Journal of Machine Learning Research 25 (2024) 1-38 Submitted 10/23; Revised 7/24; Published 10/24

Memorization With Neural Nets: Going Beyond the Worst Case

Sjoerd Dirksen S.DIRKSEN@UU.NL
Mathematical Institute
Utrecht University
3584 CD Utrecht, Netherlands

Patrick Finke P.G.FINKE@UU.NL
Mathematical Institute
Utrecht University
3584 CD Utrecht, Netherlands

Martin Genzel* MARTIN.GENZEL@MERANTIX-MOMENTUM.COM
Merantix Momentum GmbH
13355 Berlin, Germany

Editor: Mahdi Soltanolkotabi

Abstract

In practice, deep neural networks are often able to easily interpolate their training data. To understand this phenomenon, many works have aimed to quantify the memorization capacity of a neural network architecture: the largest number of points such that the architecture can interpolate any placement of these points with any assignment of labels. For real-world data, however, one intuitively expects the presence of a benign structure so that interpolation already occurs at a smaller network size than suggested by memorization capacity. In this paper, we investigate interpolation by adopting an instance-specific viewpoint. We introduce a simple randomized algorithm that, given a fixed finite data set with two classes, with high probability constructs an interpolating three-layer neural network in polynomial time. The required number of parameters is linked to geometric properties of the two classes and their mutual arrangement. As a result, we obtain guarantees that are independent of the number of samples and hence move beyond worst-case memorization capacity bounds. We verify our theoretical result with numerical experiments and additionally investigate the effectiveness of the algorithm on MNIST and CIFAR-10.

Keywords: memorization, interpolation, neural networks, random hyperplane tessellations, high-dimensional geometry

1. Introduction

The *bias-variance tradeoff* (Shalev-Shwartz and Ben-David, 2014; Hastie et al., 2009) has been a cornerstone of classical machine learning theory that illustrates the relationship between the bias of a model and its variance, and how they affect its generalization performance. It states that if the model is too simple (high bias), it may underfit as it does not capture the underlying patterns in the data. However, if it is too complex (high variance), it may overfit noise in the training data and fail to generalize well. The resulting conventional

*. Work done while at Utrecht University.

wisdom was to adjust the model complexity to achieve a balance between underfitting and overfitting, which would then lead to good generalization.

This classical viewpoint has been uprooted by modern practice in deep learning, where it is common to use heavily overparameterized neural networks that fit the used training data (almost) perfectly. In spite of this (near-)perfect fit, these models can generalize well to new data. In fact, it can be observed that as the model complexity increases, the test error first decreases, then increases (as predicted by the bias-variance trade-off), and then decreases again. This phenomenon, coined the *double descent phenomenon* (Belkin et al., 2019), is well documented not only for deep neural networks but for a wide range of machine learning methods, see, e.g., Belkin et al. (2019); Belkin (2021); Nakkiran et al. (2021); Mei and Montanari (2022); Hastie et al. (2022). The second descent of the test error is observed at the *interpolation threshold*, where the model has become complex enough to interpolate the training samples. Thus, to gain a deeper understanding of double descent it is important to identify at which size a neural network can interpolate finitely many samples.

To determine the interpolation threshold, we may look at the literature on the *memorization capacity* of neural networks, which quantifies the number of parameters and neurons necessary for a network to be able to interpolate *any* N data points with *arbitrary* labels. Thus, memorization capacity offers a worst-case quantitative analysis of the interpolation threshold. In this analysis, 'the network architecture comes first and the data comes later'. As a result, the required network complexity for memorization scales in terms of the number of training data (see Section 1.3 for more details). In practical applications, however, 'the data comes first and the network architecture comes later': the neural network architecture and size are tuned to given training data via cross-validation. Intuitively, one expects that the training data possesses some 'nice' structure so that interpolation is achievable with a smaller network complexity than suggested by memorization capacity—which assumes arbitrary data and arbitrary labels.

In this paper, we investigate interpolation by adopting an instance-specific viewpoint. We introduce a simple randomized algorithm that, given a fixed finite data set with two classes, with high probability constructs an interpolating neural network in polynomial time, see Theorem 4. We then link the required number of parameters to the *mutual complexity* of the data set, which depends on both the geometric properties of two data classes as well as their mutual arrangement. As a result, we obtain guarantees that are independent of the number of samples and instead yield a 'problem-adaptive' bound on the interpolation threshold. Finally, we carry out numerical simulation experiments to illustrate our theoretical result. In addition, we investigate the effectiveness of our interpolation algorithm on MNIST and CIFAR-10.

1.1 Summary of Results

Let us first formalize the concept of interpolation in a classification setting with two classes. The setting with binary labels is considered for simplicity—our results can be readily extended to multiple classes by a one-versus-many approach (see Section 4.4 for details). In the following, $\mathcal{X}^-, \mathcal{X}^+ \subset R\mathbb{B}_2^d$ denote disjoint and finite sets, representing two classes of objects, where $\mathbb{B}_2^d$ denotes the unit Euclidean ball in $\mathbb{R}^d$.

Definition 1 (Interpolation) *We say that a classification function* $F\colon \mathbb{R}^d \to \{\pm 1\}$ interpolates $\mathcal{X}^-$ *and* $\mathcal{X}^+$ *if, for all* $\boldsymbol{x}^- \in \mathcal{X}^-$ *and* $\boldsymbol{x}^+ \in \mathcal{X}^+$,

$$F(\boldsymbol{x}^-) = -1 \quad \textit{and} \quad F(\boldsymbol{x}^+) = +1.$$

In this work, we will formulate a concrete, randomized algorithm that takes $\mathcal{X}^-$ and $\mathcal{X}^+$ as inputs and produces an interpolating neural net as an output (Algorithm 1). As the statement of the algorithm requires some technical preparation, we postpone its discussion to Section 2. Our main result, informally stated as Theorem 4 below and developed in full detail in Section 2, shows that this algorithm succeeds with high probability in polynomial time and provides bounds on the size of the interpolating network. The bounds are phrased in terms of two structural assumptions on the data, that together quantify the difficulty of the interpolation problem.

First, we will assume that the classes are *δ-separated.* This assumption is also common in a number of works on memorization capacity, e.g., Vershynin (2020); Rajput et al. (2021); Vardi et al. (2022). Below we will write, for any sets $\mathcal{A}, \mathcal{B} \subset \mathbb{R}^d$,

$$\mathrm{d}(\boldsymbol{a}, \mathcal{B}) = \inf_{\boldsymbol{b}\in\mathcal{B}} \|\boldsymbol{a} - \boldsymbol{b}\|_2, \qquad \mathrm{d}(\mathcal{A}, \mathcal{B}) = \inf_{\boldsymbol{a}\in\mathcal{A}} \mathrm{d}(\boldsymbol{a}, \mathcal{B}).$$

Definition 2 (δ-separation) $\mathcal{A}$ *and* $\mathcal{B}$ *are* δ-separated *if* $\mathrm{d}(\mathcal{A}, \mathcal{B}) \geq \delta$.

Second, we will quantify the problem difficulty using the following notion that was first introduced in Dirksen et al. (2022a) (in a slightly different form).

Definition 3 (Mutual covering) *We call*

$$\mathcal{C}^- = \{\boldsymbol{c}_1^-, \ldots, \boldsymbol{c}_{M^-}^-\} \subset \mathcal{X}^-, \quad r_1^-, \ldots, r_{M^-}^- \geq 0,$$
$$\mathcal{C}^+ = \{\boldsymbol{c}_1^+, \ldots, \boldsymbol{c}_{M^+}^+\} \subset \mathcal{X}^+, \quad r_1^+, \ldots, r_{M^+}^+ \geq 0$$

a mutual covering *for* $\mathcal{X}^-$ *and* $\mathcal{X}^+$ *if the sets*

$$\mathcal{X}_\ell^- := \mathcal{X}^- \cap \mathbb{B}_2^d(\boldsymbol{c}_\ell^-, r_\ell^-) \quad \textit{and} \quad \mathcal{X}_j^+ := \mathcal{X}^+ \cap \mathbb{B}_2^d(\boldsymbol{c}_j^+, r_j^+),$$

for $\ell \in [M^-]$ *and* $j \in [M^+]$, *cover* $\mathcal{X}^-$ *and* $\mathcal{X}^+$, *respectively. We call these sets the* components *of the mutual covering and call* M^- *and* M^+ *the* mutual covering numbers.

As we have only finitely many inputs, clearly a mutual covering always exists. However, if the arrangement of the classes is benign, the mutual covering numbers can be much smaller than the number of samples.

To see how the notion of mutual covering allows us to quantify the difficulty of a (binary) interpolation problem, we turn to our main result. In Theorem 4 we require the existence of a mutual covering with radii

$$r_\ell^- \lesssim \frac{\mathrm{d}(\boldsymbol{c}_\ell^-, \mathcal{C}^+)}{\log^{1/2}(eR/\mathrm{d}(\boldsymbol{c}_\ell^-, \mathcal{C}^+))} \quad \text{and} \quad r_j^+ \lesssim \frac{\mathrm{d}(\boldsymbol{c}_j^+, \mathcal{C}^-)}{\log^{1/2}(eR/\mathrm{d}(\boldsymbol{c}_j^+, \mathcal{C}^-))}. \tag{1}$$

Geometrically, this means that the components covering $\mathcal{X}^-$ and $\mathcal{X}^+$ cannot intersect (more precisely, need to be slightly separated from) the ideal decision boundary between the two

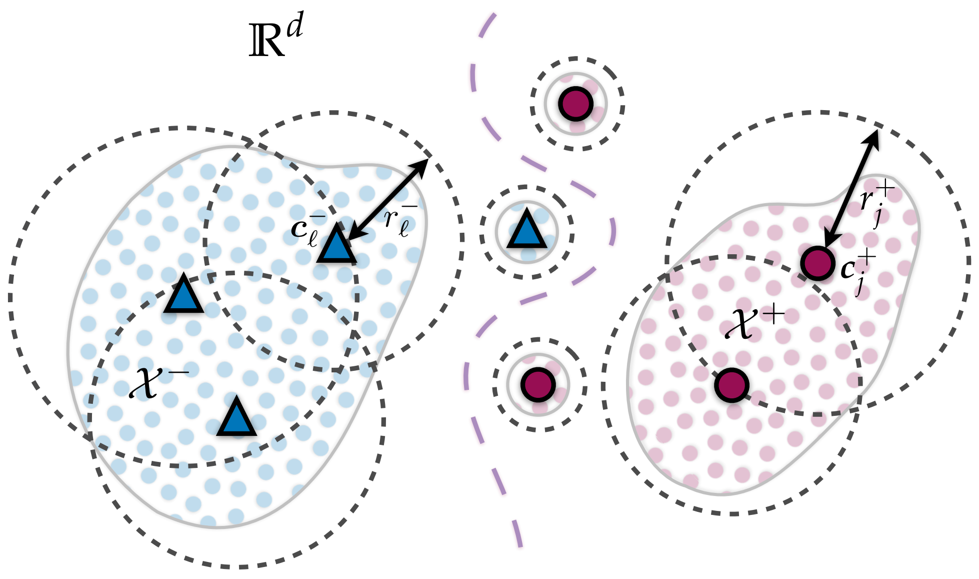


Figure 1: **The mutual covering is 'problem-adaptive'.** Condition (1) on the radii in Theorem 4 allows a covering 'adapted to' the mutual arrangement of the data: only the parts of the data that lie close to the ideal decision boundary need to be covered using balls with small diameters—other parts can be crudely covered using larger balls.

sets, as is illustrated in Figure 1. In particular, components with a small radius are only needed close to the ideal decision boundary, while parts that are far away from this boundary can be crudely covered with large components. Compared to classical coverings with balls of a fixed radius (as used in the classical notion of the Euclidean covering number of a set, see, e.g., Vershynin, 2018), this can drastically reduce the required number of components.

While the mutual covering numbers M^- and M^+ can be viewed as a measure of the *global complexity* of the data, our result also involves the *local complexity*, measured by the 'sizes' of the components. Specifically, define $\omega := \max\{\omega^-, \omega^+\}$ where

$$\omega^- := \max_{\ell\in[M^-]} \frac{w^2(\mathcal{X}_\ell^- - \boldsymbol{c}_\ell^-)}{\mathrm{d}^3(\boldsymbol{c}_\ell^-, \mathcal{C}^+)} \quad \text{and} \quad \omega^+ := \max_{j\in[M^+]} \frac{w^2(\mathcal{X}_j^+ - \boldsymbol{c}_j^+)}{\mathrm{d}^3(\boldsymbol{c}_j^+, \mathcal{C}^-)}. \tag{2}$$

The quantities ω^- and ω^+ measure the scaled version of the 'size' of the largest (centered) component of $\mathcal{X}^-$ and $\mathcal{X}^+$, respectively. Here, the *Gaussian mean width* of a set $\mathcal{A} \subset \mathbb{R}^d$ is defined as

$$w(\mathcal{A}) := \mathbb{E} \sup_{\boldsymbol{x}\in\mathcal{A}} |\langle \boldsymbol{g}, \boldsymbol{x}\rangle|,$$

where $\boldsymbol{g} \sim N(\boldsymbol{0}, \boldsymbol{I}_d)$ denotes a standard Gaussian random vector. The mean width is a well-established complexity measure in high-dimensional statistics and geometry which is sensitive to low-dimensional structures such as sparsity, unions of low-dimensional subspaces, or manifolds, see, e.g., Vershynin (2018) for a detailed discussion and examples. We refer to Remark 12 for straightforward estimates of ω.

We are now ready to present the informal version of our main result, which we state for the case of threshold activations. Intuitively, this should be the most challenging because the signal amplitude is lost. It is possible to prove analogous results for other activations. In fact, Algorithm 1 only requires an activation function σ such that $\sigma(t) = 0$ for $t \leq 0$ and $\sigma(t) > 0$ for $t > 0$, which, e.g., includes the ReLU. Note, however, that the bounds on the network size may change for different activations.

Theorem 4 (Informal) *Let $\mathcal{X}^-, \mathcal{X}^+ \subset R\mathbb{B}_2^d$ be finite and disjoint. Suppose that there is a mutual covering with δ-separated centers and radii satisfying* (1)*. Then, with high probability, Algorithm 1 terminates in polynomial time and outputs a 2-hidden-layer fully-connected neural network with threshold activations,*

$$\mathcal{O}\left(M^- + R\delta^{-1}\log(2M^-M^+) + R\omega\right)$$

neurons and

$$\mathcal{O}\left(R(d + M^-)(\delta^{-1}\log(2M^-M^+) + \omega)\right)$$

parameters, that interpolates $\mathcal{X}^-$ and $\mathcal{X}^+$.

A first interesting feature of this result is the asymmetric dependence on the complexities of the classes: the network size depends linearly on M^- but only logarithmically on M^+. As it is possible to interchange the roles of $\mathcal{X}^-$ and $\mathcal{X}^+$, we may always think of $\mathcal{X}^-$ as the 'smaller' set. Second, our bounds are independent of the number of samples. This is a fundamental difference between memorization capacity and our instance-specific approach to interpolation, see the discussion in Section 1.3. To highlight this second point further, we deduce an interpolation result for infinite sets from our analysis. In contrast to Theorem 4, the proof is nonconstructive.

Corollary 5 *Let $\mathcal{X}^-, \mathcal{X}^+ \subset R\mathbb{B}_2^d$ be (possibly infinite) sets. Suppose that there is a mutual covering with δ-separated centers and radii satisfying* (1)*. Then, there exists a neural network of the same size as in Theorem 4 that interpolates $\mathcal{X}^-$ and $\mathcal{X}^+$.*

1.2 Organization

The rest of the paper is organized as follows. In Section 1.3 we discuss related works, then introduce notation in Section 1.4. In Section 2 we present Algorithm 1 and give intuition on how it works. We also state the formal counterpart of our main result in Theorem 11. All proofs are contained in Section 3. In Section 4, we verify our theoretical findings in illustrative numerical experiments and additionally investigate the performance of our algorithm on real data sets. We conclude with a short summary in Section 5.

1.3 Related Works

Memorization capacity. Neural network architectures used in practice are powerful memorizers: it has been observed that various popular architectures for image classification do not only interpolate their training data, but can even interpolate this data when the labels are replaced by random labels (after re-training), see, e.g., Zhang et al. (2021a). To understand this phenomenon, an extensive literature has studied the memorization capacity of neural networks, by quantifying how large a network needs to be to interpolate any N points with *arbitrary labels*. In this case, we will say that the network can memorize N points. In practice, memorization results often include some assumptions on the inputs. Here we will summarize relevant memorization literature that makes similar structural assumptions on the inputs, such as δ-separation or a bound on the norm. Other works consider randomized samples or samples drawn from a distribution, see, e.g., Ge et al. (2019); Daniely (2020); Zhang et al. (2021b).

The study of the memorization capacity of neural networks with threshold activations has a rich history. Assuming that the points are in general position,[1] Baum (1988) showed that a 1-hidden-layer threshold network with $\mathcal{O}(N+d)$ parameters and $\mathcal{O}(\lceil N/d \rceil)$ neurons is enough to memorize binary labels of N points in $\mathbb{R}^d$. In Huang and Huang (1990) it was shown that $\mathcal{O}(Nd)$ parameters and $\mathcal{O}(N)$ neurons are enough to memorize real labels, without placing any additional constraints on the points. Assuming that the points are δ-separated and lie on the unit sphere, Vershynin (2020) proved that a deep threshold (or ReLU) network can memorize binary labels using $\widetilde{\mathcal{O}}(e^{1/\delta^2}(d+\sqrt{N})+N)$ parameters and $\widetilde{\mathcal{O}}(e^{1/\delta^2}+\sqrt{N})$ neurons. The exponential dependence on δ was improved by Rajput et al. (2021), who proved that $\widetilde{\mathcal{O}}(d/\delta+N)$ parameters and $\widetilde{\mathcal{O}}(1/\delta+\sqrt{N})$ neurons are enough for memorization of binary labels, while further only requiring bounded norm instead of unit norm. The constructions of both Vershynin (2020) and Rajput et al. (2021) are probabilistic, while the ones of Baum (1988) and Huang and Huang (1990) are purely deterministic.

