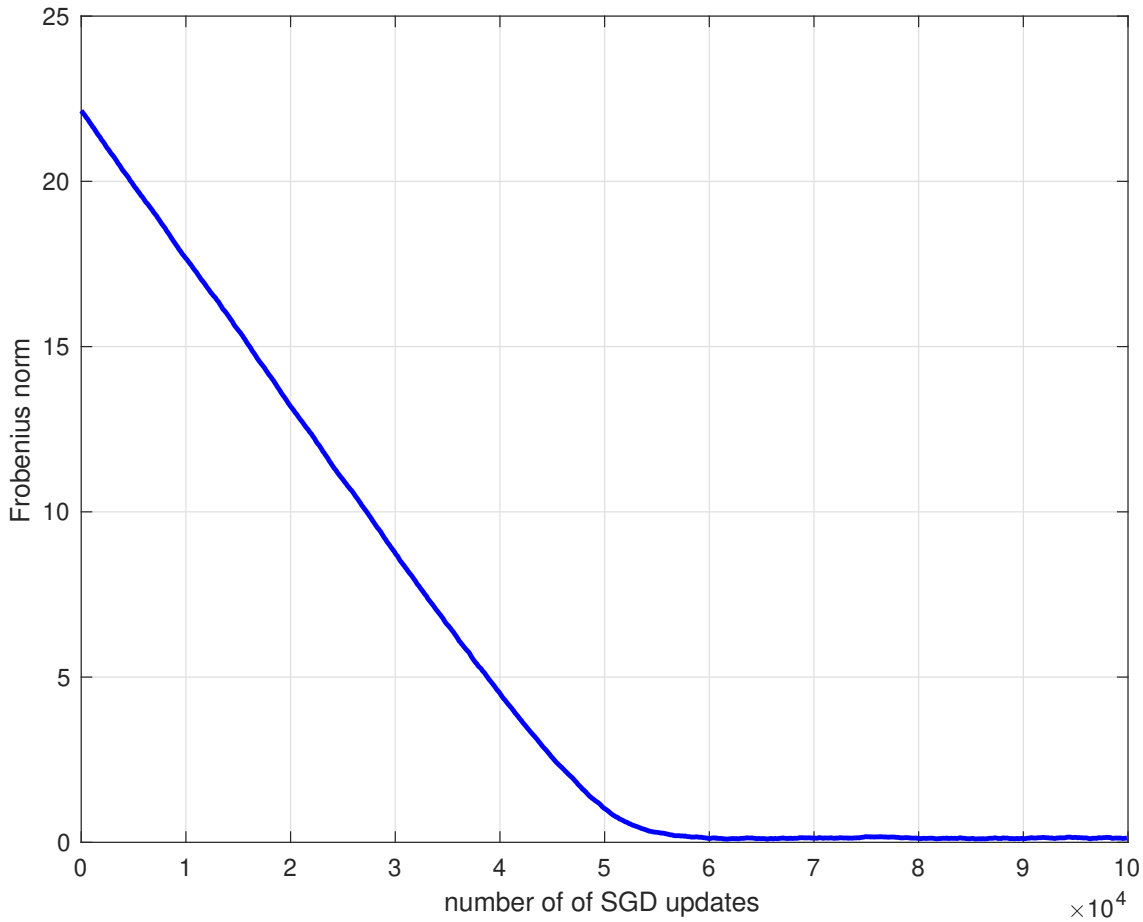


Figure 1: Norm evolution — learning noise with $x \sim N(0, I_d)$.Input dimension is $d = 30$ and the learning rate is $h = 1/d^2$.

Theorem

Assume the loss function satisfies $\sup_x L'(x) = M$ *and (a)* $\lim_{x\to 0^+} L'(x) - \lim_{x\to 0^-} L'(x) = m > 0$ *(as in hinge loss) or (b)* $L''(0) = m > 0$ *(as in cross entropy) and let* $V^{(0)} \in \mathbb{R}^d$ *be an initialization for a single neuron (including the bias term). With high probability , as h goes to zero with other parameters fixed, after training under pure label noise with* $\mathcal{D} = \mathcal{N}(0, I_d)$*, and after at most*

(a) $T = O(\frac{\|V^{(0)}\|}{mh})$ *steps,* $\|V^{(t)}\|$ *decreases to* $O\Big(dh \max\{\frac{M^2}{m}, 1)\}\Big)$.

