Convex Geometry of Two-Layer ReLU Networks: Implicit Autoencoding and Interpretable Models

Tolga Ergen
Stanford University

Mert Pilanci
Stanford University

Abstract

We develop a convex analytic framework for ReLU neural networks which elucidates the inner workings of hidden neurons and their function space characteristics. We show that rectified linear units in neural networks act as convex regularizers, where simple solutions are encouraged via extreme points of a certain convex set. For one dimensional regression and classification, we prove that finite two-layer ReLU networks with norm regularization yield linear spline interpolation. In the more general higher dimensional case, we show that the training problem for two-layer networks can be cast as a convex optimization problem with infinitely many constraints. We then provide a family of convex relaxations to approximate the solution, and a cutting-plane algorithm to improve the relaxations. We derive conditions for the exactness of the relaxations and provide simple closed form formulas for the optimal neural network weights in certain cases. Our results show that the hidden neurons of a ReLU network can be interpreted as convex autoencoders of the input layer. We also establish a connection to ℓ_0-ℓ_1 equivalence for neural networks analogous to the minimal cardinality solutions in compressed sensing. Extensive experimental results show that the proposed approach yields interpretable and accurate models.

1 Introduction

Understanding the fundamental reason why training over-parameterized Deep Neural Networks (DNNs) converges to minimizers that generalize well remains an open problem. Recently, it was empirically observed that ReLU NNs exhibit an interesting structure, where only finitely many simple functions can be obtained as optimal solutions (Maennel et al., 2018; Savarese et al., 2019). In (Savarese et al., 2019), the function space of one dimensional (1D) ReLU regression networks was studied, where it was shown that among infinitely many two-layer ReLU networks that perfectly fit the training data, the one with the minimum Euclidean norm parameters yields a linear spline interpolation. It is possible that the structure induced by over-parameterization explains remarkable generalization properties of DNNs. Despite the dramatic surge of interest in NNs, the fundamental mechanism behind these simple structures in over-parameterized networks is largely unknown.

In this paper, we develop a convex analytic framework to analyze two-layer ReLU NNs and characterize the structure that emerges as a result of over-parameterization. We show that over-parameterized networks behave like *convex regularizers*, where simple solutions are encouraged via *extreme points* of a certain convex set. Our results are analogous to ℓ_1-norm regularization where sparse solutions are encouraged as a result of the 1-sparse extreme points of the ℓ_1-ball. However, unlike these methods, we show that the extreme points in over-parameterized NNs are data-adaptive and precisely serve as *convex autoencoders*. We fully describe the extreme points via analytical expressions. In one dimensional regression and classification, the extreme points manifest as finitely many simple quantized solutions, and yield linear spline interpolations for regression/classification tasks explaining and extending recent results.

1.1 Related work

A line of research (Maennel et al., 2018; Blanc et al., 2019; Zhang et al., 2016) explored the behavior of ReLU networks in finite size cases. In (Zhang et al., 2016), the authors indicated that NNs are implicitly regularized during training since Stochastic Gradient Descent (SGD) converges to a solution with small

Proceedings of the 23$^{\mathrm{rd}}$International Conference on Artificial Intelligence and Statistics (AISTATS) 2020, Palermo, Italy. PMLR: Volume 108. Copyright 2020 by the author(s).

norm. The idea of implicit regularization was also extended to the networks trained with GD as well as SGD. Particularly, the authors in (Maennel et al., 2018) showed that implicit regularization has a strong connection with the initialization of a network and proved that network weights tend to align along certain directions determined by the input data, which implies that there are only finitely many possible simple functions for a given dataset. In order to explain generalization capabilities of ReLU networks, another line of research in (Bengio et al., 2006; Wei et al., 2018; Bach, 2017; Chizat and Bach, 2018) focused on infinitely wide two-layer ReLU networks. In (Bengio et al., 2006), the authors introduced an algorithm that can train a regularized NN with infinite width in an incremental manner. In (Wei et al., 2018), the authors adopted a margin-based perspective, where they showed that the optimal point of a weakly regularized loss has the maximum margin property, thus, over-parameterization can improve generalization bounds.

1.2 Our contributions

1) We develop a convex analytic framework for two-layer ReLU NNs to provide a deeper insight into over-parameterization and implicit regularization. We show that over-parameterized NNs behave like convex regularizers, where simple structures are encouraged in the solution via the extreme points of a well-defined regularizer; **2)** For one dimensional regression and classification, we prove that hidden layers form a linear spline interpolation. We also provide an intuitive convex geometric explanation of this fact, and derive a general formula for the hidden layer representation in higher dimensions; **3)** We provide a convex relaxation based training procedure, which is proven to be exact under certain assumptions on the training set. We also prove that these assumptions hold in generic regimes, e.g., when the training examples are i.i.d. random; **4)** We establish an ℓ_0-ℓ_1 equivalence for NNs, which parallels minimum cardinality relaxations in compressed sensing. We then provide closed form expressions for the optimal ReLU networks in certain cases.

1.3 Overview of our results

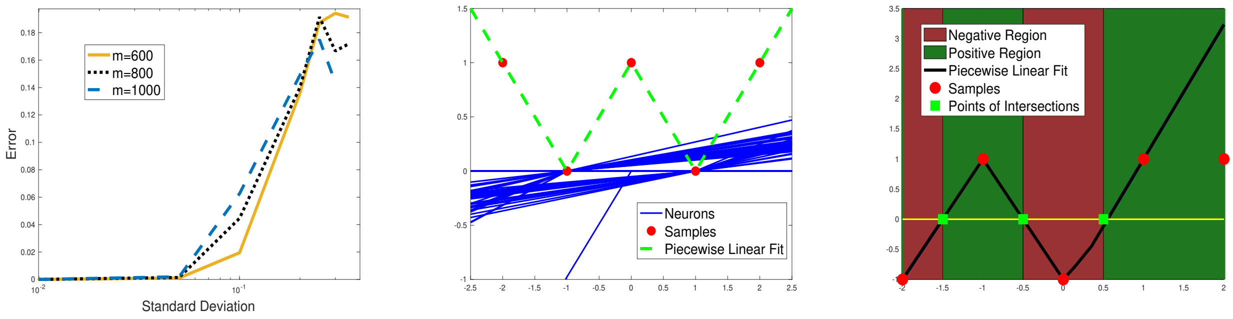
Implicit regularization plays a key role in training NNs, however, it is still theoretically elusive how NNs trained with gradient descent (GD) and no regularization obtain simple solutions, e.g., spline interpolation. In order to gain a deeper insight into the effects of initialization magnitude, we perform a simple experiment on training 1D ReLU NNs on the data shown in Figure 1b. The results in Figures 1a and 1b show that the two-layer ReLU regression network fits precisely a linear spline interpolation when the standard deviation of the (zero-mean) initialization is below a critical value. Thus, as emphasized by (Maennel et al., 2018; Chizat and Bach, 2018), initialization magnitude is critical for the final norm of the network parameters, so that GD converges solutions with smaller norm, i.e., closer to initialization, which can generalize as a result of this implicit regularization. In Figure 1b, we also display the set of neurons found by GD and the corresponding overall function fit in the case of small initialization. In this over-parameterized scenario, linear combination of the neurons with different weights and biases still outputs a linear spline interpolation. The same results also hold for the binary classification using a two-layer ReLU network with the hinge loss as illustrated in Figure 1c. The network fits a certain piecewise linear function and the decision region (to label the samples as ±1) boundaries become precisely the zero crossings of this function. The central questions we will address in this paper are: *Why are over-parameterized NNs providing a linear spline interpolation in 1D? Is there a general mechanism encouraging simple solutions in arbitrary dimensions? How are the decision regions formed ?* We show that these questions are completely and rigorously answered using *convex geometry* and *duality.*

