Convex Duality and Cutting Plane Methods for Over-parameterized Neural Networks

Tolga Ergen ERGEN@STANFORD.EDU
Mert Pilanci PILANCI@STANFORD.EDU
Stanford University, Stanford, CA 94305

Abstract

We develop a convex analytic framework for ReLU networks and study their function space characteristics. We show that the optimal parameters of two layer over-parameterized networks can be described as extreme points of a certain convex set. For one dimensional regression, our theory explains why ReLU networks yield linear spline interpolation. In higher dimensions, we show that the training problem can be cast as a convex optimization problem with infinitely many constraints. We then provide convex relaxations to approximate the solution, and a cutting-plane algorithm to improve the relaxations.

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 [13, 18]. In [18], 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 is largely unknown.

A line of research [4, 13, 23] explored the behavior of ReLU networks in finite size cases. In [23], the authors indicated that NNs are implicitly regularized during training since Stochastic Gradient Descent (SGD) converges to a solution with small norm. The idea of implicit regularization was also extended to the networks trained with GD as well as SGD. Particularly, the authors in [13] showed that implicit regularization has a strong connection with the initialization of a network and showed 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 the given dataset. In order to explain generalization capabilities of ReLU networks, another line of research in [2, 3, 6, 22] focused on infinitely wide two layer ReLU networks. In [3], the authors introduced an algorithm that can train a regularized NN with infinite width in an incremental manner. In [22], 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.

Our contributions in this work are as follows: 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, we prove that hidden layers form a linear spline interpolation. We also provide an intuitive convex geometric explanation of this fact; 3) We provide a convex relaxation based training procedure, which is proven to be exact under certain assumptions on the training set.

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 $\mathbf{0}$ 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 ReLU. We use the notation $[n]$ to denote the set of integers from 1 to 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 $f(\mathbf{A}) = (\mathbf{A}\mathbf{U} + \mathbf{1}\mathbf{b}^T)_+\mathbf{w}$. We emphasize that all the derivations in the sequel can be extended to a vector case with o outputs. In this case, we have $f(\mathbf{A}) = \left(\mathbf{A}\mathbf{U} + \mathbf{1}\mathbf{b}^T\right)_+\mathbf{W}$, where $\mathbf{W} \in \mathbb{R}^{m\times o}$ [1].

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 [13]. Given $\mathbf{A}$, the label vector $\mathbf{y} \in \mathbb{R}^n$, $\theta \in \Theta = \{(\mathbf{U}, \mathbf{b}, \mathbf{w}) \,|\, \mathbf{U} \in \mathbb{R}^{d\times m}, \mathbf{b} \in \mathbb{R}^m, \mathbf{w} \in \mathbb{R}^m\}$, and $R(\theta) = \|\mathbf{w}\|_2^2 + \|\mathbf{U}\|_F^2$, we consider the following problem

$$\min_{\theta\in\Theta} R(\theta) \text{ s.t. } f_\theta(\mathbf{A}) = \mathbf{y}, \tag{2}$$

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 suitable rescaling. This result was also presented in [15, 18].

Lemma 2.1 ([15, 18]) *The following two optimization problems are equivalent:*

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

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 (2) as

$$\min_{\theta\in\Theta} \|\mathbf{w}\|_1 \text{ s.t. } f_\theta(\mathbf{A}) = \mathbf{y}, \|\mathbf{u}_j\|_2 \leq 1, \forall j. \tag{3}$$

[1] We refer the reader to appendix for details.

However, (3) is a quite challenging optimization problem 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 function might considerably change as detailed in the next section.

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

$$\min_{\mathbf{u}_1} \left\| w_1 (\mathbf{A}\mathbf{u}_1)_+ - \mathbf{y} \right\|_2^2 \text{ s.t. } \|\mathbf{u}_1\|_2 \leq 1. \tag{4}$$

The solution of (4) is completely determined by the behavior of 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 (4) 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 first 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 $\mathbf{A}$.

2.1. Rectified ellipsoid and spike-free matrices

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. However, there exists data matrices $\mathbf{A}$ for which the set $\mathcal{Q}_{\mathbf{A}}$ is convex, e.g., diagonal data matrices. However, the aforementioned set of matrices are, in fact, a more general class.

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 $\mathbb{R}^n_+ \cap \{\mathbf{A}\mathbf{u} | \mathbf{u} \in \mathbb{R}^d, \|\mathbf{u}\|_2 \leq 1\}$, then (4) can be solved via convex optimization after the rescaling $\mathbf{u} = \mathbf{u}_1 w_1$

$$\min_{\mathbf{u}} \left\| \mathbf{A}\mathbf{u} - \mathbf{y} \right\|_2^2 \text{ s.t. } \mathbf{u} \in \{\mathbf{A}\mathbf{u} \nsucceq 0\} \cup \{-\mathbf{A}\mathbf{u} \nsucceq 0\}, \|\mathbf{u}\|_2 \leq 1 \,.$$

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*

$$\forall \mathbf{u} \in \mathcal{B}_2, \; \exists \mathbf{z} \in \mathcal{B}_2 \textit{ such that we have } \left(\mathbf{A}\mathbf{u}\right)_+ = \mathbf{A}\mathbf{z} \,. \tag{5}$$

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

$$\max_{\mathbf{u} \,:\, \|\mathbf{u}\|_2 \leq 1} \|\mathbf{A}^\dagger \left(\mathbf{A}\mathbf{u}\right)_+\|_2 \leq 1 \,. \tag{6}$$

The following lemmas show that whitened matrices with $n \leq d$ are spike-free.