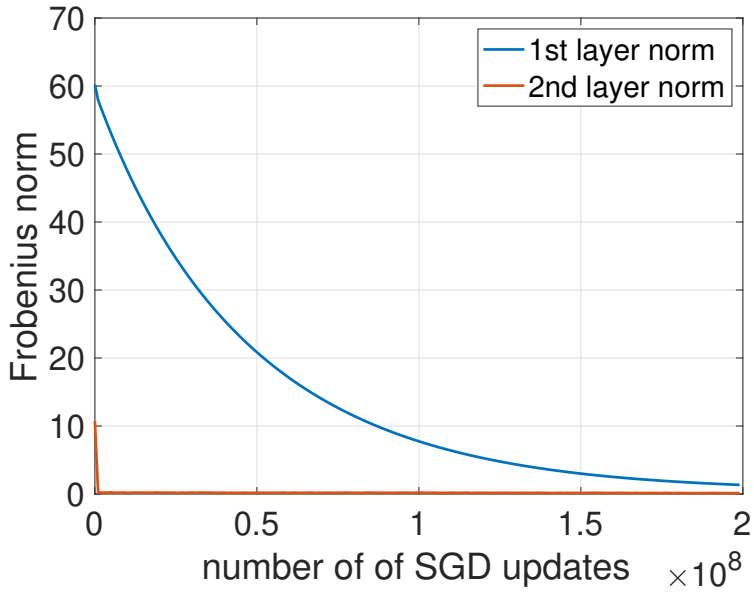
(b) $T = O(\frac{\|V^{(0)}\|}{mdh^{3/2}})$ *steps,* $\|V^{(t)}\|$ *decreases to* $O\Big(d\sqrt{h} \max\{\frac{M^2}{m}, 1\}\Big)$.

A Two Layer Network

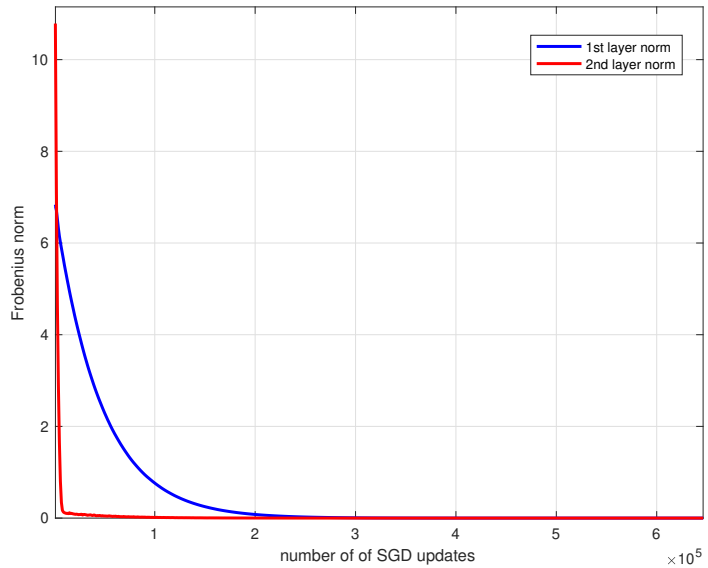
Will the network decay to zero?

No Bias Neurons

Offers insight, for example, for the hardness of learning parities using neural networks (Abbe and Sandon, 2019).