Simply stated, we show that the optimal solutions have kinks at input data points because the convex approximation[1] of a data point x_i given by $\min_{\boldsymbol{\lambda} \succcurlyeq 0, \sum_j \lambda_j = 1} |x_i - \sum_{j \in \mathcal{S}, j \neq i} \lambda_j x_j|$ is given by another data point, i.e., an extreme point of the convex hull of data points in $\mathcal{S} \backslash \{i\}$. Consequently, input data points are the only allowed hidden neuron activation thresholds at optimum for 1D networks. We further provide a general formula for the hidden neuron configurations in higher dimensions.

Specifically, we focus on minimizers for two-layer NNs with small Euclidean norm. In one dimensional data sets, through convex analytic arguments, we establish that each training sample becomes an extreme point of a certain convex set, which means that the activation threshold of a ReLU function has to correspond to one of the data samples. This result completely explains what we observe in Figure 1. Since the data samples are the activation thresholds, we observe a piecewise linear function as the neural network output, where the kinks occur exactly at the data samples (activation points). Our analysis also reveals that the hidden neurons can be interpreted as data autoencoders in higher dimensions and they can further be expressed in closed form in certain cases.

Notation: We denote the matrices and vectors as uppercase and lowercase bold letters, respectively. To denote a vector or matrix of zeros or ones, we use **0**

[1] Here x_i is an arbitrary sample, and $\mathcal{S}$ is an arbitrary subset of data points. λ_j's are mixture weights, approximating x_i as a convex mixture of the data points in $\mathcal{S} \backslash \{i\}$.



(a) Deviation of the ReLU network output from linear spline vs standard deviation of initialization plotted for different number of hidden neurons m.

(b) Contribution of each neuron along with the overall fit. Each activation point corresponds to a particular data sample.

(c) Binary classification using hinge loss. Network output is piecewise linear, and decision regions are determined by zero crossings (see Lemma 2.6).

Figure 1: Analysis of one dimensional regression and classification with a two-layer NN.

or $\mathbf{1}$, respectively, where the sizes are understood from the context. Additionally, $\mathbf{I}_k$ represents the identity matrix of the size k. We also use $(x)_+ = \max\{x, 0\}$ for the ReLU activation. Furthermore, we denote the set of integers from 1 to n as $[n]$ and use the notation $\mathbf{e}_1, ...\mathbf{e}_n$ for the ordinary basis vectors in $\mathbb{R}^n$.

2 Preliminaries

Given n data samples, i.e., $\{\mathbf{a}_i\}_{i=1}^n, \mathbf{a}_i \in \mathbb{R}^d$, we consider two-layer NNs with m hidden neurons and ReLU activations. Initially, we focus on the scalar output case for simplicity, i.e.,

$$f(\mathbf{A}) = \sum_{j=1}^{m} w_j (\mathbf{A}\mathbf{u}_j + b_j \mathbf{1})_+, \tag{1}$$

where $\mathbf{A} \in \mathbb{R}^{n \times d}$ is the data matrix, $\mathbf{u}_j \in \mathbb{R}^d$ and $b_j \in \mathbb{R}$ are the parameters of the j^{th} hidden neuron, and w_j's are the weights for the output layer. For a more compact representation, we also define $\mathbf{U} \in \mathbb{R}^{d \times m}$, $\mathbf{b} \in \mathbb{R}^m$, and $\mathbf{w} \in \mathbb{R}^m$ as the hidden layer weight matrix, the bias vector, and the output layer weight vector, respectively. Thus, (1) can be written as $f(\mathbf{A}) = (\mathbf{A}\mathbf{U} + \mathbf{1}\mathbf{b}^T)_+ \mathbf{w}$.

Note that we can assume that the bias term for the output layer is zero without loss of generality, since we can recover the general case (Maennel et al., 2018). Given the data matrix $\mathbf{A}$ and the label vector $\mathbf{y} \in \mathbb{R}^n$, consider training the network by solving the following optimization problem

$$\min_{\theta \in \Theta} \Big\| \sum_{j=1}^{m} w_j (\mathbf{A}\mathbf{u}_j + b_j \mathbf{1})_+ - \mathbf{y} \Big\|_2^2 + \beta \sum_{j=1}^{m} (\|\mathbf{u}_j\|_2^2 + w_j^2)\,, \tag{2}$$

where β is a regularization parameter and we define the overall parameter space Θ as $\theta \in \Theta = \{(\mathbf{U}, \mathbf{b}, \mathbf{w}, m) \,|\, \mathbf{U} \in \mathbb{R}^{d \times m}, \mathbf{b} \in \mathbb{R}^m, \mathbf{w} \in \mathbb{R}^m, m \in \mathbb{Z}_+\}$. Based on our observations in Figure 1a and the results in (Savarese et al., 2019; Chizat and Bach, 2018; Neyshabur et al., 2014), we first focus on a minimum norm[2] variant of (2). We define the squared Euclidean norm of the weights (without biases) as $R(\theta) = \|\mathbf{w}\|_2^2 + \|\mathbf{U}\|_F^2$. Then we consider the following optimization problem

$$\min_{\theta \in \Theta} R(\theta) \text{ s.t. } f_\theta(\mathbf{A}) = \mathbf{y}, \tag{3}$$

where the over-parameterization allows us to reach zero training error over $\mathbf{A}$ via the ReLU network in (1). The next lemma shows that the minimum squared Euclidean norm is equivalent to minimum ℓ_1 norm after a rescaling. This result was also presented in (Savarese et al., 2019; Neyshabur et al., 2014).