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

2.2. Polar convex duality

It can be shown that the dual of the problem (3) is given by[2]

$$\max_{\mathbf{v}} \mathbf{v}^T\mathbf{y} \text{ s.t. } \mathbf{v} \in \mathcal{Q}_{\mathbf{A}}^{\circ}\,, \; -\mathbf{v} \in \mathcal{Q}_{\mathbf{A}}^{\circ} \tag{7}$$

where $\mathcal{Q}_{\mathbf{A}}^{\circ}$ is the polar set [16] of $\mathcal{Q}_{\mathbf{A}}$ defined as $\mathcal{Q}_{\mathbf{A}}^{\circ} = \{\mathbf{v} | \mathbf{v}^T\mathbf{u} \leq 1 \,\forall \mathbf{u} \in \mathcal{Q}_{\mathbf{A}}\}$.

2.3. Extreme Points

We first define the extreme point of $\mathcal{Q}_{\mathbf{A}}$ along $\mathbf{v}$ as $\operatorname{argmax}_{\mathbf{z}\in\mathcal{Q}_{\mathbf{A}}} \mathbf{v}^T\mathbf{z}$. Here, we show that the extreme points of $\mathcal{Q}_{\mathbf{A}}$ are given by data samples and convex mixtures of data samples in 1D.

Lemma 2.5 *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 $\mathbf{v}$ 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, we prove that the optimal network outputs the linear spline interpolation.

3. Main results

3.1. Convex duality

Theorem 3.1 *The dual of the problem in (3) is given by*

$$\begin{aligned} D^* = \max_{\mathbf{v}\in\mathbb{R}^n} \mathbf{v}^T\mathbf{y} \qquad\qquad\qquad &= \qquad \max_{\mathbf{v}\in\mathbb{R}^n} \mathbf{v}^T\mathbf{y}\,, \\ \text{s.t. } \left|\mathbf{v}^T\left(\mathbf{A}\mathbf{u}\right)_+\right| \leq 1 \,\forall \mathbf{u} \in \mathcal{B}_2 \qquad & \qquad \text{s.t. } \mathbf{v} \in \mathcal{Q}_{\mathbf{A}}^{\circ}, -\mathbf{v} \in \mathcal{Q}_{\mathbf{A}}^{\circ} \end{aligned} \tag{8}$$

and we have $P^* \geq D^*$. For infinite size NNs[3] *($m \to \infty$) we have strong duality, i.e., $P^* = D^*$, and an optimal $\mathbf{U}$ for (3) satisfies $\|(\mathbf{A}\mathbf{U}^*)_+^T\mathbf{v}^*\|_\infty = 1$, where $\mathbf{v}^*$ is dual optimal.*[4]

Remark 1 *Note that (8) 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 $\max_{\mathbf{u}:\|\mathbf{u}\|_2\leq 1} \sum_{i=1}^n v_i \left(\mathbf{a}_i^T\mathbf{u}\right)_+$. This is related to the problem of learning halfspaces with noise, which is NP-hard to approximate within a constant factor (see e.g. [2, 9]).*

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

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

We are now ready to present our results on the structure induced by the extreme points. The following corollary directly follows from Lemma 2.5.

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 (3) that achieve the optimal value are extreme points, and therefore satisfy $\{(u_i, b_i)\}_{i=1}^m$, where $u_i = \pm 1, b_i = -sign(u_i)a_i$.*

[2] We refer the reader to appendix 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 appendix.

[3] We refer the reader to appendix for a rigorous definition and the details of infinite size networks.

[4] Similar results hold for other loss functions (e.g. hinge loss), penalized versions and vector output networks. Here we present our results in this simplified version.

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**

3.2. 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 inequality constraint in (8). However, this is not a convex problem since ReLU is a convex function. There exist several methods and relaxations to find the optimal parameters. Here, we show how to relax the problem using our spike-free relaxation as follows

$$\hat{\mathbf{u}}_1 = \operatorname*{argmax}_{\mathbf{u}:\mathbf{A}\mathbf{u}\succcurlyeq\mathbf{0},\|\mathbf{u}\|_2\le 1} \mathbf{v}^T\mathbf{A}\mathbf{u} \qquad \hat{\mathbf{u}}_2 = \operatorname*{argmin}_{\mathbf{u}:\mathbf{A}\mathbf{u}\succcurlyeq\mathbf{0},\|\mathbf{u}\|_2\le 1} \mathbf{v}^T\mathbf{A}\mathbf{u}, \tag{9}$$

where we relax the set $\{(\mathbf{A}\mathbf{u})_+ | \mathbf{u} \in \mathbb{R}^d, \|\mathbf{u}\|_2 \le 1\}$ as $\{\mathbf{A}\mathbf{u} | \mathbf{u} \in \mathbb{R}^d, \|\mathbf{u}\|_2 \le 1\} \cap \mathbb{R}^n_+$. Now, we can find the weights for the hidden layer using (9). In the cutting plane method, we first find a violating neuron using (9). After adding these parameters to $\mathbf{U}$ as columns, we solve (3). If we cannot find a new violating neuron then we terminate the algorithm. Otherwise, we first find the dual parameter for the updated $\mathbf{U}$. We then repeat this procedure till we find an optimal solution (see Algorithm 1 in appendix for the pseudocode of the cutting-plane method).

