SEPARATION CAPACITY OF RANDOMLY INITIALIZED DNNS

ARIC CUTULI, HAROLD HAODONG MIAO, AND WEITAO ZHU

Abstract. Feedforward neural networks have achieved a stunning record of performing highly complex computational tasks, especially with its recent surge of applications. However, the theoretical understanding of these networks' computational prowess has been lacking, even in the simplest case of classification tasks. A recent result of [GS22] shows through a geometric analysis that with sufficient width, a one-layer randomly initialized neural network can transform two sets into two linearly separable sets with high probability, without necessitating parameter training. Building upon this result, this current project further explores the impact of width and depth on the separation capacity of randomly initialized neural networks. Our experimental analysis shows that for randomly initialized neural networks with standard Gaussian weights and uniform biases symmetric about 0, increasing depth does not necessarily improve linear separability for a given width. We explore the impact that different parameter distributions and activation functions have on a neural network's linear separability, which sheds light onto hypothetical results of a fully trained architecture and onto the utility that rectifier networks have in this particular problem.

1. Introduction

In this project, we focus on the separation capacity of feedforward neural networks, which is the ability of neural networks to cleanly separate datasets into two classes that are separated in space by a margin. We formally define this concept with respect to datasets and neural networks below. We run experiments with networks of varying width, depth, hyperparameters, and activation functions and discuss the implications of those results.

Our experiments are conducted using randomly initialized neural networks (RINNs). Examining the experimental performance of RINNs consists of running a forward pass over many architectures of randomly sampled weight matrices and bias vectors. Effectively, this allows trials to span over the space of architectures, sidestep the added computation cost of fully training the neural network, and avoid potential black box results achieved via backpropagation. Despite much experimental effectiveness of this approach, a theoretical justification for its success in classification tasks has not been extensively explored until recently [GS22, DS21]. This study investigates this justification but focuses primarily on the separation of datasets using RINNs and on providing deeper insights through further experimentation.

1.1. Problem Statement.

Prior to formally discussing the problem, we introduce the following definitions.

Definition 1.1. Two bounded, possibly infinite sets $X^+, X^- \subset \mathbb{R}^d$ are δ-separated if

$$||x^+ - x^-|| \geq \delta \text{ for all } x^+ \in X^+ \text{ and } x^- \in X^-.$$

Definition 1.2. Let $F : \mathbb{R}^d \to \mathbb{R}^n$ denote the function defined by a feedforward neural network. $F(X^+)$ and $F(X^-)$ are *linearly separable* if there exists a and c such that

$$a^T x + c = \begin{cases} > 0 \text{ if } x \in X^+ \\ < 0 \text{ if } x \in X^- \end{cases}$$

with the *margin* of separation defined as $\inf_x\{|a^T x + c|/\|a\|\}$.

Our problem can now be phrased as follows.

Problem 1.3. Given two bounded and δ-separated data sets $X^+, X^- \subset \mathbb{R}^d$, is there a neural network F such that $F(X^+)$ and $F(X^-)$ are linearly separable?

The answer to the above problem is affirmative. Recently, [GS22] has proved the following results, which our experimental work centers around. Define the *mutual complexity* of any two finite sets X^+ and X^- by the total number of points $N = |X^+| + |X^-|$. For two arbitrary sets $X^+, X^- \subset R\mathbb{B}_2^d$, mutual complexity is indexed by parameters (δ, μ) where δ is the separation and μ is the mutual metric entropy of the two sets [Ver18, Section 7]. The following definition is from [GS22, DS21].