There have been a number of works on the memorization capacity of networks with other activations. We will only summarize the results for ReLU activations due to its popularity in practice, and refer to, e.g., Huang (2003); Park et al. (2021); Madden and Thrampoulidis (2024) and the references therein for other activations. The work Bubeck et al. (2020) extended the result of Baum (1988) to the case of real-valued labels using a network with ReLU activation with a size of the same order. Using weight sharing in the first layer, Zhang et al. (2021a) showed that a 1-hidden-layer ReLU network could memorize real-valued labels using $\mathcal{O}(N+d)$ parameters and $\mathcal{O}(N)$ neurons, with no further assumptions on the points. Yun et al. (2019) proved that both multi-class and real-valued labels can be memorized by a ReLU net with two and three hidden layers, respectively, using $\mathcal{O}(d\sqrt{N}+N)$ parameters and $\mathcal{O}(\sqrt{N})$ neurons. Park et al. (2021) achieved the first result on memorization with a sublinear number of parameters: assuming that the points are separated, they showed that ReLU (or hard-tanh) nets can memorize multiple classes using $\widetilde{\mathcal{O}}(d+N^{2/3})$ parameters, constant width and $\widetilde{\mathcal{O}}(N^{2/3})$ layers. Vardi et al. (2022) improved the above dependence on N from $N^{2/3}$ to $\sqrt{N}$, which is optimal. Specifically, assuming that the points are δ-separated and have bounded norm, they show that a ReLU net with $\widetilde{\mathcal{O}}(d+\sqrt{N})$ parameters, constant width and $\widetilde{\mathcal{O}}(\sqrt{N})$ layers is enough to memorize multi-class labels.

To directly compare the above with our results, we consider a *trivial* mutual covering that always 'works' regardless of the labels of the points: we cover each point by its own component with a radius of zero. Thus, $M^- = N^- := |\mathcal{X}^-|$, $M^+ = N^+ := |\mathcal{X}^+|$ and $\omega = 0$. Hence, in the worst case Theorem 4 yields a network with $\mathcal{O}(R(d+N^-)\delta^{-1}\log(2N^-N^+))$

1. A set of N points in $\mathbb{R}^d$ is said to be in general position if any subset of d vectors is linearly independent.

parameters and $\mathcal{O}\left(N^- + R\delta^{-1}\log(2N^-N^+)\right)$ neurons. If $N^- \simeq N^+$, the number of neurons scales (slightly worse than) linear in the number of points, which is worse than the best result on memorization capacity for networks using the threshold activation. In Proposition 13 we show that the linear scaling in terms of M^- in Theorem 4 is not a proof artifact. Hence, our method cannot recover optimal performance in the worst case. It is an interesting open question whether our method can be modified to achieve this.

Nevertheless, in practical situations one can hope that a much better mutual covering exists, due to intrinsic low-dimensional structure of the input data and/or a more benign label assignment than arbitrary labelling. In such cases Theorem 4 can guarantee a much smaller interpolating network. In particular, since our bounds are independent of the number of samples we can derive interpolation results for infinite sets (Corollary 5). In contrast, results on memorization capacity cannot have this feature. The VC-dimension[2] of feed-forward neural networks with threshold activation is $\mathcal{O}(W \log W)$ (Baum and Haussler, 1988), where W denotes the total number of parameters, i.e., the sum of the number of weights and biases over all layers. Hence, to memorize more samples than this upper bound, one would necessarily need to add more parameters to the network. Similar results hold for arbitrary piecewise linear activations such as the ReLU (Bartlett et al., 2019) or analytic definable activation functions (Sontag, 1997).

Upon acceptance of this paper, the work of Lee et al. (2024) was pointed out to us by one of the reviewers, which bears some similarities to ours. It introduces the concept of a polytope-basis cover of a dataset of two classes. They show that if this basis is known, then an interpolating three-layer fully-connected ReLU network can be associated to such a cover. They then provide upper (and some lower) bound on the network width sufficient for the existence of an interpolating net if the data is a convex polytope, structured as a simplicial complex, or can be covered by the difference of prismatic polytopes. While their theoretical guarantees are pure existence results, they also introduce a number of heuristic algorithms that yield small near-interpolating ReLU nets on, e.g., MNIST and CIFAR10. These methods are only guaranteed to terminate and, in contrast to our work, no guarantees are derived on the runtime, size of the network, and interpolation success.

Separation capacity. Related to interpolation is the question of *separation capacity* of a neural network: under what conditions can a neural network make two (not necessarily finite) classes linearly separable? Obviously, a network with separation capacity can be extended to an interpolating network by adding the separating hyperplane as an additional layer.

In An et al. (2015) it was shown that any two disjoint sets can be made linearly separable using a deterministic two-layer ReLU neural net. However, their proof is non-constructive and they provided no estimates on the size of the network. Inspired by this work, Dirksen et al. (2022a) showed that a wide enough two-layer random ReLU network can make any two δ-separated sets linearly separable if the weights and biases are chosen from appropriate distributions. Unlike the existence result of An et al. (2015), they provided bounds linking the number of required neurons to geometric properties of the classes and their mutual

2. The VC-dimension is the maximal N for which there exist points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N \in \mathbb{R}^d$ such that for every assignment of labels $y_1, \ldots, y_N \in \{\pm 1\}$ there exists a set of parameters θ such that the network interpolates the samples, i.e., $F_\theta(x_i) = y_i$ for all $i \in [N]$.

arrangement via a notion of mutual covering similar to Definition 3. This instance-specific viewpoint allows them to overcome the curse of dimensionality if the data carries a low-complexity structure. Following up on this, Ghosal et al. (2022) showed that even a wide enough one-layer ReLU net is enough to accomplish separation. They introduced a deterministic memorization algorithm which is then 'implemented' by a random neural network. As Dirksen et al. (2022a) they also used a mutual covering to capture the complexity of the data.

While the above results could be applied to interpolation, the required number of parameters would be larger than what we require in Theorem 4. Both Dirksen et al. (2022a) and Ghosal et al. (2022) yield networks scaling polynomially in terms of the mutual covering numbers, while our network scales only linearly.

The present paper is strongly influenced by Dirksen et al. (2022a)—we adopt an instance-specific viewpoint and the notion of mutual covering. However, instead of separation, we directly focus on interpolation. Together with our only partially randomized approach, this allows us to prove better bounds for this case.

Random hyperplane tesselations. As will become apparent below, our technical analysis is linked to tessellations created by random hyperplanes with Gaussian directions and uniformly distributed shifts, which were recently intensively studied in Dirksen and Mendelson (2021); Dirksen et al. (2022b). In particular, Dirksen et al. (2022b) derived a sharp bound on the number of hyperplanes needed to induce a uniform tessellation of a given set, meaning that the Euclidean distance between any two points in the set corresponds to the fraction of hyperplanes separating them up to a prespecified error. We will use some insights from these works, see in particular Lemma 18.

1.4 Setup and Notation

For any $1 \leq p \leq \infty$ we let $\|\cdot\|_p$ denote the ℓ_p norm. We use $\mathbb{B}_2^d(\boldsymbol{c}, r)$ to denote the Euclidean ball in $\mathbb{R}^d$ with center $\boldsymbol{c} \in \mathbb{R}^d$ and radius $r \geq 0$ and we denote the unit ball by $\mathbb{B}_2^d$. For $n \in \mathbb{N}$, we set $[n] \coloneqq \{1, \ldots, n\}$. For any set $\mathcal{A}$ we use $|\mathcal{A}|$ to denote its cardinality and let $\mathbb{1}_{\mathcal{A}}$ denote its indicator. We let sign denote the function

$$\operatorname{sign}(x) = \begin{cases} +1 & \text{if } x \geq 0, \\ -1 & \text{else.} \end{cases}$$

For a function $\sigma\colon \mathbb{R} \to \mathbb{R}$ and a vector $\boldsymbol{x} \in \mathbb{R}^d$ we denote the element-wise application by $\sigma(\boldsymbol{x}) = (\sigma(x_i))_{i=1}^d$. If an equality holds up to an absolute constant C, we write $A \gtrsim B$ instead of $A \geq C \cdot B$. We write $A \simeq B$ if $A \gtrsim B \gtrsim A$. We use $\mathcal{O}(\,\cdot\,)$ to omit constant terms and $\widetilde{\mathcal{O}}(\,\cdot\,)$ to additionally omit logarithmic terms. We define the distance between any point $\boldsymbol{x} \in \mathbb{R}^d$ and a set $\mathcal{X} \subset \mathbb{R}^d$ as $\mathrm{d}(\boldsymbol{x}, \mathcal{X}) \coloneqq \inf\{\|\boldsymbol{x} - \boldsymbol{y}\|_2 : \boldsymbol{y} \in \mathcal{X}\}$. We denote the hyperplane with direction $\boldsymbol{v} \in \mathbb{R}^d$ and shift $\tau \in \mathbb{R}$ by $H[\boldsymbol{v}, \tau] \coloneqq \{\boldsymbol{x} \in \mathbb{R}^d : \langle \boldsymbol{v}, \boldsymbol{x}\rangle + \tau = 0\}$. For $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$ we define $\mathbb{1}[\boldsymbol{x} = \boldsymbol{y}] \in \{0,1\}^n$ by

$$(\mathbb{1}[\boldsymbol{x} = \boldsymbol{y}])_i = \begin{cases} 1 & \text{if } x_i = y_i, \\ 0 & \text{else.} \end{cases}$$

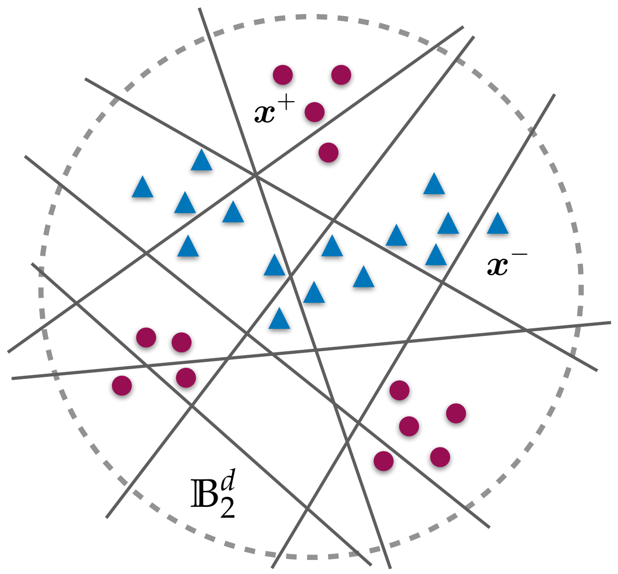


Figure 2: **Random hyperplanes in the input domain $\mathbb{R}^d$.** In Algorithm 1 we iteratively sample random hyperplanes $H[\boldsymbol{w}_i, b_i]$ until every pair of points with opposite labels is separated by at least one of them. This tessellates the space into multiple cells, where each cell is only populated with points of the same label. Each hyperplane can be associated with one of the neurons of the first layer Φ.

We denote by $\mathbf{0}, \mathbf{1} \in \mathbb{R}^d$ the vector with entries all equal to 0 and all equal to 1, respectively. We denote the standard multivariate normal distribution in d dimensions by $N(\mathbf{0}, \boldsymbol{I}_d)$ and the uniform distribution on $\mathcal{A} \subset \mathbb{R}^d$ by $\text{Unif}(\mathcal{A})$.

2. Interpolation Algorithm and Main Results

Consider any disjoint $\mathcal{X}^-, \mathcal{X}^+ \subset R\mathbb{B}_2^d$ with $N^- := |\mathcal{X}^-|$ and $N^+ := |\mathcal{X}^+|$. Let $\sigma\colon \mathbb{R} \to \mathbb{R}$ satisfy $\sigma(t) = 0$ for $t \leq 0$ and $\sigma(t) > 0$ for $t > 0$. Let us outline our method to construct an interpolating three-layer neural network:

1. To build the first layer $\Phi\colon \mathbb{R}^d \to \mathbb{R}^n$, we iteratively sample i.i.d. random hyperplanes $H[\boldsymbol{w}_i, b_i]$ until any $\boldsymbol{x}^- \in \mathcal{X}^-$ is separated from any $\boldsymbol{x}^+ \in \mathcal{X}^+$ by at least one of them (see Figure 2 and Definition 14). Each hyperplane includes a shift b_i so that it is able to separate points located on a ray emanating from the origin. In the worst case, one could have points with opposite labels close to the boundary of $R\mathbb{B}_2^d$, hence one needs the maximal shift to scale at least like R. We let $\boldsymbol{W}$ be the matrix containing the $\boldsymbol{w}_i$ as its rows and let $\boldsymbol{b}$ be the vector having the b_i as its coordinates. We define the first, random layer Φ of the network by $\Phi(\boldsymbol{x}) = \sigma(\boldsymbol{W}\boldsymbol{x} + \boldsymbol{b})$. Since all pairs of points with opposite labels are separated by at least one hyperplane, Φ has the following property: for any $(\boldsymbol{x}^-, \boldsymbol{x}^+) \in \mathcal{X}^- \times \mathcal{X}^+$ there exists at least one $i \in [n]$ with
$$\Phi_i(\boldsymbol{x}^-) = 0 \quad \text{and} \quad \Phi_i(\boldsymbol{x}^+) > 0. \tag{3}$$

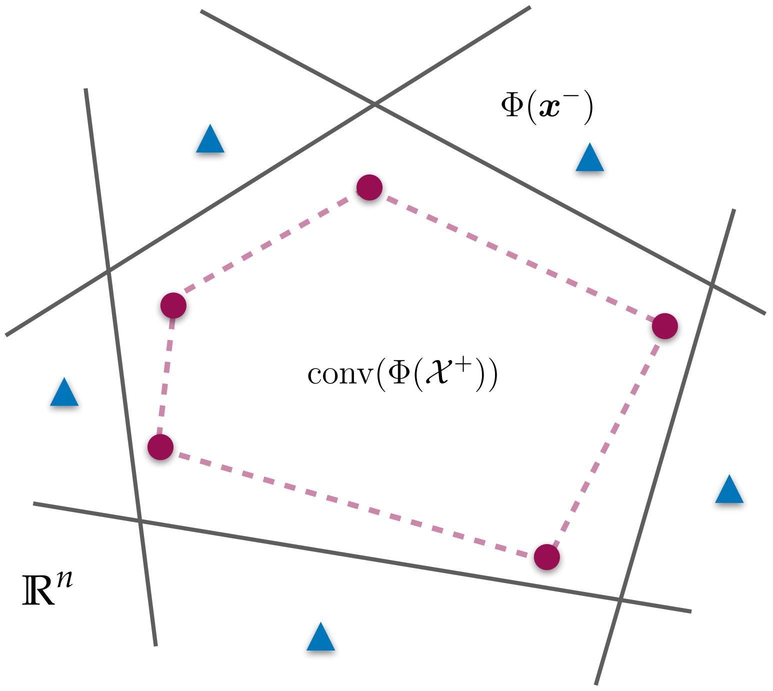


Figure 3: **The effect of the first layer Φ.** After transforming the data with the first layer Φ we can, for each $\boldsymbol{x}^- \in \mathcal{X}^-$, construct a hyperplane $H[-\boldsymbol{u}_{\boldsymbol{x}^-}, m_{\boldsymbol{x}^-}]$ that separates $\Phi(\mathcal{X}^+)$ from $\Phi(\boldsymbol{x}^-)$. Each hyperplane can be associated with one of the neurons in the second layer.

This enables us to distinguish between points of different labels.

2. We then exploit (3) in the following way. For $\boldsymbol{x}^- \in \mathcal{X}^-$ consider the mask $\boldsymbol{u}_{\boldsymbol{x}^-} = \mathbb{1}[\Phi(\boldsymbol{x}^-) = \boldsymbol{0}]$. By (3),

$$\langle \boldsymbol{u}_{\boldsymbol{x}^-}, \Phi(\boldsymbol{x}^-)\rangle = 0 \quad \text{and} \quad \langle \boldsymbol{u}_{\boldsymbol{x}^-}, \Phi(\boldsymbol{x}^+)\rangle > 0 \quad \text{for all } \boldsymbol{x}^+ \in \mathcal{X}^+.$$

Geometrically, this means that the hyperplane $H[-\boldsymbol{u}_{\boldsymbol{x}^-}, m_{\boldsymbol{x}^-}]$, where

$$m_{\boldsymbol{x}^-} = \min_{\boldsymbol{x}^+ \in \mathcal{X}^+} \langle \boldsymbol{u}_{\boldsymbol{x}^-}, \Phi(\boldsymbol{x}^+)\rangle,$$

separates $\Phi(\mathcal{X}^+)$ from $\Phi(\boldsymbol{x}^-)$ (see Figure 3). Let $\boldsymbol{U} \in \mathbb{R}^{N^- \times n}$ be the matrix with rows $\boldsymbol{u}_{\boldsymbol{x}^-}$ and let $\boldsymbol{m} \in \mathbb{R}^{N^-}$ be the vector with coordinates $m_{\boldsymbol{x}^-}$. We then define the second layer $\hat{\Phi}\colon \mathbb{R}^n \to \mathbb{R}^{\hat{n}}$ of the network by $\hat{\Phi}(\boldsymbol{z}) = \sigma(-\boldsymbol{U}\boldsymbol{z} + \boldsymbol{m})$. This layer satisfies, for every $\boldsymbol{x}^- \in \mathcal{X}^-$,

$$[\hat{\Phi}(\Phi(\boldsymbol{x}^-))]_{\boldsymbol{x}^-} > 0 \quad \text{and} \quad [\hat{\Phi}(\Phi(\boldsymbol{x}^+))]_{\boldsymbol{x}^-} = 0 \quad \text{for all } \boldsymbol{x}^+ \in \mathcal{X}^+. \tag{4}$$

Thus, in the second hidden layer, there is a dedicated neuron to detect each point of $\mathcal{X}^-$, but none of them activates on $\mathcal{X}^+$.

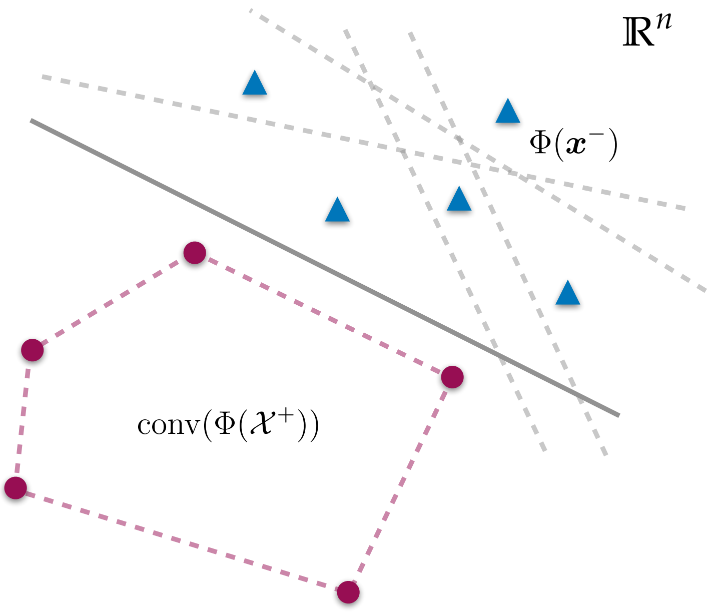


Figure 4: **Motivation for forward selection.** While each $\Phi(\boldsymbol{x}^-)$ is separated by a corresponding 'dedicated' hyperplane from $\Phi(\mathcal{X}^+)$ (depicted in dashed grey), we can identify a single hyperplane $H[-\boldsymbol{u}_{\boldsymbol{x}_*^-}, m_{\boldsymbol{x}_*^-}]$ (depicted in grey) that separates several $\Phi(\boldsymbol{x}^-)$ from $\Phi(\mathcal{X}^+)$ simultaneously. The other hyperplanes are redundant and the corresponding neurons do not need to be included in the second layer $\hat{\Phi}$.