Proposition 3.1 *When* $\mathbf{A}$ *is spike-free, the cutting plane based method globally optimizes* (8).

4. Numerical experiments

We first consider classification tasks and report the performance[5] of the algorithms on MNIST [12] and CIFAR-10 [10] 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. We also evaluate the performances on several regression data sets [14, 20]. In Table 1, we provide the test errors for each approach. Here, our approach outperforms the backpropagation, and the Least Squares approach.

5. 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 over-parameterized. 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 a cutting plane method, which achieves global optimality under certain assumptions.

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

References

[1] 20 newsgroups. http://qwone.com/~jason/20Newsgroups/.

[2] Francis Bach. Breaking the curse of dimensionality with convex neural networks. *The Journal of Machine Learning Research*, 18(1):629–681, 2017.

[3] 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.

[4] 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.

[5] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

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

[7] Miguel Angel Goberna and Marco López-Cerdá. *Linear semi-infinite optimization*. 01 1998. doi: 10.1007/978-1-4899-8044-1_3.

[8] Michael Grant and Stephen Boyd. CVX: Matlab software for disciplined convex programming, version 2.1. http://cvxr.com/cvx, March 2014.

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

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

[11] Anders Krogh and John A Hertz. A simple weight decay can improve generalization. In *Advances in neural information processing systems*, pages 950–957, 1992.

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

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

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

[15] 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.

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

[17] Walter Rudin. *Principles of Mathematical Analysis*. McGraw-Hill, New York, 1964.

[18] 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.

[19] Andreas Themelis and Panagiotis Patrinos. Supermann: a superlinearly convergent algorithm for finding fixed points of nonexpansive operators. *IEEE Transactions on Automatic Control*, 2019.

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

[21] E. van den Berg and M. P. Friedlander. SPGL1: A solver for large-scale sparse reconstruction, June 2007. http://www.cs.ubc.ca/labs/scl/spgl1.

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

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

Appendix

In this section, we present proofs of the main results, details on the algorithms and numerical results and extra figures that are not included in the main paper due to the page limit.

Appendix A. Additional details on the numerical experiments

In this section, we provide further information about our experimental setup:

In the main paper, we evaluate the performance of the introduced approach on several real data sets. For comparison, we also include the performance of a two layer NN trained with the backpropagation algorithm and the well-known linear least squares approach. For all the experiments, we use the regularization term (also known as weight decay) to let the algorithms generalize well on unseen data [11]. In order to solve the convex optimization problems in our approach, we use CVX [8]. However, notice that when dealing with large data sets, e.g., CIFAR-10, plain CVX solvers might take a lot of time or give memory errors. In order to circumvent these issues, we use SPGL1 [21] and SuperSCS [19] for large data sets. We also remark that all the data sets we use are publicly available and further information, e.g., training and test sizes, can be obtained through the provided references [1, 10, 12, 20]. Furthermore, we use the same number of hidden neurons for both our approach and the conventional backpropagation based approach to have a fair comparison.

Appendix B. Cutting plane algorithm with no bias term

In this section, we present the pseudocode for the algorithm provided in the main paper in the case of no bias term.

In the cutting plane method, we first find a violating neuron using (9). After adding these parameters to $\mathbf{U}$ as columns, we solve (3). 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).

Algorithm 1 Cutting Plane based Training Algorithm for Two Layer NNs (without bias)

```
Initialize $\mathbf{v} = \mathbf{y}$
while there exists a violating neuron do
    Find $\hat{\mathbf{u}}_1$ and $\hat{\mathbf{u}}_2$ via (9)
    $\mathbf{U} \leftarrow [\mathbf{U}\ \hat{\mathbf{u}}_1\ \hat{\mathbf{u}}_2]$
    Find $\mathbf{v}$ using the dual problem in (8)
    Check the existence of a violating neuron via (9)
end while
Solve (3) using $\mathbf{U}$
Return $\theta = (\mathbf{U}, \mathbf{w})$
```

Appendix C. Cutting plane algorithm with a bias term

Here, we include the cutting plane algorithm which accommodates a bias term. This is slightly more involved than the case with no bias because of extra constraints. We have the corresponding dual problem as in Theorem 3.1

$$\max_{\mathbf{v}\in\mathbb{R}^n,\, \mathbf{1}^T\mathbf{v}=0} \mathbf{v}^T\mathbf{y} \text{ s.t. } \left|\mathbf{v}^T\left(\mathbf{A}\mathbf{u} + b\mathbf{1}\right)_+\right| \leq 1\,, \forall \mathbf{u} \in \mathcal{B}_2 \tag{10}$$

and an optimal $\mathbf{U}$ and $\mathbf{b}$ satisfies

$$\|(\mathbf{A}\mathbf{U}^* + \mathbf{1}\mathbf{b}{*^T})_+^T\mathbf{v}^*\|_\infty = 1\,,$$

where $\mathbf{v}^*$ is the optimal dual variable.