Definition 1.4. [GS22, Definition 1.2] For any $n \in \mathbb{N}$, let $[n] := \{1, 2, \ldots, n\}$. Let $X^-, X^+ \subset R\mathbb{B}_2^d$ be two δ-separated sets. Suppose there exist two finite sets of *centers* $C^+ = \{c_i^+\}_{i=1}^{N^+} \subset \mathbb{R}_2^d$ and $C^- = \{c_i^-\}_{i=1}^{N^-} \subset \mathbb{R}^d$, and two sets of constants $r_1^+, r_2^+, \ldots, r_{N^+}^+, r_1^-, r_2^-, \ldots, r_{N^-}^- \geq 0$ of *radii* such that

(1) $X^+ \subset \bigcup_{i=1}^{N^+} X_i^+$ and $X^- \subset \bigcup_{j=1}^{N^-} X_j^-$ where $X_i^+ := X^+ \cap \mathbb{B}_2^d(c_i^+, r_i^+)$ and $X_j^- := X^- \cap \mathbb{B}_2^d(c_j^-, r_j^-)$ satisfies $c_i^+ \in \text{conv}(X_i^+)$ and $c_j^- \in \text{conv}(X_j^-)$ for all $i \in [N^+], j \in [N^-]$.
(2) $r_i^+ \leq \mu^{-1} \operatorname{dist}^2\left(c_i^+, C^-\right)$ for $i \in [N^+]$, and $r_j^- \leq \mu^{-1} \operatorname{dist}^2\left(c_j^-, C^+\right)$ for $j \in [N^-]$,a
(3) C^+ and C^- are δ-separated.

Then, we say X^- and X^+ have (δ, μ)-*mutual complexity* N, where $N = N^- + N^+$.

Their first result states that for any two arbitrary sets satisfying Definition 1.4, we can find a linearly separable neural network.

Theorem 1.5. [GS22, Theorem 2.4] *Let $X^+, X^- \subset R\mathbb{B}_2^d$ be any two δ-separated sets and k as defined as*

$$k := C\left(\frac{32R}{\delta^2}\right)^2 \left(w^2((X^+ - X^-) \cup X^+ \cup X^-) + R^2\right). \tag{1.1}$$

For any parameter $\gamma \in [0, \max \delta^2/8Rd, \delta^2/18R\sqrt{k}]$, let N be the $(\delta, 8R^2/\gamma)$-mutual complexity of X^+ and X^- and let centers C^+ and C^- be as guaranteed by Definition 1.4. Then there exists a neural network Φ such that $\Phi(X^+)$ and $\Phi(X^-)$ are linearly separable with margin at least $\sqrt{N}\mathcal{M}(\gamma/2, N)$ where $\mathcal{M}$ is as in Definition 1.2.

In addition, the previous deterministic result can be further upgraded to a probabilistic statement for randomly initialized neural networks. To state the second result, we define the *Gaussian width*

$w(T)$ of a subset $T \subset \mathbb{R}^n$ ([Ver18, Definition 7.5.1]) as

$$w(T) := \mathbb{E}\big[\sup_{x\in T} g^T x\big], \quad \text{where } g \sim \mathrm{N}(0, I_{n\times n}). \tag{1.2}$$

Theorem 1.6. [GS22, Theorem 1.4] *Let $X^+, X^- \subset R\mathbb{B}_2^d$ be δ-separated sets with $d \geq 2$. There exists a constant $C > 0$ depending only on δ and R such that the following holds. Define the* mutual complexity parameter $N := |X^+| + |X^-|$ *and the* separation parameter $\gamma := \delta^2/18R\sqrt{k}$ *if X^+ and X^- are finite where*

$$k := C\left(\frac{32R}{\delta^2}\right)^2 \left(w^2((X^+ - X^-) \cup X^+ \cup X^-) + R^2\right). \tag{1.3}$$

Otherwise N is the $(\delta, 144\delta^{-2}R^3\sqrt{k})$-mutual complexity and $\gamma := \delta^2/36R\sqrt{k}$.