3. In the output layer, we simply sum the output from the second layer $\hat{\Phi}$. By (4), for all $\boldsymbol{x}^- \in \mathcal{X}^-$ and $\boldsymbol{x}^+ \in \mathcal{X}^+$,

$$\langle \mathbf{1}, \hat{\Phi}(\Phi(\boldsymbol{x}^-))\rangle > 0 \quad \text{and} \quad \langle \mathbf{1}, \hat{\Phi}(\Phi(\boldsymbol{x}^+))\rangle = 0$$

   and hence $\operatorname{sign}(-\cdot)$ outputs the correct label.

The second step of this method is rather naive: for *every* $\boldsymbol{x}^- \in \mathcal{X}^-$, we construct a dedicated neuron

$$\hat{\varphi}_{\boldsymbol{x}^-}(\boldsymbol{z}) = \sigma(-\langle \boldsymbol{u}_{\boldsymbol{x}^-}, \boldsymbol{z}\rangle + m_{\boldsymbol{x}^-}) \tag{5}$$

that distinguishes $\Phi(\boldsymbol{x}^-)$ and $\Phi(\mathcal{X}^+)$, i.e., $\hat{\varphi}_{\boldsymbol{x}^-}(\Phi(\boldsymbol{x}^-)) > 0$ and $\hat{\varphi}_{\boldsymbol{x}^-}(\Phi(\boldsymbol{x}^+)) = 0$ for all $\boldsymbol{x}^+ \in \mathcal{X}^+$. This potentially leads to redundancy, since to get an interpolating net at the third step, it suffices if for each $\boldsymbol{x}^-$ there is *some* $\boldsymbol{x}_*^-$ such that $\hat{\varphi}_{\boldsymbol{x}_*^-}$ distinguishes $\Phi(\boldsymbol{x}^-)$ and $\Phi(\mathcal{X}^+)$. We can especially hope for this to be true if $\boldsymbol{x}^-$ is 'close enough to' $\boldsymbol{x}_*^-$ in a suitable sense. This is illustrated in Figure 4. Thus we can improve the second step by forward selection: we iteratively select elements $\boldsymbol{x}_*^-$ from $\mathcal{X}^-$ and construct the associated neuron $\hat{\varphi}_{\boldsymbol{x}_*^-}$ until there is a distinguishing neuron for each element in $\mathcal{X}^-$.

These considerations lead to our interpolation algorithm formalized in Algorithm 1.

Algorithm 1 Interpolation

Input: Disjoint and finite $\mathcal{X}^-, \mathcal{X}^+ \subset \mathbb{R}^d$, activation $\sigma\colon \mathbb{R} \to \mathbb{R}$ satisfying $\sigma(t) = 0$ for $t \leq 0$ and $\sigma(t) > 0$ for $t > 0$, (minimal) width of the first layer $n_{\min} \geq 0$.

Output: A three-layer fully-connected neural network $F\colon \mathbb{R}^d \to \{\pm 1\}$ that interpolates $\mathcal{X}^-$ and $\mathcal{X}^+$.

```
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
 1: Calculate $R \leftarrow \max_{\boldsymbol{x} \in \mathcal{X}^- \cup \mathcal{X}^+} \|\boldsymbol{x}\|_2$ and choose $\lambda \gtrsim R$.
 2: Initialize $\mathcal{S} \leftarrow \emptyset$ and $n \leftarrow 0$.
 3: while $\mathcal{S} \neq \mathcal{X}^- \times \mathcal{X}^+$ or $n < n_{\min}$ do
 4:     Update $n \leftarrow n + 1$.
 5:     Sample
            $$\boldsymbol{w}_n \sim N(\boldsymbol{0}, \boldsymbol{I}_d), \quad b_n \sim \mathrm{Unif}([-\lambda, \lambda]).$$
 6:     Update $\mathcal{S}$ according to
            $$\mathcal{S} \leftarrow \mathcal{S} \cup \{(\boldsymbol{x}^-, \boldsymbol{x}^+) \in \mathcal{X}^- \times \mathcal{X}^+ : \langle \boldsymbol{w}_n, \boldsymbol{x}^- \rangle \leq -b_n < \langle \boldsymbol{w}_n, \boldsymbol{x}^+ \rangle\}.$$
 7: end while
 8: Define $\Phi(\boldsymbol{x}) = \sigma(\boldsymbol{W}\boldsymbol{x} + \boldsymbol{b})$ with $\boldsymbol{W} \in \mathbb{R}^{n \times d}$ and $\boldsymbol{b} \in \mathbb{R}^n$ where
            $$\boldsymbol{W} \leftarrow \begin{bmatrix} \boldsymbol{w}_1^\top \\ \vdots \\ \boldsymbol{w}_n^\top \end{bmatrix} \quad \text{and} \quad \boldsymbol{b} \leftarrow \begin{bmatrix} b_1 \\ \vdots \\ b_n \end{bmatrix}.$$
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
 9: Initialize $\mathcal{C} \leftarrow \mathcal{X}^-$ and $\hat{n} \leftarrow 0$.
10: while $\mathcal{C} \neq \emptyset$ do
11:     Update $\hat{n} \leftarrow \hat{n} + 1$.
12:     Select $\boldsymbol{x}_{\hat{n}}^- \in \mathcal{C}$ uniformly at random from $\mathcal{C}$ and calculate
            $$\boldsymbol{u}_{\hat{n}} \leftarrow \mathbb{1}[\Phi(\boldsymbol{x}_{\hat{n}}^-) = \boldsymbol{0}], \quad m_{\hat{n}} \leftarrow \min_{\boldsymbol{x}^+ \in \mathcal{X}^+} \langle \boldsymbol{u}_{\hat{n}}, \Phi(\boldsymbol{x}^+) \rangle.$$
13:     Update $\mathcal{C}$ according to
            $$\mathcal{C} \leftarrow \mathcal{C} \setminus \{\boldsymbol{x}^- \in \mathcal{C} : \langle \boldsymbol{u}_{\boldsymbol{x}_{\hat{n}}^-}, \Phi(\boldsymbol{x}^-) \rangle < m_{\boldsymbol{x}_{\hat{n}}^-}\}.$$
14: end while
15: Define $\hat{\Phi}(\boldsymbol{z}) = \sigma(-\boldsymbol{U}\boldsymbol{z} + \boldsymbol{m})$ with $\boldsymbol{U} \in \mathbb{R}^{\hat{n} \times n}$ and $\boldsymbol{m} \in \mathbb{R}^{\hat{n}}$ where
            $$\boldsymbol{U} \leftarrow \begin{bmatrix} \boldsymbol{u}_1^\top \\ \vdots \\ \boldsymbol{u}_{\hat{n}}^\top \end{bmatrix} \quad \text{and} \quad \boldsymbol{m} \leftarrow \begin{bmatrix} m_1 \\ \vdots \\ m_{\hat{n}} \end{bmatrix}.$$
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
16: Return $F(\boldsymbol{x}) = \mathrm{sign}(-\langle \boldsymbol{1}, \hat{\Phi}(\Phi(\boldsymbol{x})) \rangle)$.
```

Remark 6 *First, let us briefly comment on the parameter $n_{\min}$ in the first loop of Algorithm 1, which is the minimal width of the first layer Φ. In (the proof of) Proposition 7 we will see that the first loop (and hence, the algorithm) terminates with probability* 1*, regardless of the choice of n_{min}. In Theorem 11, we will derive a lower bound on $n_{\min}$ that ensures that the algorithm terminates with high probability after $n_{\min}$ iterations and derive an upper bound on the total size of the output net F. The first condition in line 3 of the algorithm will in this case be redundant. We include this condition to ensure that the algorithm is always guaranteed to terminate, for any choice of n_{min}.*

Second, we comment on the parameter λ, which is the maximal shift of the hyperplanes in the first layer. The condition $\lambda \gtrsim R$ in the first line of Algorithm 1 is used to guarantee that every pair of samples with different labels is separated by at least one of the hyperplanes (even if they are on a line through the origin, Proposition 15), and that they induce a uniform tesselation, allowing us to relate the fraction of hyperplanes between points to their Euclidean distance (Lemmas 17 and 18). As this condition involves an unknown constant, for a practical application λ can be treated like a hyperparameter. In Section 4 we will see that $\lambda \geq R$ typically is sufficient and, depending on the data set, smaller values might also work.

Let us now state our main results.

Proposition 7 (Termination and correctness) *Let $\mathcal{X}^-, \mathcal{X}^+ \subset \mathbb{R}^d$ be disjoint and finite. Then Algorithm 1 terminates with probability* 1 *and its output F interpolates $\mathcal{X}^-$ and $\mathcal{X}^+$.*

From the discussion at the start of this section, it is clear that Algorithm 1 produces an interpolating network *if* the first loop of the algorithm terminates. We will prove termination in Section 3.1.

Additionally, the following gives an estimate of the run time of Algorithm 1.

Proposition 8 (Run time) *Let $\mathcal{X}^-, \mathcal{X}^+ \subset \mathbb{R}^d$ be finite and δ-separated. Let $N^- := |\mathcal{X}^-|$ and $N^+ := |\mathcal{X}^+|$ and denote $N := N^- + N^+$. Assume that $N^- \simeq N^+$, the input dimension d is constant and the activation function σ is computable in constant time. Then Algorithm 1 has a run time of at most*

$$\mathcal{O}(\delta^{-1}\lambda \log(N/\eta) N^2),$$

with probability at least $1 - \eta$.

Remark 9 *The run time of Algorithm 1 has a bottleneck of $\mathcal{O}(N^2)$ in terms of the number of samples, which may be serious for large data sets. This bottleneck already occurs in the first loop. In Section 4 we will consider a variation of the algorithm in which the number of hyperplanes drawn in the first layer is a hyperparameter. As we will see in Theorem 11, this algorithm is guaranteed to succeed with high probability if the number of draws is chosen large enough. In this case, the run time of the algorithm is dictated by the construction of the second layer, which takes time $\mathcal{O}(M^- N^+)$.*

To complement Proposition 7 we derive a high probability bound on the size of the network produced by Algorithm 1. This bound will (at least in our proof) depend on the choice of the activation function σ. We focus on the setting with threshold activations, i.e., we consider

$$\sigma(t) = \mathrm{Thres}(t) = \begin{cases} 1 & \text{if } t > 0, \\ 0 & \text{else.} \end{cases}$$

Let us first observe that in the limit, the shape of the activation region of every neuron in the second layer is a Euclidean ball of a 'maximal radius', i.e., that touches the closest point in the set $\mathcal{X}^+$. This gives geometric intuition on why the size of the second layer is naturally connected with the mutual covering numbers.

Proposition 10 (Limit shape of activation regions—threshold activations) *Take any $\boldsymbol{x}_*^- \in \mathcal{X}^-$ and let $\mathcal{A}_{\boldsymbol{x}_*^-}$ be the activation region of $\hat{\varphi}_{\boldsymbol{x}_*^-}$. Then, for any $\boldsymbol{x} \in \mathbb{R}^d \setminus \partial\mathcal{B}_{\boldsymbol{x}_*^-}$,*

$$\lim_{\lambda\to\infty} \lim_{n\to\infty} \mathbb{1}_{\mathcal{A}_{\boldsymbol{x}_*^-}}(\boldsymbol{x}) = \mathbb{1}_{\mathcal{B}_{\boldsymbol{x}_*^-}}(\boldsymbol{x})$$

almost surely, where $\mathcal{B}_{\boldsymbol{x}_*^-} = \mathbb{B}_2^d(\boldsymbol{x}_*^-; \mathrm{d}(\boldsymbol{x}_*^-, \mathcal{X}^+))$.

Let us give an intuitive sketch for the proof of Proposition 10. Roughly speaking, the neuron $\hat{\varphi}_{\boldsymbol{x}_*^-}$ activates when the fraction of hyperplanes separating the reference point $\boldsymbol{x}_*^-$ and the input $\boldsymbol{x}$ is smaller than a threshold value, which is the minimal fraction of hyperplanes separating $\boldsymbol{x}_*^-$ and any $\boldsymbol{x}^+ \in \mathcal{X}^+$. If $n \to \infty$, then the fraction of hyperplanes separating $\boldsymbol{x}_*^-$ and any $\boldsymbol{z}$ becomes proportional to the probability that a hyperplane separates the two. Finally, as $\lambda \to \infty$, this probability becomes proportional to $\mathrm{d}(\boldsymbol{x}_*^-, \boldsymbol{z})/\lambda$. Hence, in the double limit, the neuron activates when $\mathrm{d}(\boldsymbol{x}_*^-, \boldsymbol{x})$ is smaller than $\mathrm{d}(\boldsymbol{x}_*^-, \mathcal{X}^+)$.

Let us now state the main result of our work.

Theorem 11 (Size of interpolating net—threshold activations) *Let $\mathcal{X}^-, \mathcal{X}^+ \subset R\mathbb{B}_2^d$ be finite and disjoint. Let σ be the threshold activation and $\lambda \gtrsim R$. Suppose that there is a mutual covering of $\mathcal{X}^-$ and $\mathcal{X}^+$ such that the centers $\mathcal{C}^-$ and $\mathcal{C}^+$ are δ-separated and the radii satisfy*

$$r_\ell^- \lesssim \frac{\mathrm{d}(\boldsymbol{c}_\ell^-, \mathcal{C}^+)}{\log^{1/2}(e\lambda/\mathrm{d}(\boldsymbol{c}_\ell^-, \mathcal{C}^+))} \quad \text{and} \quad r_j^+ \lesssim \frac{\mathrm{d}(\boldsymbol{c}_j^+, \mathcal{C}^-)}{\log^{1/2}(e\lambda/\mathrm{d}(\boldsymbol{c}_j^+, \mathcal{C}^-))}$$

for all $\ell \in [M^-]$ and $j \in [M^+]$. Set $\omega := \max\{\omega^-, \omega^+\}$ where

$$\omega^- := \max_{\ell\in[M^-]} \frac{w^2(\mathcal{X}_\ell^- - \boldsymbol{c}_\ell^-)}{\mathrm{d}^3(\boldsymbol{c}_\ell^-, \mathcal{C}^+)} \quad \text{and} \quad \omega^+ := \max_{j\in[M^+]} \frac{w^2(\mathcal{X}_j^+ - \boldsymbol{c}_j^+)}{\mathrm{d}^3(\boldsymbol{c}_j^+, \mathcal{C}^-)}.$$

Suppose that

$$n_{\min} \gtrsim \lambda\delta^{-1}\log(2M^-M^+/\eta) + \lambda\omega. \tag{6}$$

Then, with probability at least $1-\eta$, the neural network computed by Algorithm 1 has layer widths $n = n_{\min}$ and $\hat{n} \leq M^-$.

Remark 12 *We give a few examples of estimates of the Gaussian mean width (see, e.g., Vershynin, 2018, for further details) to highlight some special cases of the condition* (6).

1. *For a finite set $\mathcal{A} \subset \mathbb{B}_2^d$ we have $w(\mathcal{A}) \lesssim \sqrt{\log(|\mathcal{A}|)}$. As Algorithm 1 requires a finite number N of input samples, $\omega \lesssim \delta^{-1}\log(N)$.*
2. *If $\mathcal{A} \subset \mathbb{B}_2^d$ lies in a k-dimensional subspace, then $w(\mathcal{A}) \lesssim \sqrt{k}$. Hence, for samples in a k-dimensional subspace, $\omega \lesssim \delta^{-1}k$.*
3. *The set $\Sigma_s^d := \{\boldsymbol{x} \in \mathbb{B}_2^d : \|\boldsymbol{x}\|_0 \leq s\}$ of s-sparse vectors in the unit ball, where $\|\boldsymbol{x}\|_0$ counts the number of non-zero coordinates in $\boldsymbol{x}$, satisfies $w(\Sigma_s^d) \lesssim \sqrt{s\log(ed/s)}$. Hence, if the input samples are s-sparse, $\omega \lesssim \delta^{-1}s\log(ed/s)$.*

Notice that the latter two estimates are independent of the number of samples.

The idea of the proof of Theorem 11 is to show that if Φ is wide enough, then the neuron $\hat{\varphi}_{\boldsymbol{x}_*^-}$ associated with $\boldsymbol{x}_*^-$ (defined in (5)) not only separates $\Phi(\boldsymbol{x}_*^-)$ and $\Phi(\mathcal{X}^+)$, but in fact acts as a *robust separator*: it will also separate $\Phi(\boldsymbol{x}^-)$ and $\Phi(\mathcal{X}^+)$ for all points $\boldsymbol{x}^-$ 'close enough to' $\boldsymbol{x}_*^-$. The key formal observation is stated below in Lemma 19. Intuitively, the notion of 'close enough' should be relative to the distance of $\boldsymbol{x}_*^-$ to the decision boundary. As a result, the size of the interpolating neural net is related to the 'complexity' of a mutual covering of $\mathcal{X}^-$ and $\mathcal{X}^+$ in which only the parts of $\mathcal{X}^-$ and $\mathcal{X}^+$ that lie close to the decision boundary need to be covered using components with small diameter—other parts can be crudely covered using large components (see Figure 1).

Finally, we prove that the statement of Theorem 11 cannot be improved in a certain sense. Proposition 13 below shows that the upper bound on the size of the second layer $\hat{\Phi}$, as stated in Theorem 11, cannot be improved in general, assuming that, in addition, σ is non-decreasing. Note that this assumption is satisfied by many popular activations, including the ReLU. In the proof, we construct a one-dimensional data set of points with alternating labels, which one could however embed (e.g., by appending zeros) into $\mathbb{R}^d$ for an arbitrary dimension $d \geq 1$. Note that the result holds independently of the random sampling of the first layer, so one cannot even find a benign choice of hyperplanes to improve the situation described below.

Proposition 13 *Assume that σ is non-decreasing, $\sigma(t) = 0$ for $t \leq 0$ and $\sigma(t) > 0$ for $t > 0$. Let $M^- \geq 2$ and $M^+ := M^- - 1$. Then, for all $N^- \geq M^-$ and $N^+ \geq M^+$, there exists $\mathcal{X}^-, \mathcal{X}^+ \subset [0, 1]$ with $N^- = |\mathcal{X}^-|$ and $N^+ = |\mathcal{X}^+|$, and a mutual covering $\mathcal{C}^- = \{c_1^-, \dots, c_{M^-}^-\} \subset \mathcal{X}^-$ and $\mathcal{C}^+ = \{c_1^+, \dots, c_{M^+}^+\} \subset \mathcal{X}^+$ such that the output F of Algorithm 1 has at least M^- neurons in its second layer.*

3. Proofs

In this section, we present the proofs that have previously been omitted.