Lemma 2.1 ((Savarese et al., 2019; Neyshabur et al., 2014)). *The following two optimization problems are equivalent:*

$$\begin{aligned} P^* &= \min_{\theta \in \Theta} R(\theta) \;\; s.t. \;\; f_\theta(\mathbf{A}) = \mathbf{y} \\ &= \min_{\theta \in \Theta} \|\mathbf{w}\|_1 \;\; s.t. \;\; f_\theta(\mathbf{A}) = \mathbf{y}, \|\mathbf{u}_j\|_2 = 1, \forall j. \end{aligned}$$

Lemma 2.2. *Replacing $\|\mathbf{u}_j\|_2 = 1$ with $\|\mathbf{u}_j\|_2 \leq 1$ does not change the value of the above problem.*

By Lemma 2.1 and 2.2, we can express (3) as

$$\min_{\theta \in \Theta} \|\mathbf{w}\|_1 \text{ s.t. } f_\theta(\mathbf{A}) = \mathbf{y}, \|\mathbf{u}_j\|_2 \leq 1, \forall j. \tag{4}$$

However, both (2) and (4) are quite challenging optimization problems due to the complicated behavior of an affine mapping along with the ReLU activation. In particular, depending on the properties of $\mathbf{A}$, e.g., singular values, rank, and dimensions, the geometry of the objective in (2) might considerably change.

[2] This can be regarded as *weak* regularization, i.e., $\beta \to 0$ in (2) (see e.g. (Wei et al., 2018) for a similar notion).

2.1 Geometry of the problem

In order to illustrate the geometry of (2), we particularly focus on a simple case where we have a single neuron with no bias and regularization, i.e., $m = 1$, $b_1 = 0$, and $\beta = 0$. Thus, (2) reduces to

$$\min_{\mathbf{u}_1} \left\|w_1(\mathbf{A}\mathbf{u}_1)_+ - \mathbf{y}\right\|_2^2 \text{ s.t. } \|\mathbf{u}_1\|_2 \leq 1. \tag{5}$$

The solution of (5) is completely determined by the set $\mathcal{Q}_\mathbf{A} = \{(\mathbf{A}\mathbf{u})_+ | \mathbf{u} \in \mathbb{R}^d, \|\mathbf{u}\|_2 \leq 1\}$. It is evident that (5) is solved via scaling this set by $|w_1|$ to minimize the distance to $+\mathbf{y}$ or $-\mathbf{y}$, depending on the sign of w_1. We note that since $\|\mathbf{u}\|_2 \leq 1$ describes a d-dimensional unit ball, $\mathbf{A}\mathbf{u}$ describes an ellipsoid whose shape and orientation is determined by the singular values and the output singular vectors of $\mathbf{A}$.

2.2 Rectified ellipsoid and its geometric properties

A central object in our analysis is the rectified ellipsoidal set introduced in the previous section, which is defined as $\mathcal{Q}_\mathbf{A} = \left\{\left(\mathbf{A}\mathbf{u}\right)_+ | \mathbf{u} \in \mathbb{R}^d, \|\mathbf{u}\|_2 \leq 1\right\}$. The set $\mathcal{Q}_\mathbf{A}$ is non-convex in general, as depicted in Figure 2b. However, there exist data matrices $\mathbf{A}$ for which the set $\mathcal{Q}_\mathbf{A}$ is convex, e.g., diagonal data matrices. 2a.

2.2.1 Spike-free matrices

We say that a matrix $\mathbf{A}$ is spike-free if it holds that $\mathcal{Q}_\mathbf{A} = \mathbf{A}\mathcal{B}_2 \cap \mathbb{R}^n_+$, where $\mathbf{A}\mathcal{B}_2 = \{\mathbf{A}\mathbf{u} \,|\, \mathbf{u} \in \mathcal{B}_2\}$, and $\mathcal{B}_2$ is the unit ℓ_2 ball defined as $\mathcal{B}_2 = \{\mathbf{u} \,|\, \|\mathbf{u}\|_2 \leq 1\}$. Note that $\mathcal{Q}_\mathbf{A}$ is a convex set if $\mathbf{A}$ is spike-free. In this case we have an efficient description of this set given by $\mathcal{Q}_\mathbf{A} = \{\mathbf{A}\mathbf{u} | \mathbf{u} \in \mathbb{R}^d, \|\mathbf{u}\|_2 \leq 1, \, \mathbf{A}\mathbf{u} \geq 0\}$.

If $\mathcal{Q}_\mathbf{A} = \{(\mathbf{A}\mathbf{u})_+ | \mathbf{u} \in \mathbb{R}^d, \|\mathbf{u}\|_2 \leq 1\}$ can be expressed as $\mathbb{R}^n_+ \cap \{\mathbf{A}\mathbf{u} | \mathbf{u} \in \mathbb{R}^d, \|\mathbf{u}\|_2 \leq 1\}$ (see Figure 2a), then (5) can be solved via convex optimization after the rescaling $\mathbf{u} = \mathbf{u}_1 w_1$

$$\begin{aligned} &\min_{\mathbf{u}} \left\|\mathbf{A}\mathbf{u} - \mathbf{y}\right\|_2^2 \\ &\text{s.t. } \mathbf{u} \in \{\mathbf{A}\mathbf{u} \not\succcurlyeq 0\} \cup \{-\mathbf{A}\mathbf{u} \not\succcurlyeq 0\}, \|\mathbf{u}\|_2 \leq 1\,. \end{aligned}$$

The following lemma provides a characterization of spike-free matrices

Lemma 2.3. *A matrix $\mathbf{A}$ is spike-free if and only if the following condition holds*

$$\forall \mathbf{u} \in \mathcal{B}_2, \exists \mathbf{z} \in \mathcal{B}_2 \text{ such that } \left(\mathbf{A}\mathbf{u}\right)_+ = \mathbf{A}\mathbf{z}. \tag{6}$$

Alternatively, $\mathbf{A}$ is spike free if and only if it holds that

$$\max_{\mathbf{u}\,:\,\|\mathbf{u}\|_2 \leq 1,\,(\mathbf{I}_n - \mathbf{A}\mathbf{A}^\dagger)(\mathbf{A}\mathbf{u})_+ = \mathbf{0}} \|\mathbf{A}^\dagger\left(\mathbf{A}\mathbf{u}\right)_+\|_2 \leq 1\,.$$