Consider a one-layer ReLU*-activated RINN $\Phi(x) = \mathrm{ReLU}(Wx + b)$ with standard Gaussian weights $W \sim \mathrm{N}(0, I_{n\times d})$ and bias $b \sim \mathrm{Unif}\left([-\lambda, \lambda]^n\right)$ for maximal bias $\lambda \geq 9R\sqrt{k}/8$. Then, for any $\eta \in (0, 1)$, the sets $\Phi(X^+), \Phi(X^-) \subset \mathbb{R}^n$ are linearly separable with probability at least $1 - \eta$ by a margin at least $\mathcal{M}(\gamma, N)$ defined in Definition 1.2 provided that width n satisfies the following condition*

$$n \geq \frac{\log(N/\eta)}{q}, \quad \textit{where} \quad q := \frac{R}{4\lambda\sqrt{k}}\left(\frac{2\delta^2}{81R^2}\right)^k. \tag{1.4}$$

1.2. **Expressivity of Neural Networks.** Expressivity in neural networks refers to the ability of a neural network to approximate a wide variety of functions or mappings between input and output data. Specifically, a neural network is said to be expressive if it can represent a broad class of nonlinear functions or decision boundaries that can fit the training data accurately.

Expressivity is an important property of neural networks, as it determines the complexity and diversity of the functions that the network can learn to represent. Highly expressive neural networks can learn to model complex, high-dimensional data such as images, audio, and text, while less expressive networks may struggle to capture the underlying patterns in the data.

The expressivity of a neural network is determined by its architecture, which includes the number of layers, the number of neurons per layer, the activation functions used, and the connections between neurons. In general, larger and deeper neural networks with more complex activation functions and more connections are more expressive than smaller and shallower networks [GC16, Chapter 6].

Our report is related to this property of neural networks, as finding an architecture that is linearly separable boils down to finding a parameterization that approximates a hyperplane separating the data into the two classes.

1.3. **Overview.** Here we give a brief overview of the rest of the paper. In Section 2, we introduce a number of notations and break down the mathematics behind the results in Theorems 2.5 and 1.6. With these results as our theoretical underpinning, we include our experimental analysis in Section 3. The subsections there contain the outcomes of our tests on the impact of depth, width, bias, activation functions and random initialization. Finally, we conclude our report in section 4 and expand on directions for further experiments.

2. The geometry of linear separation

In this section, we explain the mathematics behind the linear separability of one-layer neural networks in [GS22] and break down the proofs of Theorems 2.5 and 1.6.

2.1. General Setup for Linear Separation with ReLU Activation Function.

2.1.1. *ReLU Activation Function.* The popular ReLU activation function from $\mathbb{R}$ to $\mathbb{R}$ for a single dimension is defined as $\mathrm{ReLU(x)} = \mathrm{x}\mathbb{1}_{x>0}$, and we employ the multiple dimension ReLU activation function from $\mathbb{R}^n$ to $\mathbb{R}^n$ by applying ReLU dimension wise on the input n dimension data.

2.1.2. *Neural Network as Function.* We refer to a neural network by the function it represents. In our discussion, a one-layer neural network with ReLU activation function is defined as $\Phi(x) = ReLU(Wx+b)$, where $\Phi : \mathbb{R}^d \to \mathbb{R}^n$, $W \in \mathbb{R}^{n\times d}$ is the weight matrix, and $b \in \mathbb{R}^n$ is the bias. A two-layer neural network with ReLU activation function is defined as $\tilde{\Phi}(\Phi(x)) = ReLU(W(ReLU(W^*+b^*))+b)$. Here W and W^* and similarly b and b^* are assumed to be independent. However, in Section 3.3, we set them to be the same to compare the impact of different weight matrices and biases on the linear separability of one-layer neural networks.

2.2. Key Geometric Insight Behind ReLU Activation Function.

One important ingredient of the proof is the geometric property of the ReLU function, which selectively preserves the Euclidean distance encoded by the input depending on its signage. Recall that the i-th node of a neural network Φ encodes the i-th row W_i of the weight matrix W and the i-th entry b_i of b, i.e., $(\Phi(x))_i = \mathrm{ReLU}(W_i^T x + b_i)$. For any $x \in \mathbb{R}^n$, assuming that $W_i \in S^{d-1}$ is normalized, $h_i(x) = W_i^T x + b_i$ equals the signed distance between x and the hyperplane $h_i = 0$. Thus, $(\Phi(x))_i$ measures the distance between x and h_i if x is on the positive side of h_i and is zero otherwise. In our proof, this property of the ReLU function helps us differentiate data points with different signs.