3.1 Proof of Proposition 7

We use the following terminology.

Definition 14 *Let $\boldsymbol{v} \in \mathbb{R}^d \setminus \{\boldsymbol{0}\}$, $\tau \in \mathbb{R}$ and $t \geq 0$. A hyperplane $H[\boldsymbol{v}, \tau]$* t-separates $\mathcal{X}^-$ from $\mathcal{X}^+$ *if*

$$\langle \boldsymbol{v}, \boldsymbol{x}^- \rangle + \tau \leq -t \qquad \textit{for all } \boldsymbol{x}^- \in \mathcal{X}^-,$$
$$\langle \boldsymbol{v}, \boldsymbol{x}^+ \rangle + \tau > +t \qquad \textit{for all } \boldsymbol{x}^+ \in \mathcal{X}^+.$$

If $t = 0$, *we simply say that* $H[\boldsymbol{v}, \tau]$ separates $\mathcal{X}^-$ from $\mathcal{X}^+$.

To prove Proposition 7 it suffices to prove the following statement. It shows that the probability that the first loop of Algorithm 1 stops, and hence the algorithm terminates, increases exponentially in terms of the number of hyperplanes n. Allowing n to grow unbounded then directly yields Proposition 7.

Proposition 15 *Let* $\mathcal{X}^-, \mathcal{X}^+ \subset R\mathbb{B}_2^d$ *be finite and* δ*-separated with* $N^- := |\mathcal{X}^-|$ *and* $N^+ := |\mathcal{X}^+|$. *Let* $\lambda \gtrsim R$. *Assume that the loop in Algorithm 1 ran for at least* n *iterations, where*

$$n \gtrsim \delta^{-1}\lambda \cdot \log(N^- N^+/\eta). \tag{7}$$

Then, the exit condition of the loop is satisfied with probability at least $1 - \eta$.

In the proof, we will use the following lower bound on the probability that a random hyperplane from Algorithm 1 separates a fixed pair of points.

Lemma 16 *(Dirksen et al., 2022a, Theorem 18) There is an absolute constant* $c > 0$ *such that the following holds. Let* $\boldsymbol{x}^-, \boldsymbol{x}^+ \in R\mathbb{B}_2^d$. *Let* $\boldsymbol{g} \in \mathbb{R}^d$ *denote a standard Gaussian random vector and let* $\tau \in [-\lambda, \lambda]$ *be uniformly distributed. If* $\lambda \gtrsim R$, *then with probability at least* $c\|\boldsymbol{x}^+ - \boldsymbol{x}^-\|_2/\lambda$, *the hyperplane* $H[\boldsymbol{g}, \tau]$ $\|\boldsymbol{x}^+ - \boldsymbol{x}^-\|_2$*-separates* $\boldsymbol{x}^-$ *from* $\boldsymbol{x}^+$.

Proof [Proposition 15] Fix $\boldsymbol{x}^- \in \mathcal{X}^-$ and $\boldsymbol{x}^+ \in \mathcal{X}^+$. We consider i.i.d. copies $H_1, \dots, H_n$ of a hyperplane $H = H[\boldsymbol{w}, b]$, where $\boldsymbol{w} \sim N(\boldsymbol{0}, \boldsymbol{I}_d)$ and $b \sim \mathrm{Unif}([-\lambda, \lambda])$ are independent. By Lemma 16, the probability that $\boldsymbol{x}^-$ and $\boldsymbol{x}^+$ is not separated by any of these hyperplanes is at most $(1 - c\delta/\lambda)^n$. By taking a union bound over all N^-N^+ pairs of points, we see that the probability that at least one pair has no separating hyperplane is at most

$$N^-N^+\left(1 - c\frac{\delta}{\lambda}\right)^n \le N^-N^+ e^{-c\frac{\delta}{\lambda}n} \le \eta,$$

where we used that $1 + x \le e^x$ for $x \in \mathbb{R}$ and the last inequality follows from (7). ■

3.2 Proof of Proposition 8

The calculation of the radius takes time $\mathcal{O}(N^- + N^+)$. The loops run for n and $\hat{n}$ iterations where each iteration takes time $\mathcal{O}(N^-N^+)$ and $\mathcal{O}(n(N^- + N^+))$, respectively. Transforming all samples once with the first layer (which is needed to compute the second loop) takes time $\mathcal{O}(n(N^- + N^+))$. This totals $\mathcal{O}(n(N^-N^+ + \hat{n}N^- + \hat{n}N^+)) = \mathcal{O}(nN^2)$, where we used that $\hat{n} \le N$. Applying Proposition 15 completes the proof.

3.3 Proof of Proposition 10

Recall that the neuron $\hat{\varphi}_{\boldsymbol{x}_*^-}$ activates on $\boldsymbol{x} \in \mathbb{R}^d$ if and only if

$$\langle \boldsymbol{u}_{\boldsymbol{x}_*^-}, \Phi(\boldsymbol{x}) \rangle < m_{\boldsymbol{x}_*^-} = \min_{\boldsymbol{x}^+ \in \mathcal{X}^+} \langle \boldsymbol{u}_{\boldsymbol{x}_*^-}, \Phi(\boldsymbol{x}^+) \rangle.$$

We make two observations. First, for any $\boldsymbol{x} \in \mathbb{R}^d$,

$$\langle \boldsymbol{u}_{\boldsymbol{x}_*^-}, \Phi(\boldsymbol{x}) \rangle = \sum_{i=1}^n \mathbb{1}_{\{\Phi(\boldsymbol{x}_*^-)_i = 0\}} \Phi(\boldsymbol{x})_i = \sum_{i=1}^n \mathbb{1}_{\{\langle \boldsymbol{w}_i, \boldsymbol{x}_*^- \rangle + b_i \le 0 < \langle \boldsymbol{w}_i, \boldsymbol{x} \rangle + b_i\}},$$

and hence, by the law of large numbers and by symmetry,

$$\lim_{n\to\infty} \frac{1}{n} \langle \boldsymbol{u}_{\boldsymbol{x}_*^-}, \Phi(\boldsymbol{x}) \rangle = \frac{1}{2} \mathbb{P}(\operatorname{sign}(\langle \boldsymbol{w}, \boldsymbol{x}_*^- \rangle + b) \neq \operatorname{sign}(\langle \boldsymbol{w}, \boldsymbol{x} \rangle + b)) \tag{8}$$

almost surely, where $\boldsymbol{w} \sim N(\boldsymbol{0}, \boldsymbol{I}_d)$ and $b \sim \operatorname{Unif}([-\lambda, \lambda])$ are independent. Second, by (Dirksen et al., 2022b, Lemma A.1), for any $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^d$,

$$\begin{aligned}
&2\lambda \mathbb{P}_b(\operatorname{sign}(\langle \boldsymbol{w}, \boldsymbol{x} \rangle + b) \neq \operatorname{sign}(\langle \boldsymbol{w}, \boldsymbol{y} \rangle + b)) \\
&\quad = |\langle \boldsymbol{w}, \boldsymbol{x} - \boldsymbol{y} \rangle| 1_{\{|\langle \boldsymbol{w}, \boldsymbol{x} \rangle| \le \lambda, |\langle \boldsymbol{w}, \boldsymbol{y} \rangle| \le \lambda\}} \\
&\qquad + 2\lambda (1_{\{\langle \boldsymbol{w}, \boldsymbol{x} \rangle > \lambda, \langle \boldsymbol{w}, \boldsymbol{y} \rangle < -\lambda\}} + 1_{\{\langle \boldsymbol{w}, \boldsymbol{x} \rangle < -\lambda, \langle \boldsymbol{w}, \boldsymbol{y} \rangle > \lambda\}}) \\
&\qquad + (\lambda - \langle \boldsymbol{w}, \boldsymbol{x} \rangle) 1_{\{\langle \boldsymbol{w}, \boldsymbol{y} \rangle > \lambda, |\langle \boldsymbol{w}, \boldsymbol{x} \rangle| \le \lambda\}} + (\lambda - \langle \boldsymbol{w}, \boldsymbol{y} \rangle) 1_{\{\langle \boldsymbol{w}, \boldsymbol{x} \rangle > \lambda, |\langle \boldsymbol{w}, \boldsymbol{y} \rangle| \le \lambda\}} \\
&\qquad + (\lambda + \langle \boldsymbol{w}, \boldsymbol{x} \rangle) 1_{\{\langle \boldsymbol{w}, \boldsymbol{y} \rangle < -\lambda, |\langle \boldsymbol{w}, \boldsymbol{x} \rangle| \le \lambda\}} + (\lambda + \langle \boldsymbol{w}, \boldsymbol{y} \rangle) 1_{\{\langle \boldsymbol{w}, \boldsymbol{x} \rangle < -\lambda, |\langle \boldsymbol{w}, \boldsymbol{y} \rangle| \le \lambda\}},
\end{aligned}$$

where $\mathbb{P}_b$ is the probability with respect to b. As $\mathbb{P}(|\langle \boldsymbol{w}, \boldsymbol{z} \rangle| > \lambda) \le 2e^{-c\lambda^2/\|\boldsymbol{z}\|_2^2}$ for any $\boldsymbol{z} \in \mathbb{R}^d$, we find by taking expectations with respect to $\boldsymbol{w}$, taking the limit for $\lambda \to \infty$, and using monotone convergence that

$$\lim_{\lambda\to\infty} 2\lambda \mathbb{P}(\operatorname{sign}(\langle \boldsymbol{w}, \boldsymbol{x} \rangle + b) \neq \operatorname{sign}(\langle \boldsymbol{w}, \boldsymbol{y} \rangle + b)) = \mathbb{E}|\langle \boldsymbol{w}, \boldsymbol{x} - \boldsymbol{y} \rangle| = \sqrt{2/\pi} \|\boldsymbol{x} - \boldsymbol{y}\|_2. \tag{9}$$

We proceed with the proof by distinguishing two cases. Let $\boldsymbol{x} \in \mathbb{R}^d$, assume $\|\boldsymbol{x}_*^- - \boldsymbol{x}\|_2 < \min_{\boldsymbol{x}^+ \in \mathcal{X}^+} \|\boldsymbol{x}_*^- - \boldsymbol{x}^+\|_2$ and define

$$\varepsilon := \frac{\min_{\boldsymbol{x}^+ \in \mathcal{X}^+} \|\boldsymbol{x}_*^- - \boldsymbol{x}^+\|_2 - \|\boldsymbol{x}_*^- - \boldsymbol{x}\|_2}{2} > 0.$$

By (9), there exists $\Lambda > 0$ such that for $\lambda > \Lambda$,

$$\begin{aligned}
&\sqrt{2\pi}\lambda \mathbb{P}(\operatorname{sign}(\langle \boldsymbol{w}, \boldsymbol{x}_*^- \rangle + b) \neq \operatorname{sign}(\langle \boldsymbol{w}, \boldsymbol{x} \rangle + b)) \\
&\qquad < \left\|\boldsymbol{x}_*^- - \boldsymbol{x}\right\|_2 + \varepsilon \\
&\qquad = \min_{\boldsymbol{x}^+ \in \mathcal{X}^+} \left\|\boldsymbol{x}_*^- - \boldsymbol{x}^+\right\|_2 - \varepsilon \\
&\qquad < \min_{\boldsymbol{x}^+ \in \mathcal{X}^+} \sqrt{2\pi}\lambda \mathbb{P}(\operatorname{sign}(\langle \boldsymbol{w}, \boldsymbol{x}_*^- \rangle + b) \neq \operatorname{sign}(\langle \boldsymbol{w}, \boldsymbol{x}^+ \rangle + b)),
\end{aligned}$$

and hence

$$\begin{aligned}
&\mathbb{P}(\operatorname{sign}(\langle \boldsymbol{w}, \boldsymbol{x}_*^- \rangle + b) \neq \operatorname{sign}(\langle \boldsymbol{w}, \boldsymbol{x} \rangle + b)) \\
&\qquad < \min_{\boldsymbol{x}^+ \in \mathcal{X}^+} \mathbb{P}(\operatorname{sign}(\langle \boldsymbol{w}, \boldsymbol{x}_*^- \rangle + b) \neq \operatorname{sign}(\langle \boldsymbol{w}, \boldsymbol{x}^+ \rangle + b)).
\end{aligned}$$

Further, define

$$\begin{aligned}
\delta := \frac{1}{2}\big(&\min_{\boldsymbol{x}^+ \in \mathcal{X}^+} \mathbb{P}(\operatorname{sign}(\langle \boldsymbol{w}, \boldsymbol{x}_*^- \rangle + b) \neq \operatorname{sign}(\langle \boldsymbol{w}, \boldsymbol{x}^+ \rangle + b)) \\
&\qquad - \mathbb{P}(\operatorname{sign}(\langle \boldsymbol{w}, \boldsymbol{x}_*^- \rangle + b) \neq \operatorname{sign}(\langle \boldsymbol{w}, \boldsymbol{x} \rangle + b))\big) > 0.
\end{aligned}$$

By (8), almost surely, there exists $N \in \mathbb{N}$ such that for $n > N$,

$$\begin{aligned}\frac{2}{n}\langle \boldsymbol{u}_{\boldsymbol{x}_*^-}, \Phi(\boldsymbol{x})\rangle &< \mathbb{P}(\operatorname{sign}(\langle \boldsymbol{w}, \boldsymbol{x}_*^-\rangle + b) \neq \operatorname{sign}(\langle \boldsymbol{w}, \boldsymbol{x}\rangle + b)) + \delta \\ &= \min_{\boldsymbol{x}^+ \in \mathcal{X}^+} \mathbb{P}(\operatorname{sign}(\langle \boldsymbol{w}, \boldsymbol{x}_*^-\rangle + b) \neq \operatorname{sign}(\langle \boldsymbol{w}, \boldsymbol{x}^+\rangle + b)) - \delta \\ &< \min_{\boldsymbol{x}^+ \in \mathcal{X}^+} \frac{2}{n}\langle \boldsymbol{u}_{\boldsymbol{x}_*^-}, \Phi(\boldsymbol{x}^+)\rangle,\end{aligned}$$

and hence

$$\langle \boldsymbol{u}_{\boldsymbol{x}_*^-}, \Phi(\boldsymbol{x})\rangle < \min_{\boldsymbol{x}^+ \in \mathcal{X}^+} \langle \boldsymbol{u}_{\boldsymbol{x}_*^-}, \Phi(\boldsymbol{x}^+)\rangle.$$

This shows that

$$\lim_{\lambda\to\infty}\lim_{n\to\infty} \mathbb{1}_{\mathcal{A}_{\boldsymbol{x}_*^-}}(\boldsymbol{x}) = \mathbb{1}_{\mathcal{B}_{\boldsymbol{x}_*^-}}(\boldsymbol{x})$$

almost surely if $\|\boldsymbol{x}_*^- - \boldsymbol{x}\|_2 < \min_{\boldsymbol{x}^+ \in \mathcal{X}^+} \|\boldsymbol{x}_*^- - \boldsymbol{x}^+\|_2$. The remaining case $\|\boldsymbol{x}_*^- - \boldsymbol{x}\|_2 > \min_{\boldsymbol{x}^+ \in \mathcal{X}^+} \|\boldsymbol{x}_*^- - \boldsymbol{x}^+\|_2$ can be proved with only minor changes and is omitted.

3.4 Proof of Theorem 11

The key observation to prove Theorem 11 is stated in Lemma 19. To prove it we will need two ingredients. The first is a slight modification of (Dirksen et al., 2022a, Theorem 26).

Lemma 17 *There exists an absolute constant $c > 0$ such that the following holds. Let $\mathcal{X}^-, \mathcal{X}^+ \subset R\mathbb{B}_2^d$ be δ-separated sets with $N^- := |\mathcal{X}^-|$, $N^+ := |\mathcal{X}^+|$. Let $\boldsymbol{W} \in \mathbb{R}^{n\times d}$ be a matrix with standard Gaussian entries, $\boldsymbol{b} \in \mathbb{R}^n$ be uniformly distributed in $[-\lambda, \lambda]^n$ and let $\boldsymbol{W}$ and b be independent. Consider the associated random threshold layer $\Phi\colon \mathbb{R}^d \to \mathbb{R}^n$*

$$\Phi(\boldsymbol{x}) = \operatorname{Thres}(\boldsymbol{W}\boldsymbol{x} + \boldsymbol{b}), \quad \boldsymbol{x} \in \mathbb{R}^d.$$

Suppose that $\lambda \gtrsim R$ and

$$n \gtrsim \delta^{-1}\lambda \cdot \log(2N^-N^+/\eta). \tag{10}$$

Then with probability at least $1-\eta$, the following event occurs: For every $\boldsymbol{x}^- \in \mathcal{X}^-$, the vector $\boldsymbol{u}_{\boldsymbol{x}^-} \in \{0,1\}^n$

$$(\boldsymbol{u}_{\boldsymbol{x}^-})_i = \begin{cases} 1, & (\Phi(\boldsymbol{x}^-))_i = 0, \\ 0, & \textit{otherwise}, \end{cases} \tag{11}$$

satisfies $\langle \boldsymbol{u}_{\boldsymbol{x}^-}, \Phi(\boldsymbol{x}^-)\rangle = 0$ and

$$\langle \boldsymbol{u}_{\boldsymbol{x}^-}, \Phi(\boldsymbol{x}^+)\rangle \geq c\|\boldsymbol{x}^+ - \boldsymbol{x}^-\|_2 \cdot \lambda^{-1} n \qquad \textit{for all } \boldsymbol{x}^+ \in \mathcal{X}^+.$$

Geometrically, Lemma 17 states that with high probability the hyperplane $H[\boldsymbol{u}_{\boldsymbol{x}^-}, 0]$ linearly separates $\Phi(\boldsymbol{x}^-)$ from $\Phi(\mathcal{X}^+)$ and the separation margin increases with both n and the distance between $\boldsymbol{x}^-$ and $\mathcal{X}^+$.