If $\mathbf{A}$ is full row rank, then the above condition simplifies to

$$\max_{\mathbf{u}\,:\,\|\mathbf{u}\|_2 \leq 1} \|\mathbf{A}^\dagger\left(\mathbf{A}\mathbf{u}\right)_+\|_2 \leq 1\,. \tag{7}$$

We note that the condition in (7) bears a close resemblance to the irrepresentability conditions in Lasso support recovery (see e.g. (Zhao and Yu, 2006)). It is easy to see that diagonal matrices are spike-free. More generally, any matrix of the form $\mathbf{A} = \mathbf{\Sigma}\mathbf{V}^T$, where $\mathbf{\Sigma}$ is diagonal, and $\mathbf{V}^T$ is any matrix with orthogonal rows, i.e., $\mathbf{V}^T\mathbf{V} = \mathbf{I}_n$, is spike-free. In other cases, $\mathcal{Q}_\mathbf{A}$ has a non-convex shape as illustrated in Figure 2b. Therefore, the ReLU activation might exhibit significantly complicated and non-convex behavior as the dimensionality of the problem increases. Note that $\mathbf{A}\mathcal{B}_2 \cap \mathbb{R}^n_+ \subseteq \mathcal{Q}_\mathbf{A}$ always holds, and therefore the former set is a convex relaxation of the set $\mathcal{Q}_\mathbf{A}$. We call this set spike-free relaxation of $\mathcal{Q}_\mathbf{A}$.

As another example for spike-free data matrices, consider the Singular Value Decomposition of the data matrix $\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$ in compact form. We can apply a whitening transformation on the data matrix by defining $\tilde{\mathbf{A}} = \mathbf{A}\mathbf{V}\mathbf{\Sigma}^{-1}$, which is known as zero-phase whitening in the literature. Note that the empirical covariance of the whitened data is diagonal since we have $\tilde{\mathbf{A}}^T\tilde{\mathbf{A}} = \mathbf{I}_n$. The following lemmas show that whitened matrices with $n \leq d$ and rank-one data matrices with positive left singular vectors are spike-free.

Lemma 2.4. *Whitened data matrices with $n \leq d$ are spike-free.*

Lemma 2.5. *Rank-one data matrices with positive left singular vectors are spike-free.*

2.3 Polar convex duality

It can be shown that the dual of the problem (4) is given by[3]

$$\max_{\mathbf{v}} \mathbf{v}^T\mathbf{y} \text{ s.t. } \mathbf{v} \in \mathcal{Q}^\circ_\mathbf{A}\ ,\ -\mathbf{v} \in \mathcal{Q}^\circ_\mathbf{A} \tag{8}$$

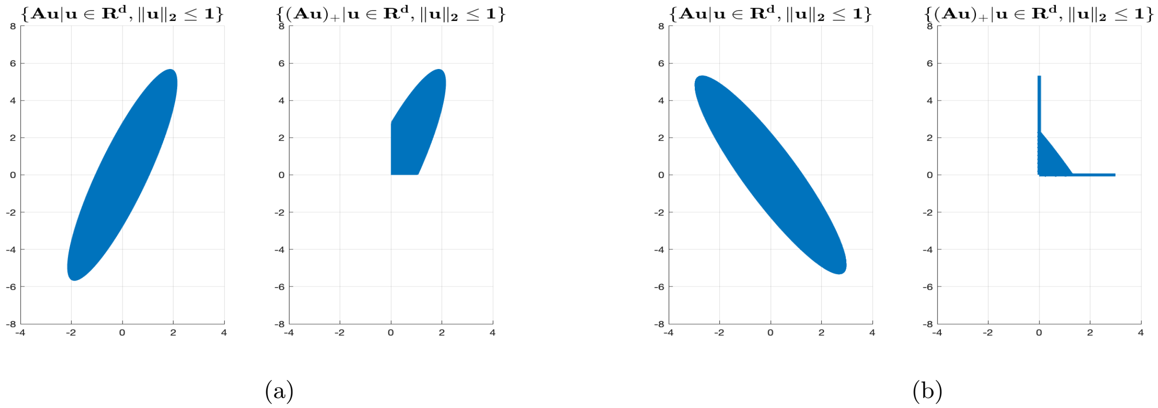
where $\mathcal{Q}^\circ_\mathbf{A}$ is the polar set (Rockafellar, 1970) of $\mathcal{Q}_\mathbf{A}$ defined as $\mathcal{Q}^\circ_\mathbf{A} = \{\mathbf{v} | \mathbf{v}^T\mathbf{u} \leq 1 \, \forall \mathbf{u} \in \mathcal{Q}_\mathbf{A}\}$.

2.4 Extreme Points

Let us first define the extreme point of $\mathcal{Q}_\mathbf{A}$ along $\mathbf{v}$ as $\operatorname{argmax}_{\mathbf{z} \in \mathcal{Q}_\mathbf{A}} \mathbf{v}^T\mathbf{z}$. Note that the endpoints of the spikes in Figure 2b are the extreme points in directions $\mathbf{e}_1$ and $\mathbf{e}_2$. In this section we show that the extreme points of $\mathcal{Q}_\mathbf{A}$ are given by data samples and convex mixtures of data samples in one dimensional and multidimensional cases, respectively. Here, we also provide a generic formulation for the extreme point along an arbitrary direction.

Lemma 2.6. *In a one dimensional data set ($d = 1$), for any vector $\mathbf{v} \in \mathbb{R}^n$, an extreme point of $\mathcal{Q}_\mathbf{A}$ along*

[3] We refer the reader to the supplementary material for the proof. For the remaining analysis, we drop the bias term, however, similar arguments also hold for a case with bias as illustrated in the supplementary file.



(a) (b)

Figure 2: Two dimensional illustration of (a) a spike-free case and (b) a non spike-free case. The extreme points (spikes) produce the piece-wise linear behavior in Figures 1b and 1c as predicted by our theory (see Lemma 2.7). The set shown in the rightmost figure acts as a structured regularizer.

$\mathbf{v}$ is achieved when $u_v = \pm 1$ and $b_v = -sign(u_v)a_i$ for a certain index $i \in [n]$.

Combined with Theorem 3.1, the above result proves that the optimal network outputs the linear spline interpolation for the input data. We now generalize the result for extreme points in the span of the ordinary basis vectors to higher dimensions. These will improve our spike-free relaxation as a first order correction. For instance, the behavior in Figure 2b is captured by the convex hull of the union of extreme points along $\mathbf{e}_1$ and $\mathbf{e}_2$, and the spike-free relaxation.