(a) cross entropy loss



(b) hinge loss

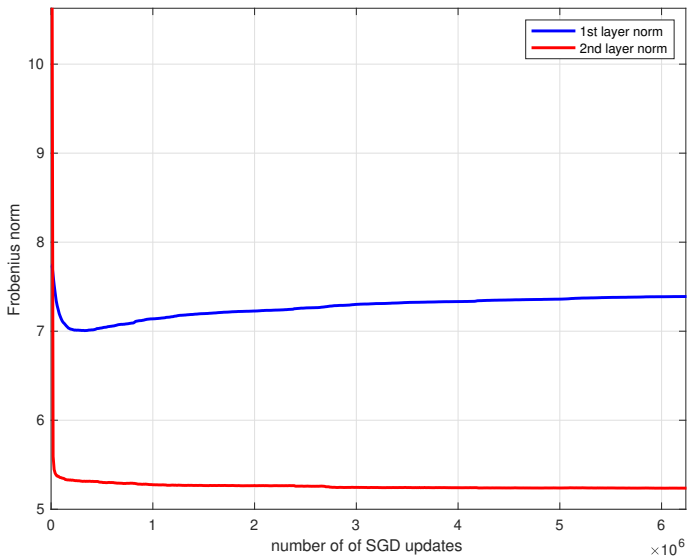
Figure 2: Norm evolution - learning noise without bias neurons with $x \sim N(0, I_d)$. Input dimension is $d = 30$ and learning rate is $h = 1/d$.

The Typical Number of Active Neurons

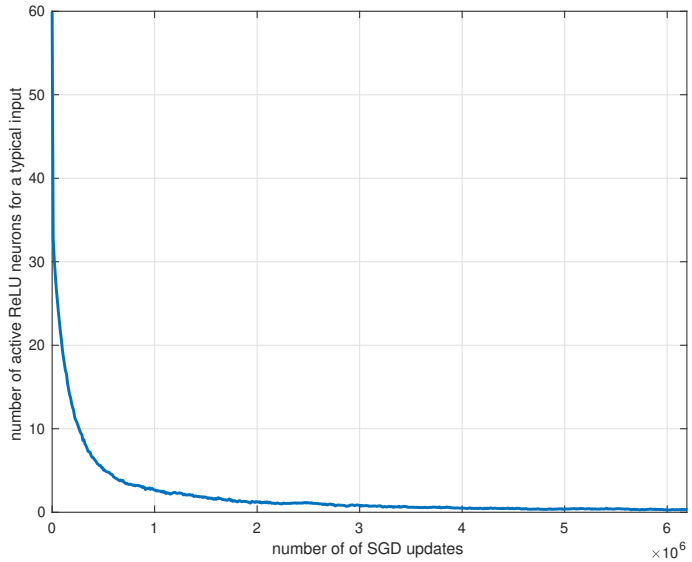
Definition

Let $N(x) = W_2 \cdot \mathrm{ReLU}(W_1 \cdot x + B_1) + B_2$ be a fully connected network with one hidden layer. For x in the dataset, the number of active neurons is $A_N(x) = |\{i \mid w_i \cdot x + b_i > 0\}|$ where (w_i, b_i) correspond to the weights of neuron i in the hidden layer. The typical number of active neurons is $\mathbb{E}_x A_N(x)$, where x is uniformly distributed in the dataset.