Among infinitely many possible unit norm weights, we need to find the weights that violate the inequality constraint in the dual form, which can be done by solving the following optimization problems

$$\mathbf{u}_1^* = \arg\max_{\mathbf{u},b} \mathbf{v}^T(\mathbf{A}\mathbf{u} + b\mathbf{1})_+ \text{ s.t. } \|\mathbf{u}\|_2 \leq 1$$
$$\mathbf{u}_2^* = \arg\min_{\mathbf{u},b} \mathbf{v}^T(\mathbf{A}\mathbf{u} + b\mathbf{1})_+ \text{ s.t. } \|\mathbf{u}\|_2 \leq 1.$$

However, the above problem is not convex since ReLU is a convex function. In this case, we can further relax the problem by applying the spike-free relaxation as follows

$$(\hat{\mathbf{u}}_1, \hat{b}_1) = \arg\max_{\mathbf{u},b} \mathbf{v}^T\mathbf{A}\mathbf{u} + b\mathbf{v}^T\mathbf{1} \text{ s.t. } \mathbf{A}\mathbf{u} + b\mathbf{1} \succcurlyeq \mathbf{0}, \|\mathbf{u}\|_2 \leq 1$$
$$(\hat{\mathbf{u}}_2, \hat{b}_2) = \arg\min_{\mathbf{u},b} \mathbf{v}^T\mathbf{A}\mathbf{u} + b\mathbf{v}^T\mathbf{1} \text{ s.t. } \mathbf{A}\mathbf{u} + b\mathbf{1} \succcurlyeq \mathbf{0}, \|\mathbf{u}\|_2 \leq 1,$$

where we relax the set $\{(\mathbf{A}\mathbf{u} + b\mathbf{1})_+ | \mathbf{u} \in \mathbb{R}^d, \|\mathbf{u}\|_2 \leq 1\}$ as $\{\mathbf{A}\mathbf{u} + b\mathbf{1} | \mathbf{u} \in \mathbb{R}^d, \|\mathbf{u}\|_2 \leq 1\} \cap \mathbb{R}_+^n$. Now, we can find the weights and biases for the hidden layer using convex optimization. However, notice that depending on the sign of $\mathbf{1}^T\mathbf{v}$ one of the problems will be unbounded. Thus, if $\mathbf{1}^T\mathbf{v} \neq 0$, then we can always find a violating constraint, which will make the problem infeasible. Note that $\mathbf{1}^T\mathbf{v} = 0$ will be enforced via the dual.

Based on our analysis, we propose the following convex optimization approach to train the two layer NN. We first find a violating neuron. After adding these parameters to $\mathbf{U}$ as a column and to $\mathbf{b}$ as a row, we try to solve the original problem. 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 until the optimality conditions are satisfied (see Algorithm 2 for the pseudocode). Since the constraint is bounded below and $\hat{\mathbf{u}}_j$'s are bounded, Algorithm 2 is guaranteed to converge in finitely many iterations Theorem 11.2 of [7].

Algorithm 2 Cutting Plane based Training Algorithm for Two Layer NNs (with bias)

```
Initialize $\mathbf{v}$ such that $\mathbf{1}^T\mathbf{v} = 0$
while there exists a violating neuron do
    Find $\hat{\mathbf{u}}_1$, $\hat{\mathbf{u}}_2$, $\hat{b}_1$ and $\hat{b}_2$
    $\mathbf{U} \leftarrow [\mathbf{U}\ \hat{\mathbf{u}}_1\ \hat{\mathbf{u}}_2]$
    $\mathbf{b} \leftarrow [\mathbf{b}^T\ \hat{b}_1\ \hat{b}_2]^T$
    Find $\mathbf{v}$ using the dual problem
    Check the existence of a violating neuron
end while
Solve the problem using $\mathbf{U}$ and $\mathbf{b}$
Return $\theta = (\mathbf{U}, \mathbf{b}, \mathbf{w})$
```

Appendix D. Infinite size neural networks

Here we briefly review infinite size, i.e., infinite width, two layer NNs [2]. We refer the reader to [3, 22] for further background and connections to our work. Consider an arbitrary measurable input space $\mathcal{X}$ with a set of continuous basis functions $\phi_{\mathbf{u}} : \mathcal{X} \to \mathbb{R}$ parametrized by $\mathbf{u} \in \mathcal{B}_2$. We then consider real-valued Radon measures equipped with the uniform norm [17]. For a signed Radon measure $\boldsymbol{\mu}$, we define the infinite size neural network output for the input $\mathbf{x} \in \mathcal{X}$ as

$$f(\mathbf{x}) = \int_{\mathbf{u}\in\mathcal{B}_2} \phi_{\mathbf{u}}(\mathbf{x}) d\boldsymbol{\mu}(\mathbf{u})\,.$$

The total variation norm of the signed measure $\boldsymbol{\mu}$ is defined as the supremum of $\int_{\mathbf{u}\in\mathcal{B}_2} q(\mathbf{u}) d\boldsymbol{\mu}(\mathbf{u})$ over all continuous functions $q(\mathbf{u})$ that satisfy $|q(\mathbf{u})| \leq 1$. Now we consider the ReLU basis functions $\phi_{\mathbf{u}}(\mathbf{x}) = \left(\mathbf{x}^T\mathbf{u}\right)_+$. For finitely many neurons as in (2), the network output is given by

$$f(\mathbf{x}) = \sum_{j=1}^{m} \phi_{\mathbf{u}_j}(\mathbf{x}) w_j\,,$$

which corresponds to the signed measure $\boldsymbol{\mu} = \sum_{j=1}^{m} w_j \delta(\mathbf{u} - \mathbf{u}_j)$ where δ is the Dirac delta function. And the total variation norm $\|\boldsymbol{\mu}\|_{TV}$ of $\boldsymbol{\mu}$ reduces to the ℓ_1 norm $\|\mathbf{w}\|_1$.