Lemma 2.7. *An extreme point in the span of each ordinary basis direction $\mathbf{e}_i$ is given by*

$$\mathbf{u}_i = \frac{\mathbf{a}_i - \sum_{\substack{j=1\\ j\neq i}}^{n} \lambda_j \mathbf{a}_j}{\left\|\mathbf{a}_i - \sum_{\substack{j=1\\ j\neq i}}^{n} \lambda_j \mathbf{a}_j\right\|_2} \text{ and } b_i = \min_{j\neq i}(-\mathbf{a}_j^T\mathbf{u}_i), \quad (9)$$

where $\boldsymbol{\lambda}$ is computed via the following problem

$$\min_{\boldsymbol{\lambda}} \left\|\mathbf{a}_i - \sum_{\substack{j=1\\ j\neq i}}^{n} \lambda_j \mathbf{a}_j\right\|_2 \text{ s.t. } \boldsymbol{\lambda} \succcurlyeq \mathbf{0}, \mathbf{1}^T\boldsymbol{\lambda} = 1.$$

Our next result characterizes extreme points along arbitrary directions for the general case.

Lemma 2.8. *For any $\boldsymbol{\alpha} \in \mathbb{R}^n$, the extreme point along the direction of $\boldsymbol{\alpha}$ can be found using*

$$\mathbf{u}_\alpha = \frac{\sum_{i\in\mathcal{S}}(\alpha_i+\lambda_i)\mathbf{a}_i - \sum_{j\in\mathcal{S}^c}\nu_j\mathbf{a}_j}{\|\sum_{i\in\mathcal{S}}(\alpha_i+\lambda_i)\mathbf{a}_i - \sum_{j\in\mathcal{S}^c}\nu_j\mathbf{a}_j\|_2}$$
$$b_\alpha = \begin{cases} \max_{i\in\mathcal{S}}(-\mathbf{a}_i^T\mathbf{u}), & \text{if } \sum_{i\in\mathcal{S}}\alpha_i \le 0 \\ \min_{j\in\mathcal{S}^c}(-\mathbf{a}_j^T\mathbf{u}), & \text{otherwise} \end{cases} \quad (10)$$

where the set of active and inactive ReLUs, i.e., $\mathcal{S}$ and $\mathcal{S}^c$, $\boldsymbol{\lambda}$, and $\boldsymbol{\nu}$ are obtained via the following problem

$$\min_{\boldsymbol{\lambda},\boldsymbol{\nu}} \max_{\mathbf{u},b} \mathbf{u}^T\left(\sum_{i\in\mathcal{S}}(\alpha_i+\lambda_i)\mathbf{a}_i - \sum_{j\in\mathcal{S}^c}\nu_j\mathbf{a}_j\right)$$
$$\text{s.t. } \boldsymbol{\lambda},\boldsymbol{\nu} \succcurlyeq 0, \sum_{i\in\mathcal{S}}(\alpha_i+\lambda_i) = \sum_{j\in\mathcal{S}^c}\nu_j, \|\mathbf{u}\|_2 = 1.$$

Lemma 2.8 proves that optimal neurons can be characterized as a linear combination of the data samples. Below, we further simplify this characterization and achieve a representer theorem for regularized NNs.

Corollary 2.1 (**A representer theorem for optimal neurons**). *Lemma 2.8 implies that each extreme point along the direction $\boldsymbol{\alpha}$ can be written in the following compact form*

$$\mathbf{u}_{\boldsymbol{\alpha}} = \frac{\sum_{i\in\mathcal{S}}\alpha_i(\mathbf{a}_i - \mathbf{a}_k)}{\|\sum_{i\in\mathcal{S}}\alpha_i(\mathbf{a}_i - \mathbf{a}_k)\|_2} \text{ and } b = -\mathbf{a}_k^T\mathbf{u}_\alpha$$

for some k and subset $\mathcal{S}$. Therefore, optimal neurons in the training objectives (2) and (4) all obey the above representation.

Remark 2.1. *We remark that an interpretation of the extreme points given above is autoencoding: the optimal neurons are convex mixture approximations of subsets of samples via other subsets.*

3 Main results

In the following, we present our main findings based on the extreme point characterization.

3.1 Convex duality

Theorem 3.1. *The dual of* (4) *is given by*

$$D^* = \max_{\mathbf{v}\in\mathbb{R}^n} \mathbf{v}^T\mathbf{y} \quad = \quad \max_{\mathbf{v}\in\mathbb{R}^n} \mathbf{v}^T\mathbf{y}\,, \tag{11}$$
$$\text{s.t. } \left|\mathbf{v}^T\big(\mathbf{A}\mathbf{u}\big)_+\right| \leq 1\,\forall \mathbf{u}\in\mathcal{B}_2 \quad \text{s.t. } \mathbf{v}\in\mathcal{Q}_{\mathbf{A}}^\circ, -\mathbf{v}\in\mathcal{Q}_{\mathbf{A}}^\circ$$

and we have $P^* \geq D^*$. For finite width NNs, there exists a large enough m such that we have strong duality, i.e., $P^* = D^*$, and an optimal $\mathbf{U}$ *for* (4) *satisfies* $\|(\mathbf{A}\mathbf{U}^*)_+^T\mathbf{v}^*\|_\infty = 1$, *where* $\mathbf{v}^*$ *is dual optimal.*[4]

Remark 3.1. *Note that* (11) *is a convex optimization problem with infinitely many constraints, and in general not polynomial-time tractable. In fact, even checking whether a point* $\mathbf{v}$ *is feasible is NP-hard: we need to solve* $\max_{\mathbf{u}:\|\mathbf{u}\|_2\leq 1}\sum_{i=1}^n v_i\big(\mathbf{a}_i^T\mathbf{u}\big)_+$. *This is related to the problem of learning halfspaces with noise, which is NP-hard to approximate within a constant factor (see e.g. (Guruswami and Raghavendra, 2009; Bach, 2017)).*

Corollary 3.1. *Theorem 3.1 implies that the optimal neuron weights are extreme points which solve*

$$\operatorname*{argmax}_{\mathbf{u}:\|\mathbf{u}\|_2\leq 1} |\mathbf{v}^{*T}\big(\mathbf{A}\mathbf{u}\big)_+|.$$

3.2 Structure of one dimensional networks

We are now ready to present our results on the structure induced by the extreme points. The following corollary directly follows from Lemma 2.6.