Proof By (11) it is clear that $\langle \boldsymbol{u}_{\boldsymbol{x}^-}, \Phi(\boldsymbol{x}^-)\rangle = 0$. Let $\boldsymbol{W} = [\boldsymbol{w}_1, \ldots, \boldsymbol{w}_n]^\top \in \mathbb{R}^{n\times d}$ and $\boldsymbol{b} = (b_1, \ldots, b_n)^\top \in \mathbb{R}^n$ be the weight matrix and bias vector of Φ, respectively. For $\boldsymbol{x}^- \in \mathcal{X}^-$ and $\boldsymbol{x}^+ \in \mathcal{X}^+$ define

$$\mathcal{I}_{\boldsymbol{x}^-,\boldsymbol{x}^+} = \{i \in [n] : \langle \boldsymbol{w}_i, \boldsymbol{x}^-\rangle \leq -b_i < \langle \boldsymbol{w}_i, \boldsymbol{x}^+\rangle\},$$

and define the events

$$B^i_{\boldsymbol{x}^-,\boldsymbol{x}^+} = \{H[\boldsymbol{w}_i, b_i] \ \left\|\boldsymbol{x}^+ - \boldsymbol{x}^-\right\|_2\text{-separates } \boldsymbol{x}^- \text{ from } \boldsymbol{x}^+\},$$

For $n'(\boldsymbol{x}^-, \boldsymbol{x}^+) > 0$ to be specified later, set

$$B_{\boldsymbol{x}^-,\boldsymbol{x}^+,n'(\boldsymbol{x}^-,\boldsymbol{x}^+)} = \Big\{ \sum_{i=1}^{n} \mathbb{1}_{B^i_{\boldsymbol{x}^-,\boldsymbol{x}^+}} \geq n'(\boldsymbol{x}^-, \boldsymbol{x}^+) \Big\},$$
$$B = \bigcap_{(\boldsymbol{x}^-,\boldsymbol{x}^+)\in\mathcal{X}^-\times\mathcal{X}^+} B_{\boldsymbol{x}^-,\boldsymbol{x}^+,n'(\boldsymbol{x}^-,\boldsymbol{x}^+)}.$$

On the event B, for every $\boldsymbol{x}^- \in \mathcal{X}^-$ and $\boldsymbol{x}^+ \in \mathcal{X}^+$,

$$\langle \boldsymbol{u}_{\boldsymbol{x}^-}, \Phi(\boldsymbol{x}^+)\rangle = \sum_{i\in\mathcal{I}_{\boldsymbol{x}^-,\boldsymbol{x}^+}} \text{Thres}(\langle \boldsymbol{w}_i, \boldsymbol{x}^+\rangle + b_i) = |\mathcal{I}_{\boldsymbol{x}^-,\boldsymbol{x}^+}| \geq n'(\boldsymbol{x}^-, \boldsymbol{x}^+).$$

For every $i \in [n]$, Lemma 16 implies that $\mathbb{P}(B^i_{\boldsymbol{x}^-,\boldsymbol{x}^+}) \geq c\left\|\boldsymbol{x}^+ - \boldsymbol{x}^-\right\|_2 \lambda^{-1}$ for an absolute constant $c > 0$ if $\lambda \gtrsim R$. Therefore, Chernoff's inequality for sums of independent Bernoulli random variables (see, e.g., Vershynin, 2018, Section 2.3) implies that

$$\mathbb{P}\left(\sum_{i=1}^{n} \mathbb{1}_{B^i_{\boldsymbol{x}^-,\boldsymbol{x}^+}} \leq \frac{c}{2}\lambda^{-1}\left\|\boldsymbol{x}^+ - \boldsymbol{x}^-\right\|_2 n\right) \leq \exp(-c'\lambda^{-1}\left\|\boldsymbol{x}^+ - \boldsymbol{x}^-\right\|_2 n),$$

where $c' > 0$ is an absolute constant. Setting $n'(\boldsymbol{x}^-, \boldsymbol{x}^+) = \frac{c}{2}\left\|\boldsymbol{x}^+ - \boldsymbol{x}^-\right\|_2 \lambda^{-1} n$, we obtain

$$\mathbb{P}(B^c_{\boldsymbol{x}^-,\boldsymbol{x}^+,n'(\boldsymbol{x}^-,\boldsymbol{x}^+)}) \leq \exp(-c'\lambda^{-1}\left\|\boldsymbol{x}^+ - \boldsymbol{x}^-\right\|_2 n) \leq \exp(-c'\lambda^{-1}\delta n).$$

Hence, by the union bound and (10),

$$\mathbb{P}(B^c) \leq N^- N^+ \exp(-c'\lambda^{-1}\delta n) \leq \eta.$$

■

Our second proof ingredient is the following lemma. It is an immediate consequence of (Dirksen and Mendelson, 2021, Theorem 2.9).

Lemma 18 *Consider $\boldsymbol{c}_1, \ldots, \boldsymbol{c}_M \subset \mathbb{R}^d$ and $\mathcal{X}_1, \ldots, \mathcal{X}_M \subset \mathbb{R}^d$ such that $\mathcal{X}_j \subset \mathbb{B}^d_2(\boldsymbol{c}_j, r_j) \subset R\mathbb{B}^d_2$ for all $j \in [M]$. Let*

$$r_j \lesssim \frac{r'_j}{\sqrt{\log(e\lambda/r'_j)}}, \qquad r' = \min_{j\in[M]} r'_j.$$

Let further $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_n \sim N(\boldsymbol{0}, \boldsymbol{I}_d)$ and $b_1, \ldots, b_n \sim \text{Unif}([-\lambda, \lambda])$ all be independent. If $\lambda \gtrsim R$ and

$$n \gtrsim \frac{\lambda}{r'}\log(2M/\eta) + \max_{j\in[M]} \frac{\lambda}{(r'_j)^3} w^2(\mathcal{X}_j - \boldsymbol{c}_j),$$

then, with probability at least $1 - \eta$, for all $j \in [M]$ and $\boldsymbol{x} \in \mathcal{X}_j$,

$$|\{i \in [n] \ : \ \text{Thres}(\langle \boldsymbol{w}_i, \boldsymbol{c}_j\rangle + b_i) \neq \text{Thres}(\langle \boldsymbol{w}_i, \boldsymbol{x}\rangle + b_i)\}| \lesssim \frac{r'_j n}{\lambda}.$$

The following result shows that the 'dedicated' neuron $\hat{\varphi}_{\boldsymbol{x}_*^-}$ associated with $\boldsymbol{x}_*^-$ (defined in (5)) not only separates $\Phi(\boldsymbol{x}_*^-)$ and $\Phi(\mathcal{X}^+)$, but in fact acts as a robust separator: it also separates $\Phi(\boldsymbol{x}^-)$ and $\Phi(\mathcal{X}^+)$ for all points $\boldsymbol{x}^-$ in the component of the mutual covering in which $\boldsymbol{x}_*^-$ resides.

Lemma 19 *Consider the setting of Theorem 11. For $\boldsymbol{x}_*^- \in \mathcal{X}^-$ we define the associated neuron $\hat{\varphi}_{\boldsymbol{x}_*^-} \colon \mathbb{R}^n \to \{0,1\}$ by*

$$\hat{\varphi}_{\boldsymbol{x}_*^-}(\boldsymbol{z}) = \mathrm{Thres}(-\langle \boldsymbol{u}_{\boldsymbol{x}_*^-}, \boldsymbol{z}\rangle + m_{\boldsymbol{x}_*^-}),$$

where

$$\boldsymbol{u}_{\boldsymbol{x}_*^-} = \mathbb{1}[\Phi(\boldsymbol{x}_*^-) = \mathbf{0}] \quad \textit{and} \quad m_{\boldsymbol{x}_*^-} = \min_{\boldsymbol{x}^+ \in \mathcal{X}^+} \langle \boldsymbol{u}_{\boldsymbol{x}_*^-}, \Phi(\boldsymbol{x}^+)\rangle.$$

Then, with probability at least $1-\eta$, for all $\ell \in [M^-]$ and $\boldsymbol{x}_*^- \in \mathcal{X}_\ell^-$,

$$\hat{\varphi}_{\boldsymbol{x}_*^-}(\Phi(\boldsymbol{x}^-)) > 0 \quad \textit{for all } \boldsymbol{x}^- \in \mathcal{X}_\ell^-, \tag{12}$$

and

$$\hat{\varphi}_{\boldsymbol{x}_*^-}(\Phi(\boldsymbol{x}^+)) = 0 \quad \textit{for all } \boldsymbol{x}^+ \in \mathcal{X}^+. \tag{13}$$

Proof Clearly, the choice of $m_{\boldsymbol{x}_*^-}$ ensures that (13) holds. It remains to show that, with probability at least $1-\eta$,

$$\langle \boldsymbol{u}_{\boldsymbol{x}_*^-}, \Phi(\boldsymbol{x}^-)\rangle < m_{\boldsymbol{x}_*^-} = \min_{\boldsymbol{x}^+ \in \mathcal{X}^+} \langle \boldsymbol{u}_{\boldsymbol{x}_*^-}, \Phi(\boldsymbol{x}^+)\rangle$$

for all $\ell \in [M^-]$ and $\boldsymbol{x}_*^-, \boldsymbol{x}^- \in \mathcal{X}_\ell^-$. Let A be the event where, for every $\ell \in [M^-]$ and $j \in [M^+]$,

$$\langle \boldsymbol{u}_{\boldsymbol{c}_\ell^-}, \Phi(\boldsymbol{c}_j^+)\rangle \geq c_1 \lambda^{-1} \left\| \boldsymbol{c}_\ell^- - \boldsymbol{c}_j^+ \right\|_2 n.$$

By Lemma 17, $\mathbb{P}(A) \geq 1-\eta$ under our assumptions. Let B be the event where, for all $\ell \in [M^-]$ and $\boldsymbol{x}^- \in \mathcal{X}_\ell^-$,

$$\begin{aligned}\left\|\Phi(\boldsymbol{c}_\ell^-) - \Phi(\boldsymbol{x}^-)\right\|_1 &= |\{i \in [n] : \mathrm{Thres}(\langle \boldsymbol{w}_i, \boldsymbol{c}_\ell^-\rangle + b_i) \neq \mathrm{Thres}(\langle \boldsymbol{w}_i, \boldsymbol{x}^-\rangle + b_i)\}| \\ &\leq c_2 \frac{(r'_\ell)^- n}{\lambda},\end{aligned}$$

and, for all $j \in [M^+]$ and $\boldsymbol{x}^+ \in \mathcal{X}_j^+$,

$$\begin{aligned}\left\|\Phi(\boldsymbol{c}_j^+) - \Phi(\boldsymbol{x}^+)\right\|_1 &= |\{i \in [n] : \mathrm{Thres}(\langle \boldsymbol{w}_i, \boldsymbol{c}_j^+\rangle + b_i) \neq \mathrm{Thres}(\langle \boldsymbol{w}_i, \boldsymbol{x}^+\rangle + b_i)\}| \\ &\leq c_2 \frac{(r'_j)^+ n}{\lambda},\end{aligned}$$

where

$$(r'_\ell)^- = \frac{c_1}{12 c_2} \mathrm{d}(\boldsymbol{c}_\ell^-, \mathcal{C}^+), \qquad (r'_j)^+ = \frac{c_1}{4 c_2} \mathrm{d}(\boldsymbol{c}_j^+, \mathcal{C}^-).$$

By Lemma 18, $\mathbb{P}(B) \geq 1-\eta$ under the stated assumptions. For the remainder of the proof, we condition on the event $A \cap B$.

By using B, we find

$$\begin{aligned}|\langle \boldsymbol{u}_{\boldsymbol{x}_*^-}, \Phi(\boldsymbol{x}^-)\rangle| &= |\langle \boldsymbol{u}_{\boldsymbol{x}_*^-}, \Phi(\boldsymbol{x}^-) - \Phi(\boldsymbol{x}_*^-)\rangle| \\ &\leq \left\|\Phi(\boldsymbol{x}^-) - \Phi(\boldsymbol{x}_*^-)\right\|_1 \\ &\leq \left\|\Phi(\boldsymbol{x}^-) - \Phi(\boldsymbol{c}_\ell^-)\right\|_1 + \left\|\Phi(\boldsymbol{c}_\ell^-) - \Phi(\boldsymbol{x}_*^-)\right\|_1 \\ &\leq 2c_2 \frac{(r'_\ell)^-}{\lambda} n.\end{aligned}$$

Now pick $j \in [M^+]$ and $\boldsymbol{x}^+ \in \mathcal{X}_j^+$. Using A and B,

$$\begin{aligned}&\langle \boldsymbol{u}_{\boldsymbol{x}_*^-}, \Phi(\boldsymbol{x}^+)\rangle \\ &\quad= \langle \boldsymbol{u}_{\boldsymbol{c}_\ell^-}, \Phi(\boldsymbol{c}_j^+)\rangle + \langle \boldsymbol{u}_{\boldsymbol{x}_*^-} - \boldsymbol{u}_{\boldsymbol{c}_\ell^-}, \Phi(\boldsymbol{c}_j^+)\rangle + \langle \boldsymbol{u}_{\boldsymbol{x}_*^-}, \Phi(\boldsymbol{x}^+) - \Phi(\boldsymbol{c}_j^+)\rangle \\ &\quad\geq \langle \boldsymbol{u}_{\boldsymbol{c}_\ell^-}, \Phi(\boldsymbol{c}_j^+)\rangle - |\langle \Phi(\boldsymbol{x}_*^-) - \Phi(\boldsymbol{c}_\ell^-), \Phi(\boldsymbol{c}_j^+)\rangle| - |\langle \boldsymbol{u}_{\boldsymbol{x}_*^-}, \Phi(\boldsymbol{x}^+) - \Phi(\boldsymbol{c}_j^+)\rangle| \\ &\quad\geq \langle \boldsymbol{u}_{\boldsymbol{c}_\ell^-}, \Phi(\boldsymbol{c}_j^+)\rangle - \left\|\Phi(\boldsymbol{x}_*^-) - \Phi(\boldsymbol{c}_\ell^-)\right\|_1 - \left\|\Phi(\boldsymbol{x}^+) - \Phi(\boldsymbol{c}_j^+)\right\|_1 \\ &\quad\geq c_1\lambda^{-1}\left\|\boldsymbol{c}_\ell^- - \boldsymbol{c}_j^+\right\|_2 n - c_2\frac{(r'_\ell)^-}{\lambda}n - c_2\frac{(r'_j)^+}{\lambda}n,\end{aligned}$$

where in the second step we used that $\boldsymbol{u}_{\boldsymbol{x}} = \boldsymbol{1} - \Phi(\boldsymbol{x})$ due to the threshold activation.

Combining the above we see that, for all $\ell \in [M^-]$, $\boldsymbol{x}_*^-, \boldsymbol{x}^- \in \mathcal{X}_\ell^-$, $j \in [M^+]$, and $\boldsymbol{x}^+ \in \mathcal{X}_j^+$,

$$\langle \boldsymbol{u}_{\boldsymbol{x}_*^-}, \Phi(\boldsymbol{x}^-)\rangle < \langle \boldsymbol{u}_{\boldsymbol{x}_*^-}, \Phi(\boldsymbol{x}^+)\rangle,$$

where we have used that

$$(r'_\ell)^- < \frac{c_1}{6c_2}\left\|\boldsymbol{c}_\ell^- - \boldsymbol{c}_j^+\right\|_2, \qquad (r'_j)^+ < \frac{c_1}{2c_2}\left\|\boldsymbol{c}_\ell^- - \boldsymbol{c}_j^+\right\|_2.$$

Since for any $\boldsymbol{x}^+ \in \mathcal{X}^+$ there is some $j \in [M^+]$ such that $\boldsymbol{x}^+ \in \mathcal{X}_j^+$, we find for all $\ell \in [M^-]$ and $\boldsymbol{x}_*^-, \boldsymbol{x}^- \in \mathcal{X}_\ell^-$,

$$\langle \boldsymbol{u}_{\boldsymbol{x}_*^-}, \Phi(\boldsymbol{x}^-)\rangle < \min_{\boldsymbol{x}^+ \in \mathcal{X}^+} \langle \boldsymbol{u}_{\boldsymbol{x}_*^-}, \Phi(\boldsymbol{x}^+)\rangle = m_{\boldsymbol{x}_*^-},$$

as desired. ■

We can now complete the proof.

Proof [Theorem 11] Throughout, we condition on the event from Lemma 19. Let us first observe that the first loop of Algorithm 1 terminates after $n_{\min}$ iterations and hence the first layer Φ of F has width $n_{\min}$. Indeed, taking $\boldsymbol{x}_*^- = \boldsymbol{x}^-$ in (12), we see that $\hat{\varphi}_{\boldsymbol{x}^-}(\boldsymbol{x}^-) > 0$ for any $\boldsymbol{x}^- \in \mathcal{X}^-$ and hence

$$0 = \langle \boldsymbol{u}_{\boldsymbol{x}^-}, \Phi(\boldsymbol{x}^-)\rangle < m_{\boldsymbol{x}^-} = \min_{\boldsymbol{x}^+ \in \mathcal{X}^+} \langle \boldsymbol{u}_{\boldsymbol{x}^-}, \Phi(\boldsymbol{x}^+)\rangle.$$

This estimate implies that for all $\boldsymbol{x}^+ \in \mathcal{X}^+$, there must be a hyperplane that separates $\boldsymbol{x}^-$ from $\boldsymbol{x}^+$.

Next, using induction we show that the second loop terminates after at most M^- steps, thus $\hat{n} \leq M^-$. In the first iteration, we select $\boldsymbol{x}_1^- \in \mathcal{C} = \mathcal{X}^-$ which is part of at least one component of the mutual covering, say $\mathcal{X}^-_{i_1}$. By Lemma 19, the associated neuron $\hat{\varphi}_{\boldsymbol{x}^-_1}$ activates on all of $\mathcal{X}^-_{i_1}$ and hence $\mathcal{C} \cap \mathcal{X}^-_{i_1} = \emptyset$ after the update. Suppose that the p-th iteration finished, thus $\mathcal{C} \cap \mathcal{X}^-_{i_j} = \emptyset$ for all $j \in [p]$. We select $\boldsymbol{x}^-_{p+1} \in \mathcal{C} \subset \mathcal{X}^- \setminus (\mathcal{X}^-_{i_1} \cup \cdots \cup \mathcal{X}^-_{i_p})$ which must be part of a new component, say $\mathcal{X}^-_{i_{p+1}}$. Again, by the lemma the associated neuron activates on all of the component, and thus, after the update $\mathcal{C} \cap \mathcal{X}^-_{i_j} = \emptyset$ for all $j \in [p+1]$. By induction, after at most M^- iterations $\mathcal{C} = \emptyset$ and hence the algorithm terminates with $\hat{n} \leq M^-$. ■

3.5 Proof of Corollary 5

Let $\mathcal{C}^-$ and $\mathcal{C}^+$ denote the centers of the mutual covering. We apply Algorithm 1 to $\mathcal{C}^-$ and $\mathcal{C}^+$ (with $\lambda \approx R$ and $n_{\min}$ from Theorem 11) with the following change: when computing the biases in the second layer, instead of taking the minimum only over $\mathcal{C}^+$ we set, for all $\ell \in [M^-]$,

$$m_{\boldsymbol{c}^-_\ell} = \min_{\boldsymbol{x}^+ \in \mathcal{X}^+} \langle \boldsymbol{u}_{\boldsymbol{c}^-_\ell}, \Phi(\boldsymbol{x}^+) \rangle.$$

Inspecting the proof of Theorem 11, we see that with positive probability this network has the asserted size and, moreover, interpolates $\mathcal{X}^-$ and $\mathcal{X}^+$.