The infinite dimensional version of the problem (3) corresponds to

$$\begin{aligned} &\min \|\boldsymbol{\mu}\|_{TV} \\ &\text{s.t. } \ f(\mathbf{x}_i) = y_i\,, \forall i \in [n]\,. \end{aligned}$$

For finitely many neurons, i.e., when the measure $\boldsymbol{\mu}$ is a mixture of Dirac delta basis functions, the equivalent problem is

$$\begin{aligned} &\min \|\mathbf{w}\|_1 \\ &\text{s.t. } \; f(\mathbf{x}_i) = y_i \, , \forall i \in [n] \, . \end{aligned}$$

which is identical to (3). Similar results also hold with regularized objective functions, different loss functions and vector outputs.

Appendix E. Proofs of the main results

In this section, we present the proofs of the theorems and lemmas provided in the main paper.
Proof of Lemma 2.1 For any $\theta \in \Theta$, we can rescale the parameters as $\bar{\mathbf{u}}_j = \alpha_j \mathbf{u}_j$, $\bar{b}_j = \alpha_j b_j$ and $\bar{w}_j = w_j/\alpha_j$, for any $\alpha_j > 0$. Then, (1) becomes

$$f_{\bar{\theta}}(\mathbf{A}) = \sum_{j=1}^m \bar{w}_j (\mathbf{A}\bar{\mathbf{u}}_j + \bar{b}_j \mathbf{1})_+ = \sum_{j=1}^m \frac{w_j}{\alpha_j} (\alpha_j \mathbf{A}\mathbf{u}_j + \alpha_j b_j \mathbf{1})_+ = \sum_{j=1}^m w_j (\mathbf{A}\mathbf{u}_j + b_j \mathbf{1})_+,$$

which proves $f_\theta(\mathbf{A}) = f_{\bar{\theta}}(\mathbf{A})$. In addition to this, we have the following basic inequality

$$\sum_{j=1}^m (w_j^2 + \|\mathbf{u}_j\|_2^2) \geq 2 \sum_{j=1}^m (|w_j| \, \|\mathbf{u}_j\|_2),$$

where the equality is achieved with the scaling choice $\alpha_j = \big(\frac{|w_j|}{\|\mathbf{u}_j\|_2}\big)^{\frac{1}{2}}$. Since the scaling operation does not change the right-hand side of the inequality, we can set $\|\mathbf{u}_j\|_2 = 1, \forall j$. Therefore, the right-hand side becomes $\|\mathbf{w}\|_1$. ■

Proof of Lemma 2.2 Consider the following problem

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

where the unit norm equality constraint is relaxed. Let us assume that for a certain index j, we obtain $\|\mathbf{u}_j\|_2 \leq 1$ with $w_j \neq 0$ as the optimal solution of the above problem. This shows that the unit norm inequality constraint is not active for $\mathbf{u}_j$, and hence removing the constraint for $\mathbf{u}_j$ will not change the optimal solution. However, when we remove the constraint, $\|\mathbf{u}_j\|_2 \to \infty$ reduces the objective value since it yields $w_j = 0$. Hence, we have a contradiction, which proves that all the constraints that correspond to a nonzero w_j must be active for an optimal solution. ■

Proof of Lemma 2.3 The first condition immediately implies that $\{\big(\mathbf{A}\mathbf{u}\big)_+ | \mathbf{u} \in \mathcal{B}_2\} \subseteq \mathbf{A}\mathcal{B}_2$. Since we also have $\{\big(\mathbf{A}\mathbf{u}\big)_+ | \mathbf{u} \in \mathcal{B}_2\} \subseteq \mathbb{R}_+^n$, it holds that $\{\big(\mathbf{A}\mathbf{u}\big)_+ | \mathbf{u} \in \mathcal{B}_2\} \subseteq \mathbf{A}\mathcal{B}_2 \cap \mathbb{R}_+^n$. Since the projection of $\mathbf{A}\mathcal{B}_2 \cap \mathbb{R}_+^n$ onto the positive orthant is a subset of $\mathcal{Q}_\mathbf{A}$, we have $\mathcal{Q}_\mathbf{A} = \mathbf{A}\mathcal{B}_2 \cap \mathbb{R}_+^n$.

The second condition follows from the min-max representation

$$\max_{\mathbf{u} \in \mathcal{B}_2} \; \min_{\mathbf{z} \colon \mathbf{A}\mathbf{z} = (\mathbf{A}\mathbf{u})_+} \|\mathbf{z}\|_2 \leq 1 \iff (5)\,,$$

and the fact that the minimum norm solution to $\mathbf{A}\mathbf{z} = (\mathbf{A}\mathbf{u})_+$ is given by $\mathbf{A}^\dagger(\mathbf{A}\mathbf{u})_+$ under the full row rank assumption on $\mathbf{A}$. ■