Corollary 3.2. *Let* $\{a_i\}_{i=1}^n$ *be a one dimensional training set i.e.,* $a_i \in \mathbb{R}$, $\forall i \in [n]$. *Then, a set of solutions to* (4) *that achieve the optimal value are extreme points, and therefore satisfy* $\{(u_i, b_i)\}_{i=1}^m$, *where* $u_i = \pm 1, b_i = -sign(u_i)a_i$.

Proposition 3.1. *The solution provided in Corollary 3.2 may not be unique. In the supplementary file, we present a counter-example where an optimal solution is not in this form, i.e., not a piecewise linear spline.*

3.3 Closed form solutions and ℓ_0-ℓ_1 equivalence

A considerable amount of literature have been published on the equivalence of minimal ℓ_1 and ℓ_0 solutions in under-determined linear systems, where it was shown that the equivalence holds under assumptions on the data matrices (see e.g. (Candes and Tao, 2005; Donoho, 2006; Fung and Mangasarian, 2011)). We now prove a similar equivalence for two-layer NNs. Consider the minimal cardinality problem

$$\min_{\theta\in\Theta}\|\mathbf{w}\|_0 \text{ s.t. } f_\theta(\mathbf{A}) = \mathbf{y}, \|\mathbf{u}_j\|_2 = 1, \forall j. \tag{12}$$

The following results provide a characterization of the optimal solutions to the above problem

Lemma 3.1. *Suppose that* $n \leq d$, $\mathbf{A}$ *is full row rank and* $\mathbf{y}$ *contains both positive and negative entries, and define* $\mathbf{A}^\dagger = \mathbf{A}^T(\mathbf{A}\mathbf{A}^T)^{-1}$. *Then an optimal solution to the problem in* (12) *is given by*

$$\mathbf{u}_1 = \frac{\mathbf{A}^\dagger\big(\mathbf{y}\big)_+}{\|\mathbf{A}^\dagger\big(\mathbf{y}\big)_+\|_2}\;,\; w_1 = \|\mathbf{A}^\dagger\big(\mathbf{y}\big)_+\|_2$$
$$\mathbf{u}_2 = \frac{\mathbf{A}^\dagger\big(-\mathbf{y}\big)_+}{\|\mathbf{A}^\dagger\big(-\mathbf{y}\big)_+\|_2},\; w_2 = -\|\mathbf{A}^\dagger\big(-\mathbf{y}\big)_+\|_2\,.$$

Lemma 3.2. *We have* ℓ_1-ℓ_0 *equivalence, i.e., the optimal solutions of* (12) *and* (4) *coincide if the following condition holds*

$$\min_{\mathbf{v}:\mathbf{v}^T\big(\mathbf{A}\mathbf{u}_1\big)_+=1,\mathbf{v}^T\big(\mathbf{A}\mathbf{u}_2\big)_+=-1}\quad\max_{\mathbf{u}:\|\mathbf{u}\|_2\leq 1}\left|\mathbf{v}^T\big(\mathbf{A}\mathbf{u}\big)_+\right| \leq 1\,.$$

Furthermore, whitened data matrices with $n \leq d$ *satisfy* ℓ_1-ℓ_0 *equivalence.*

3.4 A cutting plane method

In this section, we introduce a cutting plane based training algorithm for the NN in (1). Among infinitely many possible unit norm weights, we need to find the weights that violate the constraint in (11), which can be done by solving the following optimization problems

$$\mathbf{u}_1^* = \operatorname*{argmax}_{\mathbf{u}:\|\mathbf{u}\|_2\leq 1}\mathbf{v}^T(\mathbf{A}\mathbf{u})_+,\; \mathbf{u}_2^* = \operatorname*{argmin}_{\mathbf{u}:\|\mathbf{u}\|_2\leq 1}\mathbf{v}^T(\mathbf{A}\mathbf{u})_+. \tag{13}$$

However, (13) is not a convex problem. There exist several methods and relaxations to find the optimal parameters for (13). As an example, one can use the Frank-Wolfe algorithm (Frank and Wolfe, 1956) in order to approximate the solution iteratively. Here, we show how to relax the problem using our spike-free relaxation as follows

$$\hat{\mathbf{u}}_1 = \operatorname*{argmax}_{\mathbf{u}:\mathbf{A}\mathbf{u}\succcurlyeq\mathbf{0},\|\mathbf{u}\|_2\leq 1}\mathbf{v}^T\mathbf{A}\mathbf{u},\; \hat{\mathbf{u}}_2 = \operatorname*{argmin}_{\mathbf{u}:\mathbf{A}\mathbf{u}\succcurlyeq\mathbf{0},\|\mathbf{u}\|_2\leq 1}\mathbf{v}^T\mathbf{A}\mathbf{u}, \tag{14}$$

where we relax the set $\{(\mathbf{A}\mathbf{u})_+|\mathbf{u}\in\mathbb{R}^d, \|\mathbf{u}\|_2 \leq 1\}$ as $\{\mathbf{A}\mathbf{u}|\mathbf{u}\in\mathbb{R}^d, \|\mathbf{u}\|_2 \leq 1\} \cap \mathbb{R}_+^n$. Now, we can find the weights for the hidden layer using (14). In the cutting plane method, we first find a violating neuron using (14). After adding these parameters to $\mathbf{U}$ as columns, we solve (4). If we cannot find a new violating neuron then we terminate the algorithm. Otherwise, we find the dual parameter for the updated $\mathbf{U}$. We repeat this procedure till we find an optimal solution (see Algorithm 1 in the supplementary file for the pseudocode of the cutting-plane method).

[4] Similar results hold for other loss functions and vector output networks. We defer these results to the supplementary file and present our results in this simplified version.

Proposition 3.2. *When* $\mathbf{A}$ *is spike-free as defined in Lemma 2.3, the cutting plane based training method globally optimizes* (11)*.*

The following theorem shows that random high dimensional i.i.d. Gaussian matrices asymptotically satisfy the spike-free condition.

Theorem 3.2. *Let* $\mathbf{A} \in \mathbb{R}^{n \times d}$ *be an i.i.d. Gaussian random matrix. Then* $\mathbf{A}$ *is asymptotically spike-free as* $d \to \infty$*. More precisely, we have*

$$\lim_{d \to \infty} \mathbb{P}\left[\max_{\mathbf{u} \in \mathcal{B}_2} \|\mathbf{A}^\dagger \left(\mathbf{A}\mathbf{u}\right)_+\|_2 > 1\right] = 0\,.$$

We now consider improving the basic relaxation by including the extreme points in our relaxation, and provide some theoretical results.