3.6 Proof of Proposition 13

Before we construct a data set that satisfies the properties of the proposition, we make some preliminary observations. Consider any $\mathcal{X}^-, \mathcal{X}^+ \subset \mathbb{R}^d$. Let Φ denote the first layer of the output F of Algorithm 1. For a given $\boldsymbol{x}^-_* \in \mathcal{X}^-$, consider its associated neuron $\varphi_{\boldsymbol{x}^-_*}$ defined in (5). Consider $\boldsymbol{x}^-_* \neq \boldsymbol{x}^- \in \mathcal{X}^-$ and for $t \geq 0$ set

$$\boldsymbol{x}_t = \boldsymbol{x}^-_* + t(\boldsymbol{x}^- - \boldsymbol{x}^-_*),$$

so that $\{\boldsymbol{x}_t \ : \ t \geq 0\}$ is the ray originating from $\boldsymbol{x}^-_*$ and passing through $\boldsymbol{x}^-$.

First, we claim that $t \mapsto \langle \boldsymbol{u}_{\boldsymbol{x}^-_*}, \Phi(\boldsymbol{x}_t) \rangle$ is non-decreasing. This is an immediate consequence of the fact that $\Phi_i(\boldsymbol{x}_t) \leq \Phi_i(\boldsymbol{x}_s)$ for all $i \in [n]$ such that $\Phi_i(\boldsymbol{x}^-_*) = 0$ and for all $0 \leq t \leq s$. This is clear in the case $\Phi_i(\boldsymbol{x}_t) = 0$. Assuming $\Phi_i(\boldsymbol{x}_t) > 0$ (and hence $t > 0$), the assumptions imposed on σ imply that

$$0 < \langle \boldsymbol{w}_i, \boldsymbol{x}_t \rangle + b_i = \langle \boldsymbol{w}_i, \boldsymbol{x}^-_* \rangle + b_i + t \langle \boldsymbol{w}_i, \boldsymbol{x}^- - \boldsymbol{x}^-_* \rangle.$$

As $t > 0$ and $\langle \boldsymbol{w}_i, \boldsymbol{x}^-_* \rangle + b_i \leq 0$ due to $\Phi_i(\boldsymbol{x}^-_*) = 0$, it follows that

$$\langle \boldsymbol{w}_i, \boldsymbol{x}^- - \boldsymbol{x}^-_* \rangle > 0.$$

Finally, since σ is non-decreasing,

$$\begin{aligned}
\Phi_i(\boldsymbol{x}_t) &= \sigma(\langle \boldsymbol{w}_i, \boldsymbol{x}_t \rangle + b_i) \\
&\leq \sigma(\langle \boldsymbol{w}_i, \boldsymbol{x}_t \rangle + b_i + (s-t)\langle \boldsymbol{w}_i, \boldsymbol{x}^- - \boldsymbol{x}^-_* \rangle) \\
&= \sigma(\langle \boldsymbol{w}_i, \boldsymbol{x}_s \rangle + b_i) = \Phi_i(\boldsymbol{x}_s),
\end{aligned}$$

proving our claim.

Now let us make the following observation: suppose there is $\boldsymbol{x}^+ \in \mathcal{X}^+$ which lies between $\boldsymbol{x}_*^-$ and $\boldsymbol{x}^-$ in the sense that there exists $t^+ \in (0,1)$ such that $\boldsymbol{x}_{t^+} = \boldsymbol{x}^+$. Then, the neuron $\hat{\varphi}_{\boldsymbol{x}_*^-}$ does not activate on $\Phi(\boldsymbol{x}^-)$. To see this, we simply invoke the above claim, which yields

$$m_{\boldsymbol{x}_*^-} \leq \langle \boldsymbol{u}_{\boldsymbol{x}_*^-}, \Phi(\boldsymbol{x}^+) \rangle \leq \langle \boldsymbol{u}_{\boldsymbol{x}_*^-}, \Phi(\boldsymbol{x}^-) \rangle,$$

and directly implies $\hat{\varphi}_{\boldsymbol{x}_*^-}(\Phi(\boldsymbol{x}^-)) = 0$.

With these observations, we can now prove the statement of the proposition. Consider the interval $[0,1]$ and place points $\boldsymbol{c}_\ell^-$ and $\boldsymbol{c}_j^+$ in an alternating fashion on an equispaced grid: formally, for $\ell \in [M^-]$ and $j \in [M^+]$ we set

$$\boldsymbol{c}_\ell^- = \frac{\ell - 1}{M^- - 1} \quad \text{and} \quad \boldsymbol{c}_j^+ = \frac{j - 1/2}{M^- - 1}.$$

Let r_ℓ^- and r_j^+ be as in Theorem 11. Choose the remaining $N^- - M^-$ points $\boldsymbol{x}^- \in \mathcal{X}^-$ and $N^+ - M^+$ points $\boldsymbol{x}^+ \in \mathcal{X}^+$ such that for each of them there exists $\ell \in [M^-]$ with $\left\|\boldsymbol{x}^- - \boldsymbol{c}_\ell^-\right\|_2 \leq r_\ell^-$ and $j \in [M^+]$ with $\left\|\boldsymbol{x}^+ - \boldsymbol{c}_j^+\right\|_2 \leq r_j^+$, respectively. Then, $\mathcal{C}^- = \{\boldsymbol{c}_1^-, \ldots, \boldsymbol{c}_{M^-}^-\}$ and $\mathcal{C}^+ = \{\boldsymbol{c}_1^+, \ldots, \boldsymbol{c}_{M^+}^+\}$ form a mutual covering of $\mathcal{X}^-$ and $\mathcal{X}^+$ as required by Theorem 11.

Let $\ell \in [M^-]$ be fixed. By our earlier observation, for each $\boldsymbol{x}^- \in \mathcal{X}^- \backslash \mathcal{X}_\ell^-$, $\varphi_{\boldsymbol{x}^-}(\Phi(\boldsymbol{c}_\ell^-)) = 0$, as there is a point $\boldsymbol{c}_j^+ \in \mathcal{X}^+$ between $\boldsymbol{x}^-$ and $\boldsymbol{c}_\ell^-$. Thus, to classify $\boldsymbol{c}_\ell^-$ correctly, we need to choose (at least) one neuron corresponding to a point in $\mathcal{X}_\ell^-$. As we need to classify the points $\boldsymbol{c}_\ell^-$ for all $\ell \in [M^-]$ correctly, we cannot include less than M^- neurons in the second layer.

4. Numerical Experiments

In this section, we study the performance of Algorithm 1 through numerical simulations on different data sets.[3] In particular, we want to investigate how the interpolation probability (approximated as the fraction of a fixed amount of runs that produce an interpolating network) and the width of the second layer respond to changes in the width of the first layer n and the maximal bias λ. Recall that the algorithm was designed in such a way that it adapts the width of the first layer to guarantee interpolation on the input data. To have free control over this parameter we adapt the algorithm slightly for the experiments.

Hence, we formulate Algorithm 2 which has both n and λ as hyperparameters. As the first layer might be such that not every pair of samples with different labels is separated by at least one hyperplane, we have to adjust the construction of the second layer. We keep track of the set $\mathcal{C}$ of candidate samples whose associated neurons might be accepted into the second layer, the set $\mathcal{U}$ of samples that have yet to be correctly classified by a neuron (the universe), and the set $\mathcal{A}$ of samples whose associated neurons have been accepted into the second layer. Note that $\mathcal{C} \subset \mathcal{U}$ but there might not be equality. The algorithm stops if we either run out of candidates or all points are classified correctly. In every iteration, we

3. Code is available at `https://github.com/patrickfinke/memo`. We use Python 3, Scikit-learn, and NumPy.

draw a candidate sample at random and compute the associated neuron. If the neuron at least correctly classifies the candidate itself, we accept it into the second layer and remove every point that the neuron classifies correctly from both $\mathcal{C}$ and $\mathcal{U}$. This check could be omitted in Algorithm 1 due to the construction of the first layer which also guaranteed that $\mathcal{C} = \mathcal{U}$.

Algorithm 2 Interpolation (experiments)

```
Input: Disjoint and finite $\mathcal{X}^-, \mathcal{X}^+ \subset \mathbb{R}^d$ with $N^- := |\mathcal{X}^-|$, $N^+ := |\mathcal{X}^+|$, activation $\sigma\colon \mathbb{R} \to \mathbb{R}$ satisfying $\sigma(t) = 0$ for $t \leq 0$ and $\sigma(t) > 0$ for $t > 0$, width of first layer $n \geq 1$, maximal bias $\lambda \geq 0$.
Output: A three-layer fully-connected neural network $F\colon \mathbb{R}^d \to \{\pm 1\}$.
--------------------------------------------------------------------------------
1:  Randomly sample $\boldsymbol{W} \in \mathbb{R}^{n\times d}$ and $\boldsymbol{b} \in \mathbb{R}^n$ where
        $$\boldsymbol{W}_i \sim N(\boldsymbol{0}, \boldsymbol{I}_d) \quad \text{and} \quad b_i \sim \mathrm{Unif}([-\lambda, \lambda]).$$
    are all independent and define the first layer $\Phi(\boldsymbol{x}) = \sigma(\boldsymbol{W}\boldsymbol{x} + \boldsymbol{b})$.
--------------------------------------------------------------------------------
2:  Initialize $\mathcal{C} \leftarrow \mathcal{X}^-$, $\mathcal{U} \leftarrow \mathcal{X}^-$ and $\mathcal{A} \leftarrow \emptyset$.
3:  while $\mathcal{C} \neq \emptyset$ and $\mathcal{U} \neq \emptyset$ do
4:      Select a candidate $\boldsymbol{x}_*^- \in \mathcal{C}$ at random and update $\mathcal{C} \leftarrow \mathcal{C} \setminus \{\boldsymbol{x}_*^-\}$.
5:      Calculate $\boldsymbol{u}_{\boldsymbol{x}_*^-} \in \{0,1\}^n$ and $m_{\boldsymbol{x}_*^-} \geq 0$ according to
            $$\boldsymbol{u}_{\boldsymbol{x}_*^-} \leftarrow \mathbb{1}[\Phi(\boldsymbol{x}_*^-) = \boldsymbol{0}] \quad \text{and} \quad m_{\boldsymbol{x}_*^-} \leftarrow \min_{\boldsymbol{x}^+ \in \mathcal{X}^+} \langle \boldsymbol{u}_{\boldsymbol{x}_*^-}, \Phi(\boldsymbol{x}^+)\rangle.$$
6:      if $m_{\boldsymbol{x}_*^-} > 0$ then
7:          Calculate $\mathcal{T} \leftarrow \{\boldsymbol{x}^- \in \mathcal{U} : \langle \boldsymbol{u}_{\boldsymbol{x}_*^-}, \Phi(\boldsymbol{x}^-)\rangle < m_{\boldsymbol{x}_*^-}\}$.
8:          Update $\mathcal{C}$, $\mathcal{U}$ and $\mathcal{A}$ according to
                $$\mathcal{C} \leftarrow \mathcal{C} \setminus \mathcal{T}, \quad \mathcal{U} \leftarrow \mathcal{U} \setminus \mathcal{T} \quad \text{and} \quad \mathcal{A} \leftarrow \mathcal{A} \cup \{\boldsymbol{x}_*^-\}.$$
9:      end if
10: end while
11: Define $\hat{\Phi}(\boldsymbol{z}) = \sigma(-\boldsymbol{U}\boldsymbol{z} + \boldsymbol{m})$ with $\boldsymbol{U} \in \mathbb{R}^{|\mathcal{A}|\times n}$ and $\boldsymbol{m} \in \mathbb{R}^{|\mathcal{A}|}$ where
        $$\boldsymbol{U} \leftarrow \Big[\boldsymbol{u}_{\boldsymbol{x}_*^-}^\top\Big]_{\boldsymbol{x}_*^- \in \mathcal{A}} \quad \text{and} \quad \boldsymbol{m} \leftarrow \Big[m_{\boldsymbol{x}_*^-}\Big]_{\boldsymbol{x}_*^- \in \mathcal{A}}.$$
--------------------------------------------------------------------------------
12: Return $F(\boldsymbol{x}) = \mathrm{sign}(-\langle \boldsymbol{1}, \hat{\Phi}(\Phi(\boldsymbol{x}))\rangle)$.
```

In the following, we present five experiments. First, we focus on the verification of our theoretical results through illustrative experiments on simple data sets. In Section 4.1, we apply Algorithm 2 to the Two Moons data set, which allows us to verify our main result and illustrate the underlying geometric intuition. In Section 4.2 we verify that, in a controlled setting which is guaranteed to satisfy our assumptions, the network size indeed does not depend on the number of samples. Next, we examine the performance on real world data. In Section 4.3 we investigate binary classification subproblems of the MNIST data set. We introduce an extension to multi-class classification in Section 4.4 and apply it to MNIST. Additionally, in Section 4.5, we consider the CIFAR-10 data set. Finally, we present a worst-case example in Section 4.6. In all experiments, we let σ be the threshold activation.

4.1 Binary Classification on Two Moons

In this section, we apply Algorithm 2 to the 2D Two Moons[4] data set (Figure 5a), allowing us to easily visualize the output of the algorithm in the input domain. While this is only a synthetic toy data set, it provides a clear geometric structure with well-separated classes. At the same time, the data is not linearly separable, and not all pairs of samples with different labels can be efficiently separated by hyperplanes that pass through the origin, making it a good first testing ground for the effect of the parameter λ.

Interpolation probability. In Figure 5b we observe a clear phase transition in the interpolation probability which is in line with the prediction of Theorem 11, where we treat all complexity terms depending on the data set as constant. As can be seen from the contour lines, for λ larger than the data radius, $n \gtrsim \lambda$ is enough to guarantee interpolation with any fixed probability. On the other hand, one can observe that a large enough λ is also necessary for efficient interpolation, as for $\lambda = 0$ interpolation does not happen for any value of n.

It is noteworthy that the optimal value of λ is smaller than the data radius. This is intuitive here, as a maximal bias exceeding the radius of $\mathcal{X}^+$ already guarantees the efficient separation of pairs of opposite labels in the first layer.

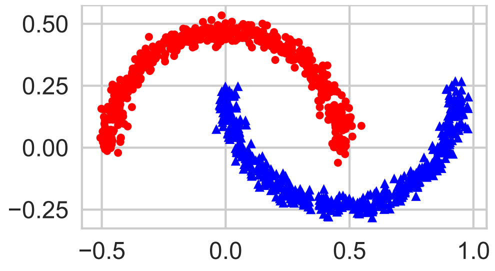
Width of the second layer $\hat{\Phi}$. As can be seen in Figure 5c, the width of the second layer becomes much smaller than the number of points. We are mainly interested in the part of the parameter space where the interpolation probability is close to one. In this region, the width attains its minimum and is essentially constant.

Due to the two-dimensionality of the data, it is possible to visualize the decision boundary of our method in input space, see Figure 6. Neurons of the second layer have (approximately) circular activation regions that are centered at their corresponding candidate points and which extend all the way to the other class. The third layer takes a union of these regions—the boundary of this union is the decision boundary. We can repeat this visualization for different values of the hyperparameters, see Figure 7. For $\lambda = 0$ the method fails to separate pairs of samples with opposite labels because all hyperplanes pass through the origin. If λ is large enough and as n grows, the method begins to succeed. In line with Proposition 10, the activation regions of the individual neurons become more circular as n increases, which can be best seen in the rightmost column of Figure 7.

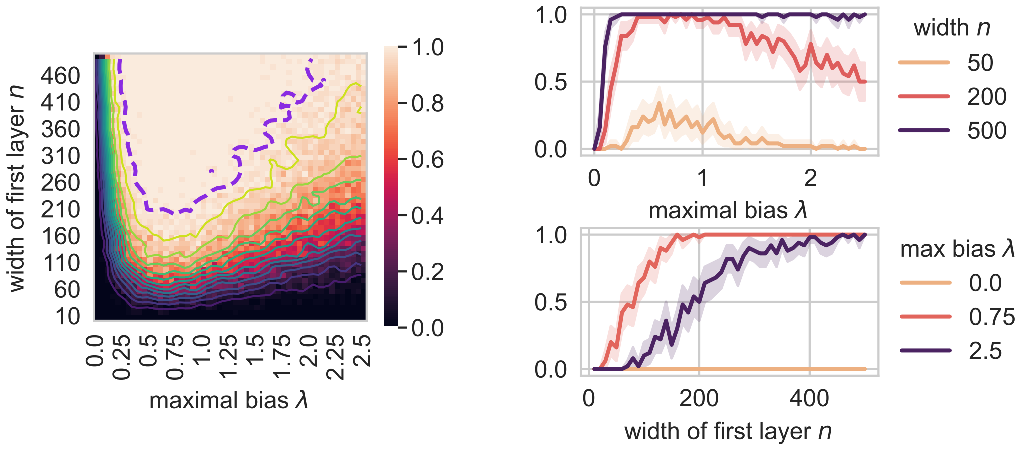
4.2 Behaviour in the Sample Size Limit

In Theorem 11, the size of the interpolating network is independent of the number of samples and only dictated by the parameters of the mutual covering. To illustrate this numerically,

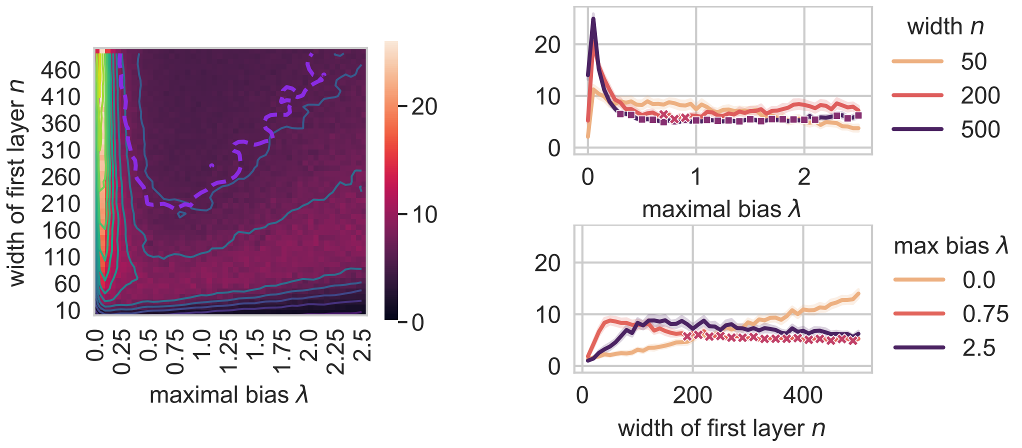
4. See `https://scikit-learn.org/stable/modules/generated/sklearn.datasets.make_moons.html`.



(a) **Two Moons.** A $d = 2$ dimensional data set of two interleaving half circles. Each class has $N^- = N^+ = 500$ samples and the radius is $R = 1$.