With Bias Neurons



(a) Norm evolution

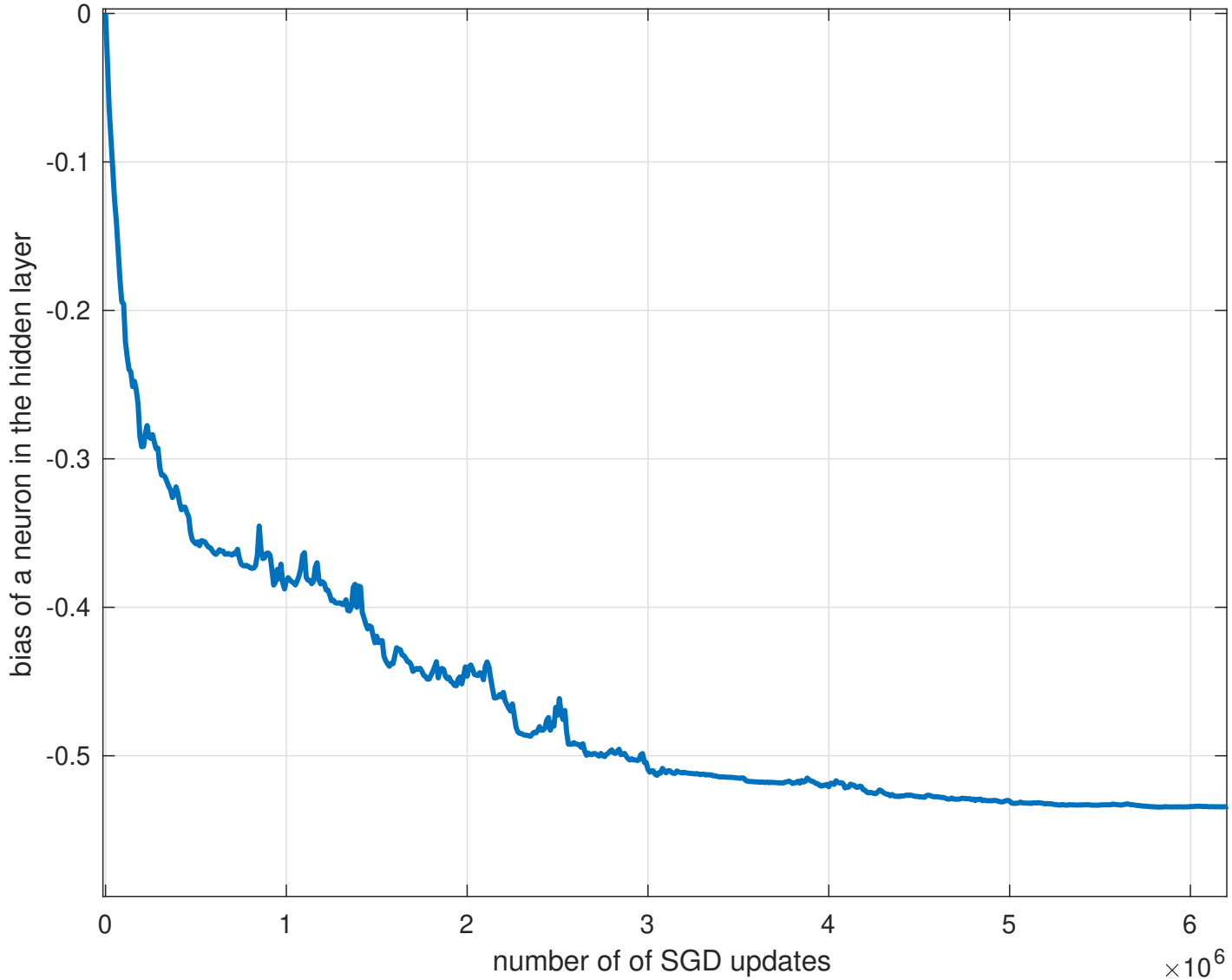


(b) Number of active neurons

Figure 3: Learning noise with bias neurons with $x \sim N(0, I_d)$. The learning rate is $h = 1/d$.

With Bias Neurons

Offers insight for the phenomenon of dying ReLUs.



Noisy Labels

Does adding some label noise hurts training and performance?