Proof of Lemma 2.4 We have

$$\begin{aligned}\max_{\mathbf{u}\,:\,\|\mathbf{u}\|_2\leq 1} \|\mathbf{A}^T(\mathbf{A}\mathbf{A}^T)^{-1}\big(\mathbf{A}\mathbf{u}\big)_+\|_2 &\leq \sigma_{\max}(\mathbf{A}^T(\mathbf{A}\mathbf{A}^T)^{-1}) \max_{\mathbf{u}\,:\,\|\mathbf{u}\|_2\leq 1} \|\big(\mathbf{A}\mathbf{u}\big)_+\|_2 \\ &= \sigma_{\min}^{-1}(\mathbf{A}) \max_{\mathbf{u}\,:\,\|\mathbf{u}\|_2\leq 1} \|\big(\mathbf{A}\mathbf{u}\big)_+\|_2 \\ &\leq \sigma_{\min}^{-1}(\mathbf{A}) \max_{\mathbf{u}\,:\,\|\mathbf{u}\|_2\leq 1} \|\mathbf{A}\mathbf{u}\|_2 \\ &\leq \sigma_{\min}^{-1}(\mathbf{A})\sigma_{\max}(\mathbf{A}) \\ &\leq 1\,.\end{aligned}$$

where the last inequality follows from the fact that $\mathbf{A}$ is whitened. ■

Proof of Lemma 2.5 The extreme point along the direction of $\mathbf{v}$ can be found as follows

$$\max_{u,b} \sum_{i=1}^{n} v_i(a_i u + b)_+ \text{ s.t. } |u| = 1, \tag{11}$$

Since each neuron separates the samples into two sets, for some samples, ReLU will be active, i.e., $\mathcal{S} = \{i|a_i u + b \geq 0\}$, and for the others, it will be inactive, i.e., $\mathcal{S}^c = \{j|a_j u + b < 0\} = [n]/\mathcal{S}$. Thus, we modify (11) as

$$\max_{u,b} \sum_{i\in\mathcal{S}} v_i(a_i u + b) \text{ s.t. } a_i u + b \geq 0, \forall i \in \mathcal{S}, a_j u + b \leq 0, \forall j \in \mathcal{S}^c, |u| = 1. \tag{12}$$

In (12), u can only take two values, i.e., ± 1. Thus, we can separately solve the optimization problem for each case and then take the maximum one as the optimal. Let us assume that $u = 1$. Then, (12) reduces to finding the optimal bias. We note that due to the constraints in (12), $-a_i \leq b \leq -a_j, \forall i \in \mathcal{S}, \forall j \in \mathcal{S}^c$. Thus, the range for the possible bias values is $[\max_{i\in\mathcal{S}}(-a_i),\ \min_{j\in\mathcal{S}^c}(-a_j)]$. Therefore, depending on the direction $\mathbf{v}$, the optimal bias can be selected as follows

$$b_v = \begin{cases} \max_{i\in\mathcal{S}}(-a_i), & \text{if } \sum_{i\in\mathcal{S}} v_i \leq 0 \\ \min_{j\in\mathcal{S}^c}(-a_j), & \text{otherwise} \end{cases}. \tag{13}$$

Similar arguments also hold for $u = -1$ and the min version of (11). ■

Proof of Theorem 3.1 and Corollary 3.1

Using an indicator function, we can reformulate the problem as follows

$$P^* = \min_{\theta\in\Theta\backslash\{\mathbf{w}\}} \max_{\mathbf{v}} \mathbf{v}^T\mathbf{y} + \mathcal{I}(\|(\mathbf{A}\mathbf{U})_+^T\mathbf{v}\|_\infty \leq 1), \text{ s.t. } \|\mathbf{u}_j\|_2 \leq 1, \forall j,$$

where $\mathcal{I}(x \leq a) = 0$, if $x \leq a$, $\mathcal{I}(x \leq a) = -\infty$, otherwise. Since the set $\|(\mathbf{A}\mathbf{U})_+^T\mathbf{v}\|_\infty \leq 1$ is closed, the function $\Phi(\mathbf{v}, \mathbf{U}) := \mathbf{v}^T\mathbf{y} + \mathcal{I}(\|(\mathbf{A}\mathbf{U})_+^T\mathbf{v}\|_\infty \leq 1)$ is the sum of a linear function and an upper-semicontinuous indicator function and therefore upper-semicontinuous. The constraint over $\mathbf{U}$

is convex and compact. We use P^* to denote the value of the above $\min$-$\max$ program. Exchanging the order of $\min$ and $\max$ we obtain the dual problem given in (8), which establishes a lower bound D^* for the above problem:

$$P^* \geq D^* = \max_{\mathbf{v}} \min_{\theta \in \Theta \backslash \{\mathbf{w}\}} \mathbf{v}^T \mathbf{y} + \mathcal{I}(\|(\mathbf{A}\mathbf{U})_+^T \mathbf{v}\|_\infty \leq 1), \text{ s.t. } \|\mathbf{u}_j\|_2 \leq 1, \forall j,$$

We now show that strong duality holds for infinite size NNs. The dual of the semi-infinite program in (8) is given by (see Section 2.2 of [7] and also [2])

$$\begin{aligned} &\min \|\boldsymbol{\mu}\|_{TV} \\ &\text{s.t.} \int_{\mathbf{u} \in \mathcal{B}_2} \big(\mathbf{A}\mathbf{u}\big)_+ \boldsymbol{\mu}(d\mathbf{u}) = \mathbf{y}\,, \end{aligned}$$

where TV is the total variation norm of the Radon measure $\boldsymbol{\mu}$. This expression coincides with the infinite-size neural network as given in Section D, and therefore strong duality holds. Next we invoke the semi-infinite optimality conditions for the dual problem in (8), in particular we apply Theorem 7.2 of [7]. We first define the set

$$\mathbf{K} = \mathbf{cone}\left\{ \left(\begin{array}{c} s\big(\mathbf{A}\mathbf{u}\big)_+ \\ 1 \end{array} \right), \mathbf{u} \in \mathcal{B}_2, s \in \{-1, +1\}; \left(\begin{array}{c} \mathbf{0} \\ -1 \end{array} \right) \right\}.$$