Theorem 3.3. *Let* $\mathcal{C}_a$ *denote the convex hull of* $\{\mathbf{a}_i\}_{i=1}^n$*. If each sample is a vertex of* $\mathcal{C}_a$*, then a feasible solution to* (4) *can be achieved with* n *neurons, which are the extreme points along the ordinary basis vectors. Consequently, the weights given in Lemma 2.7 achieve zero training error.*

Our next result shows that the above condition is likely to hold high dimensional random matrices.

Theorem 3.4. *Let* $\mathbf{A} \in \mathbb{R}^{n \times d}$ *be a data matrix generated i.i.d. from a standard Gaussian distribution* $\mathcal{N}(0,1)$*. Suppose that the dimensions of the data matrix obey* $d > 2n\log(n-1)$*. Then, every row of* $\mathbf{A}$ *is an extreme point of the convex hull of the rows of* $\mathbf{A}$ *with high probability.*

4 Regularized two-layer ReLU networks

A penalized version can also be formulated instead of the equality form in (4). We next present a duality result for the penalized case.

Theorem 4.1. *An optimal* $\mathbf{U}$ *for the following regularized version of* (4) *given by*

$$\min_{\theta \in \Theta} \|(\mathbf{A}\mathbf{U})_+\mathbf{w} - \mathbf{y}\|_2^2 + \beta\|\mathbf{w}\|_1 \;\; s.t. \;\; \|\mathbf{u}_j\|_2 \leq 1, \forall j,$$

can be found through the following dual problem

$$\max_{\mathbf{v}} -\|\mathbf{v} - \mathbf{y}\|_2^2 \;\; s.t. \;\; \mathbf{v} \in \beta\mathcal{Q}_{\mathbf{A}}^\circ, -\mathbf{v} \in \beta\mathcal{Q}_{\mathbf{A}}^\circ\,,$$

where β *is the regularization (weight decay) parameter.*

5 Two-layer ReLU networks with hinge loss

Now we consider the classification problem with the hinge loss.

Theorem 5.1. *An optimal* $\mathbf{U}$ *for the binary classification task with the hinge loss given by*

$$\begin{aligned} &\min_{\theta \in \Theta} \sum_{i=1}^n \max\{0, 1 - y_i(\mathbf{a}_i^T\mathbf{U})_+\mathbf{w}\} + \beta\|\mathbf{w}\|_1 \qquad (15) \\ &s.t. \;\; \|\mathbf{u}_j\|_2 \leq 1, \forall j, \end{aligned}$$

can be found through the following dual

$$\begin{aligned} &\max_{\mathbf{v}} \mathbf{v}^T\mathbf{y} \\ &s.t. \;\; 0 \leq y_i v_i \leq 1 \; \forall i \in [n], \mathbf{v} \in \beta\mathcal{Q}_{\mathbf{A}}^\circ, -\mathbf{v} \in \beta\mathcal{Q}_{\mathbf{A}}^\circ\,. \end{aligned}$$

Consequently, in the 1D case, the optimal neuron weights are given by the extreme points as a result of Theorem 3.1. Therefore the optimal network network output is given by the piecewise linear function

$$f(\mathbf{a}) = \sum_{j=1}^m w_j(\mathbf{a}u_j + b_j)_+\,,$$

for some output weights $w_1, \ldots, w_m$ where $u_j = \pm 1$ and $b_j = \mp a_j$ for some j.

This explains Figure 1c, where the decision regions are determined by the zero crossings of the above piecewise linear function. Moreover, the dual problem reduces to a finite dimensional minimum ℓ_1 norm Support Vector Machine (SVM), whose solution can be easily determined. As it can be seen in Figure 1c, the piecewise linear fit passes through the data samples which are on the margin, i.e., the network output is ±1. This corresponds to the maximum margin decision regions and separates the green shaded area from the red shaded area.

It is straightforward to see that this is equivalent to applying the kernel map $\kappa(a, a_j) = (a - a_j)_+$, forming the corresponding kernel matrix

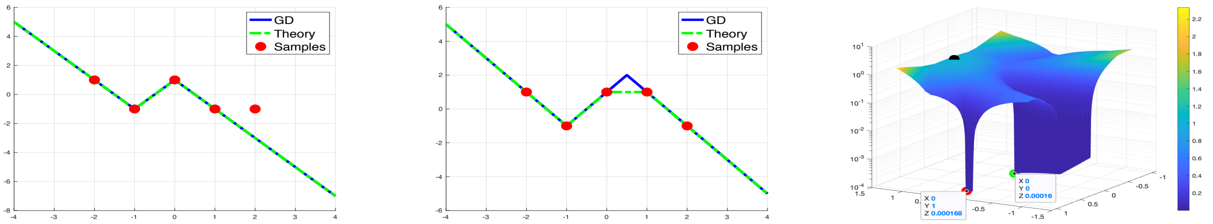
$$\mathbf{K}_{ij} = (a_i - a_j)_+\,,$$

and solving minimum ℓ_1-norm SVM on the kernelized data matrix.

Theorem 5.2. *For a one dimensional dataset* $\mathbf{a} \in \mathbb{R}^n$*, applying* ℓ_1*-norm SVM on* $\left(\mathbf{a}{\mathbf{u}^*}^T + \mathbf{1}{\mathbf{b}^*}^T\right)_+$ *finds the optimal solution* θ^* *to* (15)*, where* $\mathbf{u}^*$ *and* $\mathbf{b}^*$ *are* $2n-2$ *dimensional vectors defined as* $\{u_i^* = \pm 1, b_i^* = -sign(u_i^*)a_i\}_{i=2}^{n-1}$*,* $\{u_n^* = -1, b_n^* = a_n\}$*, and* $\{u_1^* = 1, b_n^* = -a_1\}$ [5]*.*

The proof directly follows from Theorem 3.1 and Lemma 2.6. We also verify Theorem 5.2 using the experiments in Figure 3. In this figure, we observe that whenever there is a sign change, the corresponding two

[5] Notice that we do not include $\{u_n^* = 1, b_n^* = -a_n\}$ and $\{u_1^* = -1, b_1^* = a_1\}$ since they output a vector of zeros, which are obviously not extreme points.



(a) $L_t = 1.600 \times 10^{-4}$ and $L_{gd} = 1.600 \times 10^{-4}$. GD and our theory agrees.

(b) $L_t = 1.600 \times 10^{-4}$ and $L_{gd} = 1.679 \times 10^{-4}$. GD is stuck at a local minima.

(c) Visualization of the loss landscape in (b) ($L_t = 1.600 \times 10^{-4}$ and $L_{gd} = 1.679 \times 10^{-4}$).