Learning High Dimensional Cubes

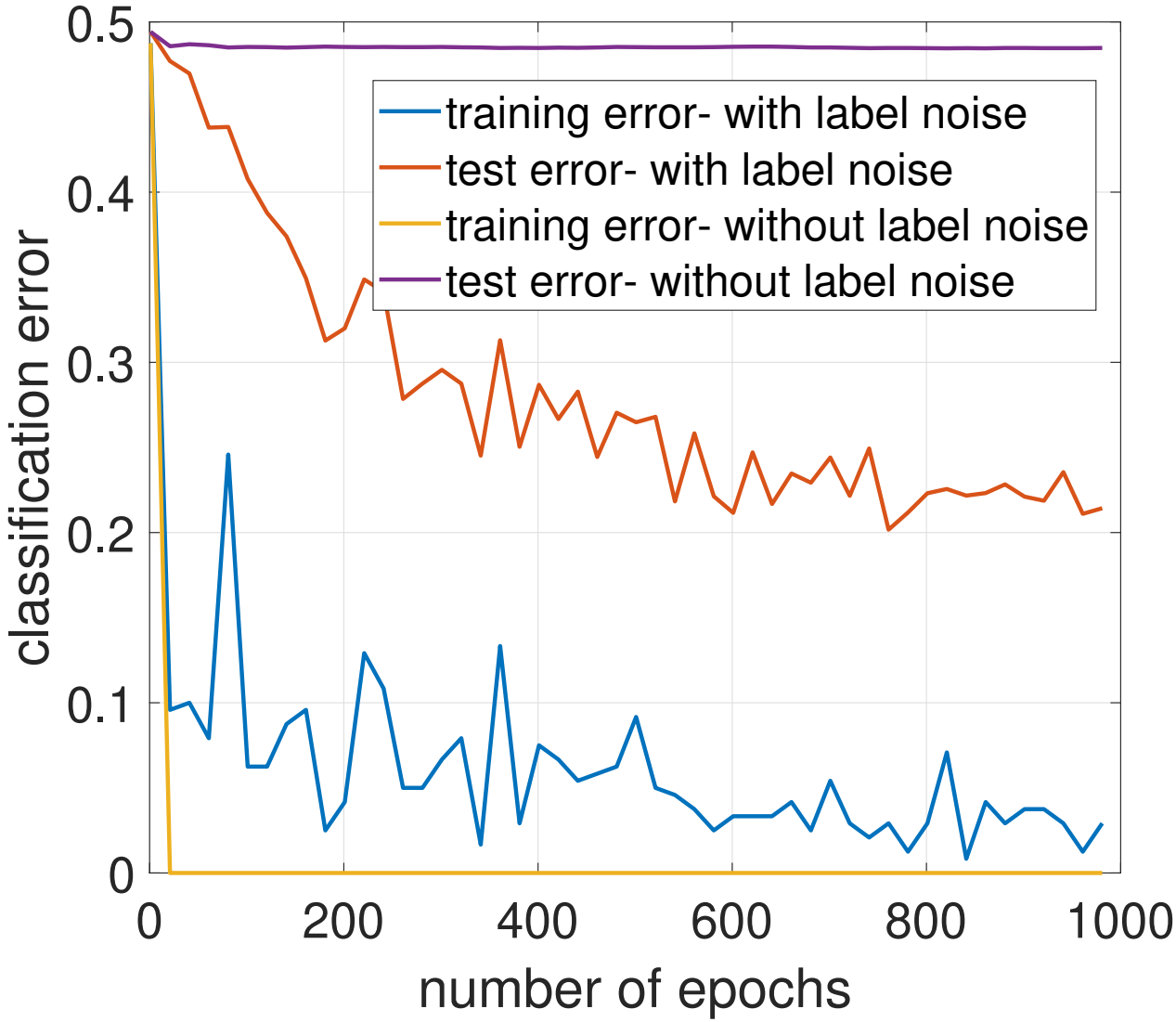


Figure 5: Learning squares. The input dimension is $d = 60$ and the learning rate is $h = 1/d$.

Learning High Dimensional Cubes

Offers another perspective for Blanc et al. 2020.