Note that $\mathbf{K}$ is the union of finitely many convex closed sets, since the function $\big(\mathbf{A}\mathbf{u}\big)_+$ can be expressed as the union of finitely many convex closed sets. Therefore the set $\mathbf{K}$ is closed. By Theorem 5.3 [7], this implies that the set of constraints in (8) forms a Farkas-Minkowski system. By Theorem 8.4 of [7], primal and dual values are equal, given that the system is consistent. Moreover, the system is discretizable, i.e., there exists a sequence of problems with finitely many constraints whose optimal values approach to the optimal value of (8). The optimality conditions in Theorem 7.2 [7] implies that $\mathbf{y} = \big(\mathbf{A}\mathbf{U}^*\big)_+ \mathbf{w}^*$ for some vector $\mathbf{w}^*$. Since the primal and dual values are equal, we have $\mathbf{v}^{*T}\mathbf{y} = \mathbf{v}^{*T}\big(\mathbf{A}\mathbf{U}^*\big)_+ \mathbf{w}^* = \|\mathbf{w}^*\|_1$, which shows that the primal-dual pair $(\{\mathbf{w}^*, \mathbf{U}^*\}, \mathbf{v}^*)$ is optimal. Thus, the optimal neuron weights $\mathbf{U}^*$ satisfy $\|(\mathbf{A}\mathbf{U}^*)_+^T \mathbf{v}^*\|_\infty = 1$. ■

Proof of Proposition 3.1 Since the constraint in (9) is bounded below and the hidden layer weights are constrained to the unit Euclidean ball, the convergence of the cutting plane method directly follows from Theorem 11.2 of [7]. ■

Appendix F. Polar convex duality

In this section we derive the polar duality and present a connection to minimum ℓ_1 solutions to linear systems. Recognizing the constraint $\mathbf{v} \in \mathcal{Q}_\mathbf{A}$ can be stated as

$$\mathbf{v} \in \mathcal{Q}_\mathbf{A}^\circ, \, \mathbf{v} \in -\mathcal{Q}_\mathbf{A}^\circ\,,$$

which is equivalent to

$$\mathbf{v} \in \mathcal{Q}_\mathbf{A}^\circ \cap -\mathcal{Q}_\mathbf{A}^\circ\,.$$

Note that the support function of a set can be expressed as the gauge function of its polar set (see e.g. [16]). The polar set of $\mathcal{Q}_{\mathbf{A}}^{\circ} \cap -\mathcal{Q}_{\mathbf{A}}^{\circ}$ is given by

$$\big(\mathcal{Q}_{\mathbf{A}}^{\circ} \cap -\mathcal{Q}_{\mathbf{A}}^{\circ}\big)^{\circ} = \mathbf{Co}\big(\mathcal{Q}_{\mathbf{A}} \cup -\mathcal{Q}_{\mathbf{A}}\big)\,.$$

Using this fact, we express the dual problem (8) as

$$\begin{aligned} D^* = &\inf_{t\in\mathbb{R}}\ t \\ &\text{s.t. } \mathbf{y} \in t\mathbf{Co}\big(\mathcal{Q}_{\mathbf{A}} \cup -\mathcal{Q}_{\mathbf{A}}\big)\,, \end{aligned} \tag{14}$$

where $\mathbf{Co}\big(\,\big)$ represents the convex hull of a set.

Let us restate dual of the two layer ReLU neural network training problem given by

$$\max_{\mathbf{v}} \mathbf{v}^T\mathbf{y} \text{ s.t. } \mathbf{v} \in \mathcal{Q}_{\mathbf{A}}^{\circ}\,,\ -\mathbf{v} \in \mathcal{Q}_{\mathbf{A}}^{\circ} \tag{15}$$

where $\mathcal{Q}_{\mathbf{A}}^{\circ}$ is the polar dual of $\mathcal{Q}_{\mathbf{A}}$ defined as $\mathcal{Q}_{\mathbf{A}}^{\circ} = \{\mathbf{v} | \mathbf{v}^T\mathbf{u} \le 1\, \forall \mathbf{u} \in \mathcal{Q}_{\mathbf{A}}\}\,.$

Remark 1 *The dual problem given in* (15) *is analogous to the convex duality in minimum* ℓ_1 *norm solutions to linear systems. In particular, for the latter it holds that*

$$\min_{\mathbf{w}\,:\,\mathbf{A}\mathbf{w}=\mathbf{y}} \|\mathbf{w}\|_1 = \max_{\mathbf{v}\in\mathbf{Co}\big(\{\big)\hat{\mathbf{a}}_1,\ldots,\hat{\mathbf{a}}_d\}^{\circ},\ -\mathbf{v}\in\mathbf{Co}\big(\{\big)\hat{\mathbf{a}}_1,\ldots,\hat{\mathbf{a}}_d\}^{\circ}} \mathbf{v}^T\mathbf{y}\,,$$

where $\hat{\mathbf{a}}_1, \ldots, \hat{\mathbf{a}}_d$ *are the columns of* $\mathbf{A}$. *The above optimization problem can also be put in the gauge optimization form as follows.*

$$\min_{\mathbf{w}\,:\,\mathbf{A}\mathbf{w}=\mathbf{y}} \|\mathbf{w}\|_1 = \inf_{t\in\mathbb{R}}\ t \textit{ s.t. } \mathbf{y} \in t\mathbf{Co}\big(\{\big) \pm \hat{\mathbf{a}}_1, \ldots\hat{\mathbf{a}}_d\},$$

which parallels the gauge optimization form in (14).