(b) **Interpolation probability.** (Left) The interpolation probability (average over 250 runs) as a function of the width of the first layer n and maximal bias λ. The 99% contour line is at the dashed purple line. (Right) Horizontal (top) and vertical (bottom) slices of the heatmap with 95% confidence intervals.



(c) **Width of the second layer $\hat{\Phi}$.** (Left) The width of the second layer $\hat{\Phi}$ (average over 250 runs) as a function of the width of the first layer n and maximal bias λ. The 99% interpolation probability contour line is at the dashed purple line. (Right) Horizontal (top) and vertical (bottom) slices of the heatmap with 95% confidence intervals. Markers indicate an interpolation probability $\geq 99\%$, compare Figure 5b.

Figure 5: Binary classification on the Two Moons data set.

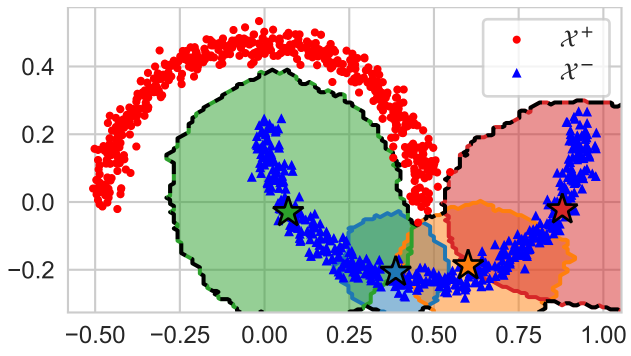


Figure 6: **Decision boundary.** Each star marks an accepted point and the region of the same color is the activation region of its associated neuron. The decision boundary of the network is the boundary of the union of these regions. Here, we used $n = 2\,000$ and $\lambda = 1$.

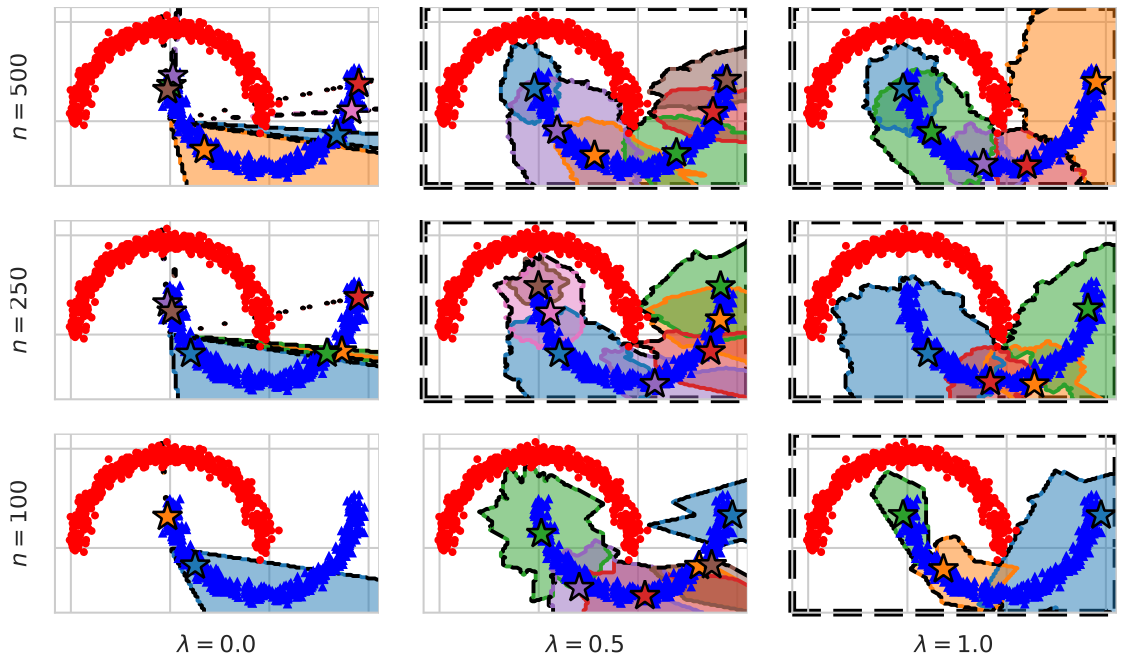


Figure 7: **Decision boundaries for different choices of hyper-parameters.** Similar to Figure 6 but includes all combinations of hyper-parameters $n \in \{100, 250, 500\}$ (rows) and $\lambda \in \{0, 0.5, 1\}$ (columns). Plots in which the network interpolates the data are marked with a thick dashed frame.

we consider a scenario where we sample points from a distribution whose support consists of two disjoint, compact sets representing two classes. We expect that as we iteratively sample points from the distribution, the size of the interpolating network should saturate and be bounded by the parameters of the mutual covering of the support of the distribution (satisfying the restrictions in Theorem 11).

To verify this, we return to the Two Moons data set from Section 4.1. We fix the maximal bias $\lambda = 1$ and vary the number of points N by drawing samples from the data distribution.[5]

Interpolation probability. The contour lines in the heatmap in Figure 8b show which width of the first layer is required to achieve interpolation with a fixed probability for a certain number of samples. We can observe that there is an increase in the required width up to around 800 000 samples. After this threshold, however, a constant width of the first layer is enough to interpolate any number of samples.

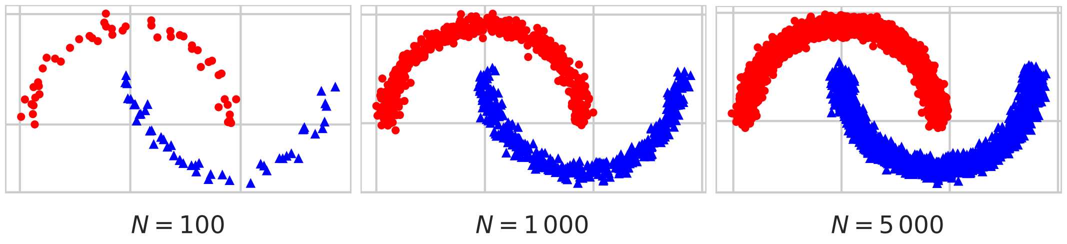
Width of the second layer $\hat{\Phi}$. As in the other experiments we are interested in the part of the parameter space where the interpolation probability is almost one. Similar to the contour lines of the interpolation probability we observe that to obtain a fixed width of the second layer there is an increase in the required width of the first layer only up to a certain threshold (again, around 800 000 samples). After this threshold, a constant width of the first layer is enough to obtain a fixed width of the second layer.

Combining the above observations we note the following: there is a threshold in the number of samples such that for larger sample sizes there is a width of the first layer for which the network interpolates with probability close to one and the width of the second layer stays constant. Hence, as the width of the second layer is only lower for smaller sample sizes, a neural network of constant size (whose parameters can be computed via our algorithm) suffices to interpolate any number of samples.

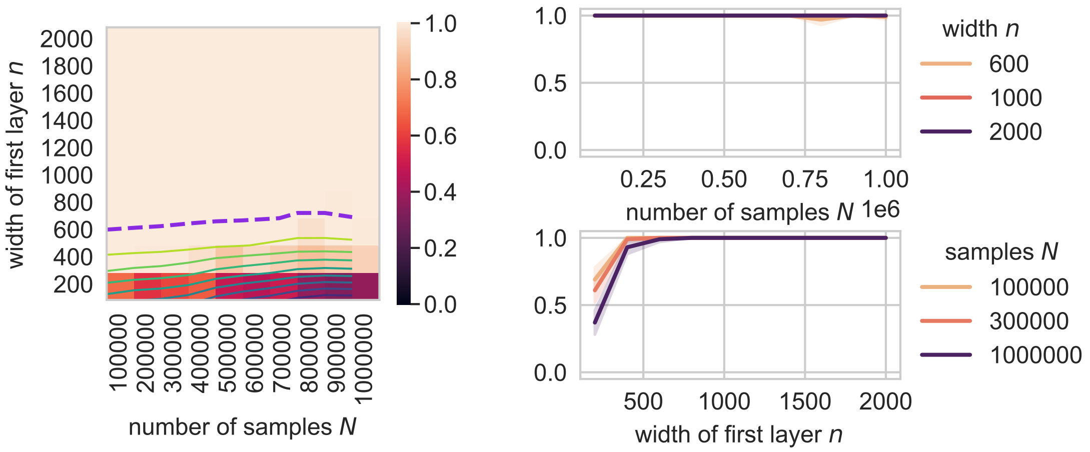
4.3 Binary Classification on MNIST

In the previous section, we ran a controlled experiment with a data generating distribution that was guaranteed to satisfy the assumptions of our main theorem and which could be used to draw an unlimited number of samples. It is natural to ask if the network size can also be observed to saturate in terms of the number of samples on real data. Examining binary classification subproblems of the MNIST data set (LeCun et al., 1998), we find that the answer is ‘only sometimes’. We illustrate this in Figure 9, which depicts the results for the ‘1 vs. 9’ and ‘1 vs. 8’ subproblems. For ‘1 vs. 9’, the second layer width clearly saturates as the number of samples grows. On the other hand, for ‘1 vs. 8’, although the curve seems to flatten a little, the second layer essentially grows linearly. For other binary subproblems, we observed that it was more common that the network size did not completely saturate. We emphasize that this does not contradict our claim that our approach yields a network of a size that is independent from the number of samples. Let us point to two possible explanations. First, MNIST may not contain enough samples to accurately represent the

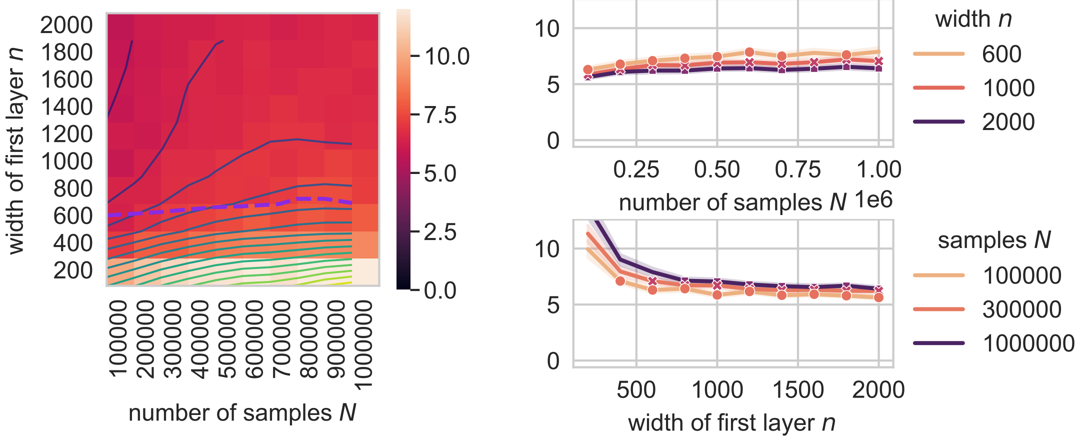
5. We use `sklearn.datasets.make_moons(n_samples=N, noise=0.05)` from the scikit-learn Python package to generate the samples.



(a) **Two Moons.** Continuously drawing samples from the distribution of Two Moons (Figure 5a) leads to a better representation of the support. Each class has always $N/2$ samples and the radius is $R = 1$.

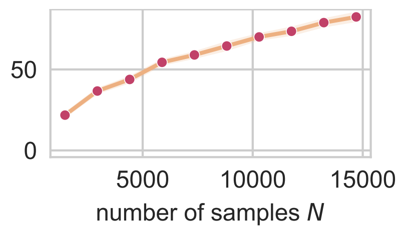


(b) **Interpolation probability.** (Left) The interpolation probability (average over 100 runs) as a function of the width of the first layer n and the number of samples N. The 99% contour line is at the dashed purple line. (Right) Horizontal (top) and vertical (bottom) slices of the heatmap with 95% confidence intervals.

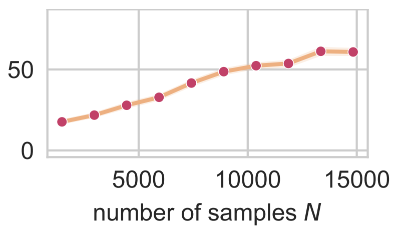


(c) **Width of the second layer $\hat{\Phi}$.** (Left) The width of the second layer $\hat{\Phi}$ (average over 100 runs) as a function of the width of the first layer n and the number of samples N. The 99% interpolation probability contour line is at the dashed purple line. (Right) Horizontal (top) and vertical (bottom) slices of the heatmap with 95% confidence intervals. Markers indicate an interpolation probability $\geq 99\%$, compare Figure 8b.

Figure 8: Sample size limit on the Two Moons data set.



(a) 1 vs. 8



(b) 1 vs. 9

Figure 9: **Width of the second layer $\hat{\Phi}$ on MNIST subproblems.** The width of the second layer $\hat{\Phi}$ (average over 25 runs) as a function of the number of samples N for two binary classification subproblems of MNIST. We fixed $n = 5\,000$ and $\lambda = 0.5$. Markers indicate an interpolation probability $\geq 99\%$, which is everywhere on all curves in this figure.

underlying distribution. Recall from Section 4.2 that, even for the simple Two Moons data set, we needed around 800 000 samples to demonstrate a clear saturation effect. Second, there may be an overlap in the class distributions which violates our separation assumption. This can happen quite easily for real data due to the presence of noise.

4.4 Multi-Class Classification on MNIST

Recall that our method is designed for binary problems. One-versus-many is a common strategy to extend binary classification methods to multi-class problems: for each class, train a binary classifier to distinguish between this class and all other classes. At inference time, query all classifiers and output the class label corresponding to the classifier with the highest confidence score.

We extend Algorithm 2 to multi-class problems in a similar manner. However, as the first layer is obtained in an identical way for every execution of our method, we reuse it across all classes. One can use a simple union bound argument to prove high success probability for this case. Let $K \geq 2$ denote the total number of classes and $\mathcal{X}_k$ the set of samples of class $k \in [K]$. Sample the first layer Φ at random as in Algorithm 2. Then, for each class $k \in [K]$ compute the second and third layer while using Φ as the first layer and $\mathcal{X}^- = \mathcal{X}_k$ and $\mathcal{X}^+ = \bigcup_{\ell \neq k} \mathcal{X}_\ell$ as input data. It is convenient to modify the third layer to map samples of $\mathcal{X}^-$ to 1 and samples of $\mathcal{X}^+$ to 0. Denote the concatenation of the second and third layers by F_k. Define the final classifier $F\colon (\mathcal{X}_1 \cup \cdots \cup \mathcal{X}_K) \to \{0,1\}^K$ by

$$F(\boldsymbol{x}) = (F_1(\Phi(\boldsymbol{x})), \ldots, F_K(\Phi(\boldsymbol{x})))$$

which outputs the class label as a one-hot encoding. We apply this method to MNIST.

Interpolation probability. In Figure 10b we again observe a clear phase transition in the interpolation probability. As in the case of Two Moons, this behaves as predicted by Theorem 11, as for λ larger than the radius of the data, $n \gtrsim \lambda$ is enough to guarantee interpolation with any fixed probability. For $\lambda = 0$ the method not only interpolates but it does so with the narrowest first layer. That this works can be intuitively explained by the angular separation of MNIST. The minimal angle between two samples from MNIST is around 0.17 (in contrast to about $2.44 \cdot 10^{-6}$ for Two Moons). Hence, it is possible to efficiently separate pairs of samples with hyperplanes through the origin.

Width of the second layer $\hat{\Phi}$. Again we are interested in the part of the parameter space where the interpolation probability is close to one. In Figure 10c we observe that, while $\lambda = 0$ seems to be the optimal choice (for the interpolation probability), increasing n may still lead to a reduction of the width of the second layer. Figure 11 reveals that the width does decrease well after interpolation is possible, and in fact, $\lambda \approx 0.5$ yields an even lower value. This might be due to the effect that can be seen in Figure 7, where for $\lambda = 0$ the activation regions of the neurons of the second layer are 'wedges' and become more circular for larger λ, which then might prove beneficial to the width of the second layer. Compared to the binary classification experiments in the previous sections, the width of the second layer is relatively large. For the most part, this is due to the larger number of classes: due to our one-versus-many approach, the width of the second layer scales as $\sum_{i=1}^{K} M_i^-$, where K is the number of classes and M_i^- is the mutual covering number for the one-versus-rest problem for class i. Additionally, MNIST may simply not admit a 'small' mutual covering. Although the concept of mutual covering adapts to the relative positioning of the classes, it is not clear whether a covering with Euclidean balls yields the right complexity measure for image data. It would be an interesting future research direction to adapt our method to a different notion of covering that is more suitable for specific types of data, such as images.

4.5 Multi-Class Classification on CIFAR-10

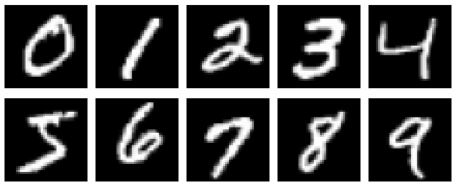
Next, we apply the extension for multi-class problems from the previous section to CIFAR-10. Due to the color channels and a slightly higher resolution, the dimension is larger than that of MNIST. Additionally, photos of real objects provide more variety than handwritten digits.

Interpolation probability. In Figure 12b we see a clear phase transition in the interpolation probability. As in the other experiments, this behaves as predicted by the Theorem 11: for λ larger than the data radius, $n \gtrsim \lambda$ yields interpolation for any fixed probability. As with MNIST in the previous section, $\lambda = 0$ is the best choice.

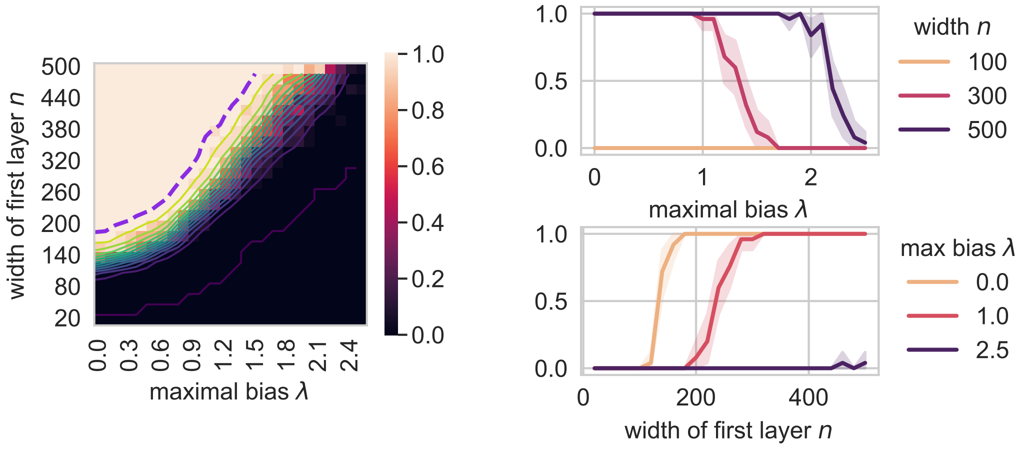
Width of the second layer $\hat{\Phi}$. In Figure 12c we observe that the width of the second layer seems almost constant in the part of the parameter space where the interpolation probability is close to one. Considering even larger values of n in Figure 13, the width of the second layer decreases well beyond the interpolation threshold and the optimal choice of the maximal bias seems to be around $\lambda = 0.25$. Again, relative to the number of samples, the width of the second layer is very large. As in the case of MNIST in the previous section, one might conjecture that the data is either ill-conditioned in terms of the mutual covering or violates one of our assumptions. We will come back to this in Section 5.