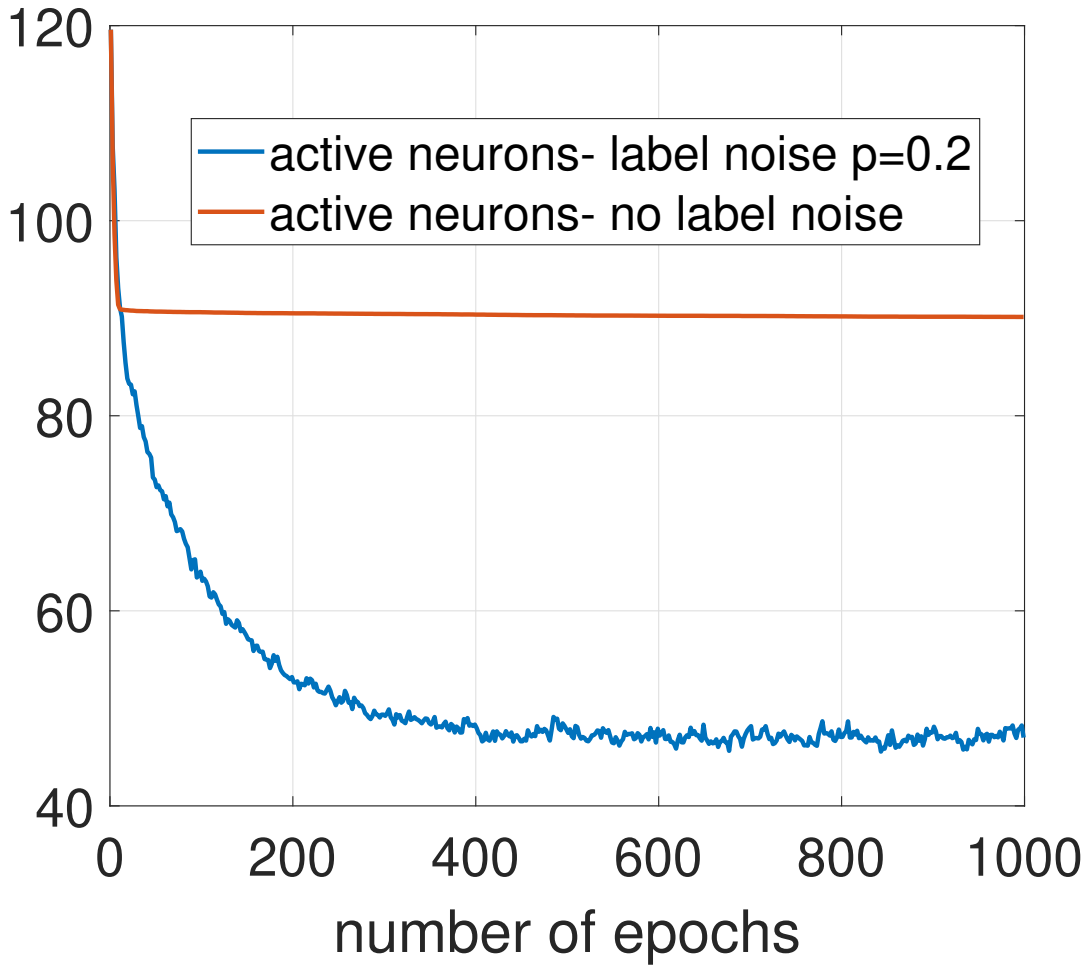


Figure 6: Learning squares. The input dimension is $d = 60$ and the learning rate is $h = 1/d$.