Appendix G. Regularized two layer ReLU networks

A penalized version can also be formulated instead of the equality form in (3). We next present a duality result for the penalized case.

Theorem G.1 *An optimal* $\mathbf{U}$ *for the following regularized version of* (3) *given by*

$$\min_{\theta\in\Theta} \frac{1}{2}\|(\mathbf{A}\mathbf{U})_+\mathbf{w} - \mathbf{y}\|_2^2 + \beta\|\mathbf{w}\|_1 \textit{ s.t. } \|\mathbf{u}_j\|_2 \le 1, \forall j, \tag{16}$$

as $m \to \infty$ *can be found through the following dual problem*

$$\max_{\mathbf{v}} -\frac{1}{2}\|\mathbf{v} - \mathbf{y}\|_2^2 + \frac{1}{2}\|\mathbf{y}\|_2^2 \textit{ s.t. } \mathbf{v} \in \beta\mathcal{Q}_{\mathbf{A}}^{\circ}, -\mathbf{v} \in \beta\mathcal{Q}_{\mathbf{A}}^{\circ}\,,$$

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

The proof follows from a similar argument as in the proof of Theorem 3.1, and is omitted.

Appendix H. Two layer ReLU networks with general loss functions

Now we consider the scalar output two layer ReLU networks with an arbitrary loss function

$$\min_{\theta\in\Theta} \ell(\mathbf{A}\mathbf{U})_+\mathbf{w},\mathbf{y}) + \beta\|\mathbf{w}\|_1 \text{ s.t. } \|\mathbf{u}_j\|_2 \leq 1, \forall j, \tag{17}$$

where $\ell(\cdot,\mathbf{y})$ is a convex loss function.

Theorem 1 *The dual of* (17) *is given by*

$$\max_{\mathbf{v}} -\ell^*(\mathbf{v}) \textit{ s.t. } \mathbf{v} \in \beta\mathcal{Q}^\circ_{\mathbf{A}}, -\mathbf{v} \in \beta\mathcal{Q}^\circ_{\mathbf{A}}\,,$$

where ℓ^* is the Fenchel conjugate function defined as

$$\ell^*(\mathbf{v}) = \max_{\mathbf{z}} \mathbf{z}^T\mathbf{v} - \ell(\mathbf{z},\mathbf{y})\,.$$

The proof follows from classical Fenchel duality [5], and a similar argument as in the proof of Theorem 3.1. We omit the details. The general form of the dual can be easily extended to vector output networks.

Appendix I. Extension to vector output neural networks

In this section, we describe the implementation of the cutting plane algorithm when we have vector outputs, particularly, o outputs. In this case, we have $\mathbf{Y} \in \mathbb{R}^{n\times o}$ and $f(\mathbf{A}) = \big(\mathbf{A}\mathbf{U}\big)_+\mathbf{W}$, where $\mathbf{W} \in \mathbb{R}^{m\times o}$. Then, we formulate the following dual problem

$$\max_{\mathbf{V}} \mathbf{tr}(\mathbf{V}^T\mathbf{Y}) \text{ s.t. } \|\mathbf{V}^T\big(\mathbf{A}\mathbf{u}\big)_+\|_\infty \leq 1\,, \forall \mathbf{u} \in \mathcal{B}_2$$

and an optimal $\mathbf{U}$ satisfies

$$\|(\mathbf{A}\mathbf{U}^*)_+^T\mathbf{V}^*\|_\infty = 1\,,$$

where $\mathbf{V}^*$ is the optimal dual variable and **tr** represents the trace of a matrix. Note that we can also consider block ℓ_1-ℓ_2 norms and their duals in formulating the vector output objective. We use this particular form as it admits a simpler solution with the cutting-plane method.

We again relax the problem using the spike-free relaxation and then we solve the following problem for each $k \in [o]$

$$\hat{\mathbf{u}}_{k,1} = \arg\max_{\mathbf{u}} \mathbf{v}_k^T\mathbf{A}\mathbf{u} \text{ s.t. } \mathbf{A}\mathbf{u} \succcurlyeq \mathbf{0}, \|\mathbf{u}\|_2 \leq 1$$
$$\hat{\mathbf{u}}_{k,2} = \arg\min_{\mathbf{u}} \mathbf{v}_k^T\mathbf{A}\mathbf{u} \text{ s.t. } \mathbf{A}\mathbf{u} \succcurlyeq \mathbf{0}, \|\mathbf{u}\|_2 \leq 1,$$

where $\mathbf{v}_k$ is the k^{th} column of $\mathbf{V}$. After solving these optimization problems, we select the two neurons that achieve the maximum and minimum objective value among o neurons for each problem. Thus, we can find the weights for the hidden layers using convex optimization.