4.6 A Worst-Case Example

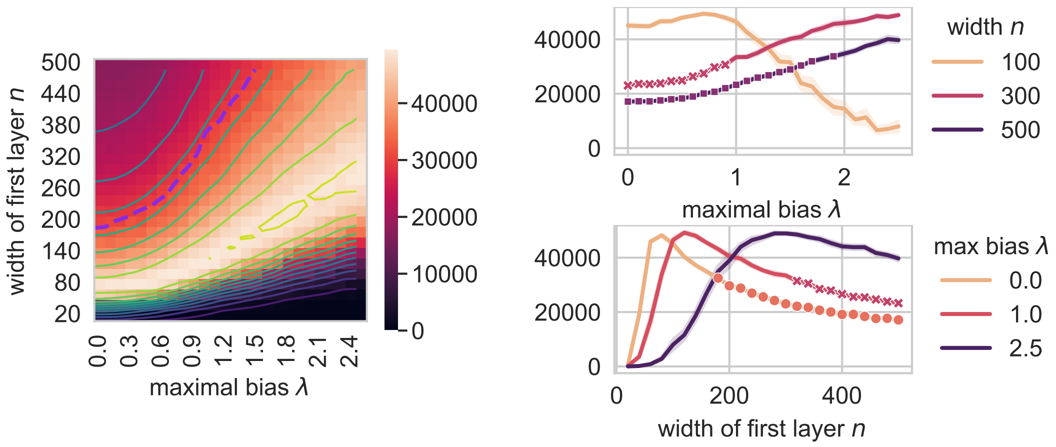
We conclude with a constructed example that demonstrates that our algorithm can in certain cases fail to produce a small interpolating net. Figure 14 shows samples drawn from two



(a) **MNIST.** A multi-class classification data set containing a total of 70.000 grayscale images of handwritten digits. Each image has dimension $d = 28 \times 28 = 784$. We mapped the pixel values from $\{0, \dots, 255\}$ to $[0, 1]$ and normalized the radius to $R = 1$.



(b) **Interpolation probability.** (Left) The interpolation probability (average over 25 runs) as a function of the width of the first layer n and maximal bias λ. The 99% contour line is at the dashed purple line. (Right) Horizontal (top) and vertical (bottom) slices of the heatmap with 95% confidence intervals.



(c) **Width of the second layer $\hat{\Phi}$.** (Left) The width of the second layer $\hat{\Phi}$ (average over 25 runs) as a function of the width of the first layer n and maximal bias λ. The 99% interpolation probability contour line is at the dashed purple line. (Right) Horizontal (top) and vertical (bottom) slices of the heatmap with 95% confidence intervals. Markers indicate an interpolation probability $\geq 99\%$, compare Figure 10b.

Figure 10: Multi-class classification on the MNIST data set.

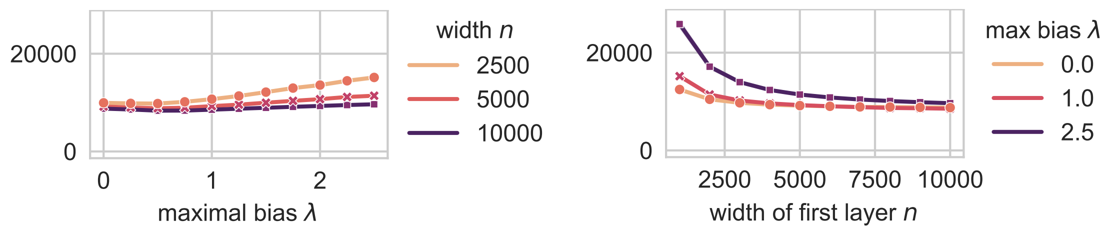
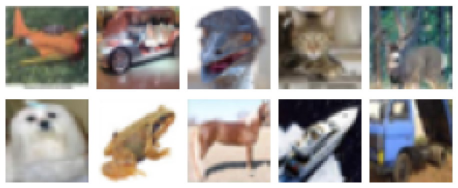


Figure 11: **Width of the second layer $\hat{\Phi}$ on MNIST.** Horizontal (left) and vertical (right) slices of the heatmap in Figure 10c for an extended range of the width n of the first layer. Markers again indicate an interpolation probability $\geq 99\%$, which is everywhere on all curves in this figure.

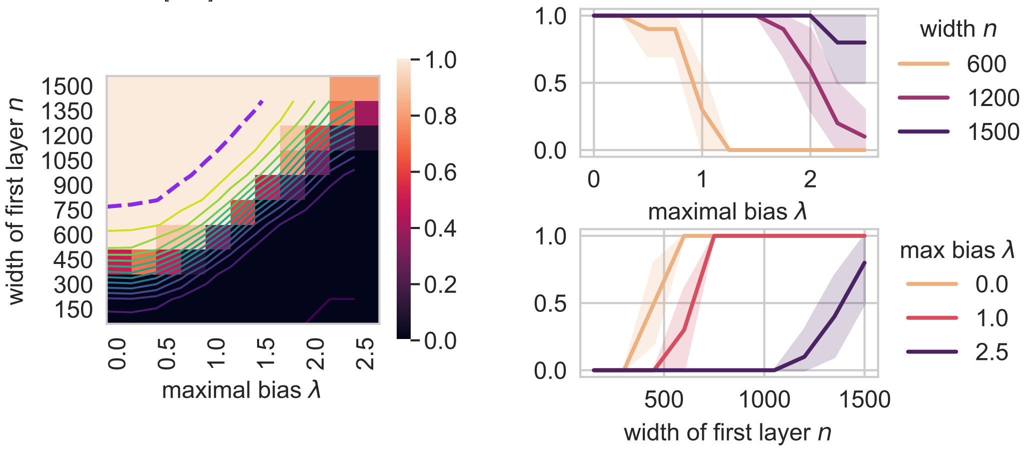
parallel lines, where the distances of samples between classes are smaller than the distances of samples within each class. This forces the components of the mutual covering (and the activation regions of the neurons in the second layer) to be so small that they only cover a single point. Hence, the width of the second layer scales as the number of samples, which is the worst case. This example shows that, although our algorithm is guaranteed to produce small interpolating neural networks on data with a small mutual covering number, it may not take advantage of alternative benign structures (linear separability in this constructed example).

5. Conclusion

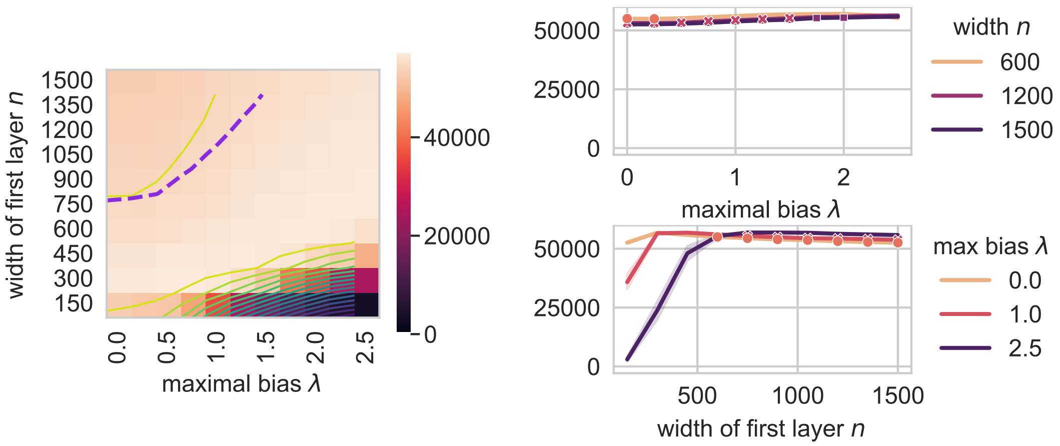
In this paper, we presented an instance-specific viewpoint on the memorization problem for neural networks. We quantified the sufficient network size that guarantees interpolation of given data with two classes in terms of a mutual covering that takes both the geometric complexities and the mutual arrangement of the classes into account. Under our assumptions, the network size depends only on the mutual covering and does not depend on the number of samples in the data set. In this way, our result moves beyond worst-case memorization capacity bounds, which cannot be independent of the number of samples. We gave a constructive proof by presenting a randomized algorithm that is guaranteed to produce an interpolating network for given input data with high probability. We illustrated our theoretical guarantees, in particular the independence of the number of samples, by testing our randomized interpolation algorithm in controlled numerical experiments. In addition, we tested our algorithm on image data and found that it produced relatively large interpolating networks in many cases. In future work, we aim to improve our algorithm for real data by making it robust to noise and by making it tailored to low-complexity structures present in real data such as images.



(a) **CIFAR-10.** The CIFAR-10 data set consists of 60 000 color images in 10 classes of different objects, with 6 000 images per class. Each image has dimension $d = 32 \times 32 \times 3 = 3072$. The pixel values reside in $[0, 1]$ and we normalized the radius to $R = 1$.



(b) **Interpolation probability.** (Left) The interpolation probability (average over 10 runs) as a function of the width of the first layer n and maximal bias λ. The 99% contour line is at the dashed purple line. (Right) Horizontal (top) and vertical (bottom) slices of the heatmap with 95% confidence intervals.



(c) **Width of the second layer $\hat{\Phi}$.** (Left) The width of the second layer $\hat{\Phi}$ (average over 10 runs) as a function of the width of the first layer n and maximal bias λ. The 99% interpolation probability contour line is at the dashed purple line. (Right) Horizontal (top) and vertical (bottom) slices of the heatmap with 95% confidence intervals. Markers indicate an interpolation probability $\geq 99\%$, compare Figure 10b.

Figure 12: Multi-class classification on the CIFAR-10 data set.

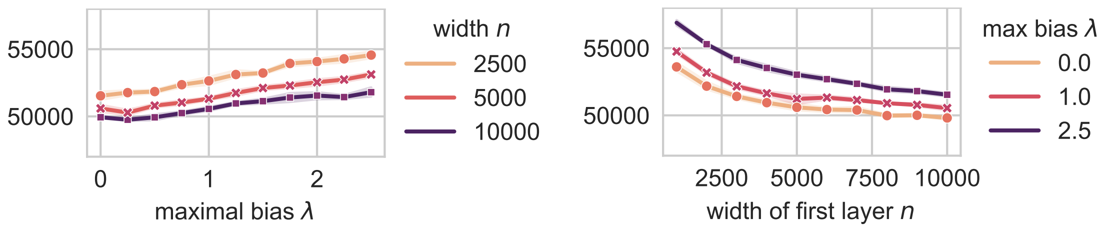


Figure 13: **Width of the second layer $\hat{\Phi}$ on CIFAR-10.** Horizontal (left) and vertical (right) slices of the heatmap in Figure 12c for an extended range of the width n of the first layer. Markers again indicate an interpolation probability $\geq 99\%$, which is everywhere on all curves in this figure.

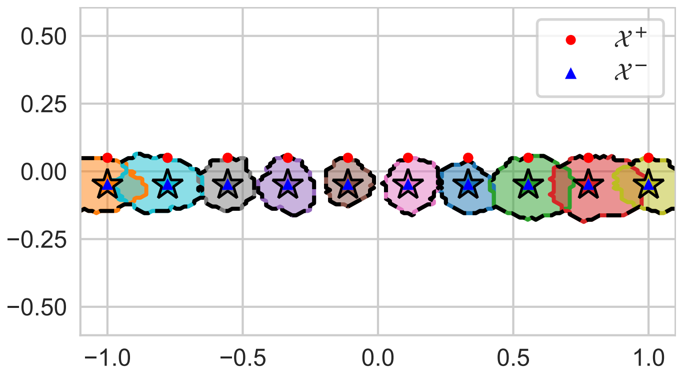


Figure 14: **Parallel lines.** Points are sampled from two parallel lines such that the distance of samples between classes is smaller than the distance of samples within each class. In this case, each neuron in the second layer activates only for its associated point, and hence the second layer has maximal width. Note, as all points of $\mathcal{X}^-$ are accepted into the second layer, they are all marked with stars. Here, we used $n = 2\,000$ and $\lambda = 1.0$.

Acknowledgments

S.D. and M.G. acknowledge support by the DFG Priority Programme DFG-SPP 1798 Grant DI 2120/1-1. The authors thank the anonymous reviewers for their comments and suggestions that lead to improvements in our work.

References

Senjian An, Farid Boussaid, and Mohammed Bennamoun. How can deep rectifier networks achieve linear separability and preserve distances? In *International Conference on Machine Learning*, pages 514–523. PMLR, 2015.

Peter L. Bartlett, Nick Harvey, Christopher Liaw, and Abbas Mehrabian. Nearly-tight VC-dimension and pseudodimension bounds for piecewise linear neural networks. *Journal of Machine Learning Research*, 20(63):1–17, 2019.

Eric Baum and David Haussler. What size net gives valid generalization? In *Advances in Neural Information Processing Systems*, volume 1, pages 81–90. Morgan-Kaufmann, 1988.

Eric B. Baum. On the capabilities of multilayer perceptrons. *Journal of Complexity*, 4(3): 193–215, 1988.

Mikhail Belkin. Fit without fear: remarkable mathematical phenomena of deep learning through the prism of interpolation. *Acta Numerica*, 30:203–248, 2021.

Mikhail Belkin, Daniel Hsu, Siyuan Ma, and Soumik Mandal. Reconciling modern machine-learning practice and the classical bias–variance trade-off. *Proceedings of the National Academy of Sciences*, 116(32):15849–15854, 2019.

Sebastien Bubeck, Ronen Eldan, Yin T. Lee, and Dan Mikulincer. Network size and size of the weights in memorization with two-layers neural networks. In *Advances in Neural Information Processing Systems*, volume 33, pages 4977–4986. Curran Associates, Inc., 2020.

Amit Daniely. Neural networks learning and memorization with (almost) no over-parameterization. In *Advances in Neural Information Processing Systems*, volume 33, pages 9007–9016. Curran Associates, Inc., 2020.

Sjoerd Dirksen and Shahar Mendelson. Non-Gaussian hyperplane tessellations and robust one-bit compressed sensing. *Journal of the European Mathematical Society*, 23(9):2913–2947, 2021.

Sjoerd Dirksen, Martin Genzel, Laurent Jacques, and Alexander Stollenwerk. The separation capacity of random neural networks. *Journal of Machine Learning Research*, 23 (309):1–47, 2022a.

Sjoerd Dirksen, Shahar Mendelson, and Alexander Stollenwerk. Sharp estimates on random hyperplane tessellations. *SIAM Journal on Mathematics of Data Science*, 4(4):1396–1419, 2022b.

Rong Ge, Runzhe Wang, and Haoyu Zhao. Mildly overparametrized neural nets can memorize training data efficiently. Preprint arXiv:1909.11837, 2019.

Promit Ghosal, Srinath Mahankali, and Yihang Sun. Randomly initialized one-layer neural networks make data linearly separable. Preprint arXiv:2205.11716, 2022.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, volume 2. Springer, 2009.

Trevor Hastie, Andrea Montanari, Saharon Rosset, and Ryan J. Tibshirani. Surprises in high-dimensional ridgeless least squares interpolation. *Annals of Statistics*, 50(2):949–986, 2022.

Guang-Bin Huang. Learning capability and storage capacity of two-hidden-layer feedforward networks. *IEEE Transactions on Neural Networks*, 14(2):274–281, 2003.

Shih-Chi Huang and Yih-Fang Huang. Bounds on number of hidden neurons of multilayer perceptrons in classification and recognition. In *1990 IEEE International Symposium on Circuits and Systems*, pages 2500–2503. IEEE, 1990.

Yann LeCun, Corinna Cortes, and Christopher J.C. Burges. The MNIST database of handwritten digits. `http://yann.lecun.com/exdb/mnist/`, 1998.

Sangmin Lee, Abbas Mammadov, and Jong Chul Ye. Defining neural network architecture through polytope structures of datasets. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235, pages 26789–26836. PMLR, 2024.

Liam Madden and Christos Thrampoulidis. Memory capacity of two layer neural networks with smooth activations. *SIAM Journal on Mathematics of Data Science*, 6(3):679–702, 2024.

Song Mei and Andrea Montanari. The generalization error of random features regression: Precise asymptotics and the double descent curve. *Communications on Pure and Applied Mathematics*, 75(4):667–766, 2022.

Preetum Nakkiran, Gal Kaplun, Yamini Bansal, Tristan Yang, Boaz Barak, and Ilya Sutskever. Deep double descent: where bigger models and more data hurt. *Journal of Statistical Mechanics: Theory and Experiment*, 2021(12):124003, 2021.

Sejun Park, Jaeho Lee, Chulhee Yun, and Jinwoo Shin. Provable memorization via deep neural networks using sub-linear parameters. In *Proceedings of Thirty Fourth Conference on Learning Theory*, pages 3627–3661. PMLR, 2021.

Shashank Rajput, Kartik Sreenivasan, Dimitris Papailiopoulos, and Amin Karbasi. An exponential improvement on the memorization capacity of deep threshold networks. In *Advances in Neural Information Processing Systems*, volume 34, pages 12674–12685. Curran Associates, Inc., 2021.

Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, 2014.

Eduardo D. Sontag. Shattering all sets of k points in "general position" requires $(k-1)/2$ parameters. *Neural Computation*, 9(2):337–348, 1997.

Gal Vardi, Gilad Yehudai, and Ohad Shamir. On the optimal memorization power of ReLU neural networks. In *International Conference on Learning Representations*, 2022.

Roman Vershynin. *High-Dimensional Probability: An Introduction with Applications in Data Science*, volume 47. Cambridge University Press, 2018.

Roman Vershynin. Memory capacity of neural networks with threshold and rectified linear unit activations. *SIAM Journal on Mathematics of Data Science*, 2(4):1004–1033, 2020.

Chulhee Yun, Suvrit Sra, and Ali Jadbabaie. Small ReLU networks are powerful memorizers: a tight analysis of memorization capacity. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning (still) requires rethinking generalization. *Communications of the ACM*, 64(3):107–115, 2021a.

Jiawei Zhang, Yushun Zhang, Mingyi Hong, Ruoyu Sun, and Zhi-Quan Luo. When expressivity meets trainability: Fewer than n neurons can work. In *Advances in Neural Information Processing Systems*, volume 34, pages 9167–9180. Curran Associates, Inc., 2021b.