Learning MNIST

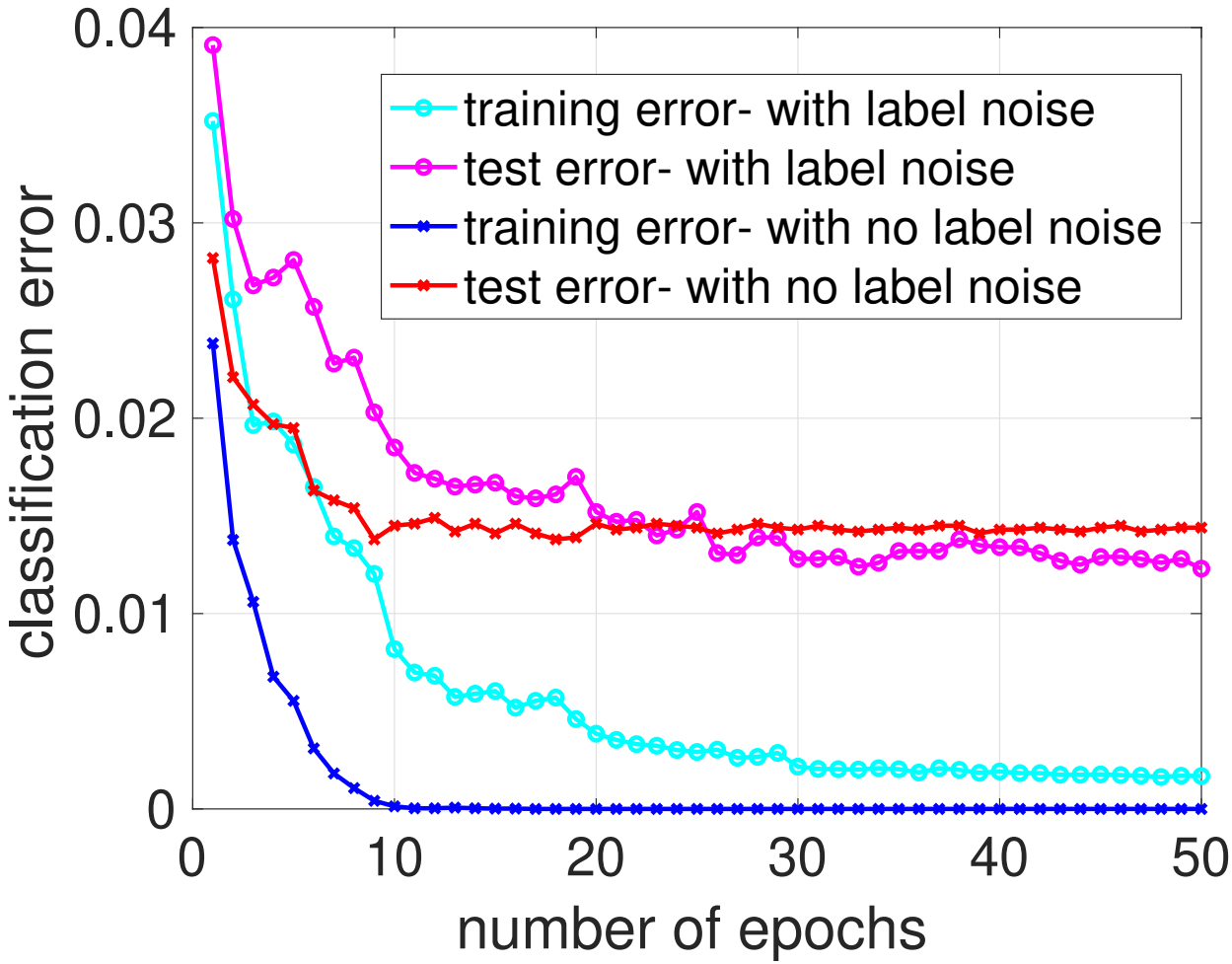


Figure 7: Test classification error and empirical classification error.