Figure 3: Binary classification using hinge loss, where we apply GD and our approach in Theorem 5.2. Here, we denote the objective value in (15) as L_t and L_{gd} for our theoretical approach (Theorem 5.2) and GD, respectively. In (c), we also provide 3D illustration of the loss surface of the example in (b), where we mark the initial point (black), the GD solution (red), and our solution (green).

Table 1: Classification Accuracies (%) and test errors

	MNIST	CIFAR-10	Bank	Boston	California	Elevators	News20	Stock
`One Layer NN (Least Squares)`	86.04%	36.39%	0.9258	0.3490	0.8158	0.5793	1.0000	1.0697
`Two-Layer NN (Backpropagation)`	96.25%	41.57 %	0.6440	0.1612	0.8101	0.4021	0.8304	0.8684
`Two-Layer NN Convex`	96.94%	42.16%	**0.5534**	**0.1492**	**0.6344**	**0.3757**	**0.8043**	**0.6184**
`Two-Layer Convex-RF`	**97.72%**	**80.28%**	-	-	-	-	-	-

samples determine the decision boundary, which resembles the idea of support vector. Thus, the piecewise linear fit passes through these samples. On the other hand, when there is no sign change, the piecewise fit does not create any kink as in Figure 3a. We also observe that GD might fail to globally optimize (15) unlike our approach as illustrated in Figure 3b. In Figure 3c, we also provide a visualization of the loss landscape for this case.

6 Numerical experiments

We first consider classification tasks and report the performance of the algorithms on MNIST (LeCun) and CIFAR-10 (Krizhevsky et al., 2014) [6]. For these data sets, we do not perform any preprocessing except a normalization on the pixels in MNIST so that each pixel is in the range $[0, 1]$. In Table 1, we observe that our approach denoted as `Convex`, which is solely based on convex optimization techniques, outperforms the non-convex backpropagation based approach. In addition, we use an alternative approach, denoted as `Convex-RF` in Table 1 which uses (9) on image patches[7]. This unsupervised training approach for the hidden layer surprisingly increases the accuracy by almost 40% compared to the convex approach with the cutting plane algorithm. We also evaluate the performances on several regression data sets, namely Bank, Boston Housing, California Housing, Elevators, Stock (Torgo), and the Twenty Newsgroups text classification data set (Mitchell and Learning, 1997). In Table 1, we provide the test errors for each approach. Here, our convex approach outperforms the backpropagation, and the one layer NN approach in each case.

7 Concluding remarks

We have studied two-layer ReLU networks via a convex analytic framework that explains why simple solutions are achieved even when networks are overparameterized. In particular, we showed that the extreme points characterize simple structures and explain why training of regularized NNs yields a linear spline interpolation in 1D. Using these observations, we have also provided a training algorithm based on cutting planes, which achieves global optimality under certain assumptions. We conjecture that similar extreme point characterizations in deep networks may explain their extraordinary generalization properties.

References

Francis Bach. Breaking the curse of dimensionality with convex neural networks. *The Journal of Ma-*

[6] We use a generalized vector output version of our method discussed in the supplementary material.

[7] Further information about our experimental setup can be found in the supplementary material.

chine Learning Research, 18(1):629–681, 2017.

Yoshua Bengio, Nicolas L Roux, Pascal Vincent, Olivier Delalleau, and Patrice Marcotte. Convex neural networks. In *Advances in neural information processing systems*, pages 123–130, 2006.

Guy Blanc, Neha Gupta, Gregory Valiant, and Paul Valiant. Implicit regularization for deep neural networks driven by an ornstein-uhlenbeck like process. *CoRR*, abs/1904.09080, 2019. URL `http://arxiv.org/abs/1904.09080`.

E. J. Candes and T. Tao. Decoding by linear programming. *IEEE Trans. Info Theory*, 51(12):4203–4215, December 2005.

Lenaic Chizat and Francis Bach. A note on lazy training in supervised differentiable programming. *arXiv preprint arXiv:1812.07956*, 2018.

D. L. Donoho. For most large underdetermined systems of linear equations, the minimal ℓ_1-norm solution is also the sparsest solution. *Communications on Pure and Applied Mathematics*, 59(6):797–829, June 2006.

Marguerite Frank and Philip Wolfe. An algorithm for quadratic programming. *Naval Research Logistics Quarterly*, 3(1-2):95–110, 1956. doi: 10.1002/nav.3800030109. URL `https://onlinelibrary.wiley.com/doi/abs/10.1002/nav.3800030109`.

GM Fung and OL Mangasarian. Equivalence of minimal ℓ_0-and ℓ_p-norm solutions of linear equalities, inequalities and linear programs for sufficiently small p. *Journal of optimization theory and applications*, 151(1):1–10, 2011.

Venkatesan Guruswami and Prasad Raghavendra. Hardness of learning halfspaces with noise. *SIAM Journal on Computing*, 39(2):742–765, 2009.

Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. The CIFAR-10 dataset. `http://www.cs.toronto.edu/kriz/cifar.html`, 2014.

Yann LeCun. The MNIST database of handwritten digits. `http://yann.lecun.com/exdb/mnist/`.

Hartmut Maennel, Olivier Bousquet, and Sylvain Gelly. Gradient descent quantizes relu network features. *arXiv preprint arXiv:1803.08367*, 2018.

Tom M Mitchell and Machine Learning. Mcgraw-hill science. *Engineering/Math*, 1:27, 1997.

Behnam Neyshabur, Ryota Tomioka, and Nathan Srebro. In search of the real inductive bias: On the role of implicit regularization in deep learning. *arXiv preprint arXiv:1412.6614*, 2014.

R. T. Rockafellar. *Convex Analysis.* Princeton University Press, Princeton, 1970.

Pedro Savarese, Itay Evron, Daniel Soudry, and Nathan Srebro. How do infinite width bounded norm networks look in function space? *CoRR*, abs/1902.05040, 2019. URL `http://arxiv.org/abs/1902.05040`.

L. Torgo. Regression data sets. `http://www.dcc.fc.up.pt/~ltorgo/Regression/DataSets.html`.

Colin Wei, Jason D Lee, Qiang Liu, and Tengyu Ma. On the margin theory of feedforward neural networks. *arXiv preprint arXiv:1810.05369*, 2018.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. *arXiv preprint arXiv:1611.03530*, 2016.

Peng Zhao and Bin Yu. On model selection consistency of lasso. *Journal of Machine learning research*, 7 (Nov):2541–2563, 2006.