Learning MNIST

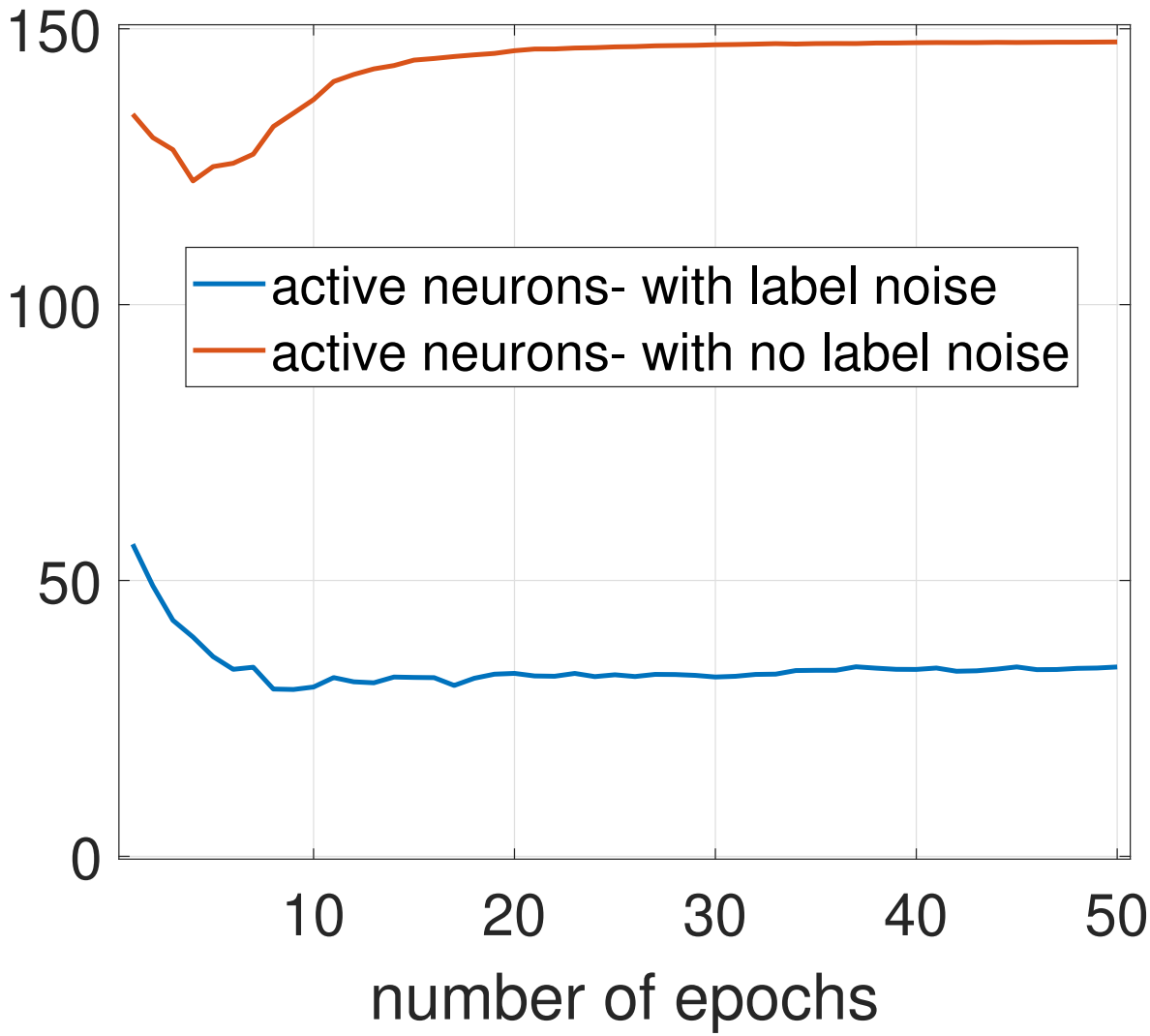


Figure 8: Number of active neurons

Step Size and Sparsification

Theorem

Let N be a network with one hidden layer with $2k$ ReLU neurons: $N(x) = V \cdot ReLU(Wx)$ and no bias neurons. Assume only W is trained and that its weights are initialized iid $U[-1, 1]$ and that we fix V to be split equally between 1 and -1: $V = (1, \ldots, 1, -1, \ldots, -1)$.
As k grows, w.h.p., if N is trained with the hinge loss under pure label noise with $\mathcal{D} = U\{e_1, \ldots, e_d\}$ (uniform over the standard basis), we will have after training:

1. *For a learning rate $h \geq 1$, for all i it holds $\mathrm{ReLU}(We_i) = \overrightarrow{0}$.*
2. *For a learning rate $h \leq 1/k$, for all i it holds $\mathrm{ReLU}(We_i) \neq \overrightarrow{0}$. Furthermore, at most $k + o(k)$ (i.e., $1/2 + o(1)$ fraction) of the coordinates of the vector $\mathrm{ReLU}(We_i)$ are zero.*

Summary

- SGD implicitly regularizes the Frobenius norm of neural networks in classification tasks.
- Analysis of learning pure noise offers insight for the hardness of learning parities and the phenomenon of dying ReLUs.
- Perhaps counter-intuitively, adding label noise may improve the generalization error.
- Label noise implicitly regularize another aspect of the network. It sparsifies the firing pattern of the hidden layer for a typical input.

- Stronger theoretical foundations are still missing...

Questions?