More specifically, we denote $X+$ and $X-$ as the two sets of points with different labels. Under this setup, by Definition 1.2, to show that the neural network Φ linearly separates the points in $X+$ and $X-$, it suffices to find a vector (or matrix) $\vec{a}$ such that $\vec{a}^T\Phi(x+) > 0$ and $\vec{a}^T\Phi(x-) < 0$ for every $x+ \in X+$ and every $x- \in X-$. The key idea is to understand which of the coordinates of $\Phi(x+)$ for $\mathrm{x+} \in \mathrm{X+}$ are strictly positive, and which of the coordinates of $\Phi(x-)$ for $\mathrm{x-} \in \mathrm{X-}$ are strictly negative and to select the weight matrix $\vec{a}$ accordingly to achieve this property. In fact, it is sufficient to construct a weight vector such that a subset of the coordinates are linearly separable, which is encoded in the following lemma.

Lemma 2.1. [GS22] *For $S^+, S^- \subset \mathbb{R}^k$ and any projection π onto a subspace indexed by coordinates $\Sigma \subset \{1, \dots, k\}$, if $\pi(S^+)$ and $\pi(S^-)$ are linearly separable with margin μ, then so are S^+ and S^-.*

To build up the proof for linear separability for sets with arbitrary mutual complexity, we start from the case of finite sets.

2.3. A One-Layer Deterministic neural network Separates Finite Sets.

We first give the deterministic NN analogs of the finite set cases of Theorem 1.6 with an algorithm, which constructs the one-layer deterministic neural networks through separating points x_i one by one in descending order of norm with hyperplanes h_i. For each point x_i, we select a hyperplane h_i such that x_i and

all points in the other class set with smaller norms are on different sides of h_i each with distance at least γ. Then, when we ReLU-activate the one-layer neural network Φ with h_i as the i-th node, $(\Phi(x_i))_i = h_i(x_i) > 0$ and for any point x in the other set with smaller magnitude, $(\Phi(x))_i = 0$ as $h_i(x) \leq 0$. More specially, the algorithm is as follows. Recall k from (1.3) and $N := |X^+| + |X^-|$ and define *sign* $\sigma : X^+ \cup X^- \to \{+1, -1\}$ by

$$\sigma(x) = +1 \quad \text{if} \quad x \in X^+, \quad \text{and} \quad \sigma(x) = -1 \quad \text{if} \quad x \in X^-. \tag{2.1}$$

Algorithm 1 To construct a one-layer neural network separating two finite sets with a margin [GS22, Algorithm 1]

Input: Two δ-separated finite sets $X^+, X^- \subset R\mathbb{B}_2^d$ and $\gamma \in [0, \max(\delta^2/8Rd, \delta^2/18R\sqrt{k})]$.

Output: A one-layer neural network Φ such that $\Phi(X^+)$ and $\Phi(X^-)$ are linearly separable.

1: **for** $i = 1, 2, \ldots, N$ **do**
2: $x_i \leftarrow x$ where x has the i-th largest norm in $X^+ \cup X^-$.
3: **end for**
4: **for** $i = 1, 2, \ldots, N$ **do**
5: Find $w \in \mathbb{S}^{d-1}$ and $b \in \mathbb{R}$ such that $w^T x_i + b \geq \gamma$ and $w^T x_j + b \leq -\gamma$ for all j with $j > i$ and $\sigma(x_i) \neq \sigma(x_j)$. If no such j exists, ask $w^T x_i + b \geq \gamma$ and $w^T x + b \leq 0$ for some $x \in R\mathbb{B}_2^d$.
6: $w_i \leftarrow w$, $b_i \leftarrow b$, $h_i \leftarrow$ the function $w_i^T x + b_i$
7: **end for**
8: $W \leftarrow$ the matrix with row vectors w_i^T
9: $b \leftarrow$ the vector with components b_i
10: **return** the function $\Phi(x) = \mathrm{ReLU}(Wx + b)$

In particular, the existence of h_i in line 6 of the algorithm is established via a probabilistic argument in Theorem 2.2 in [GS22] for γ satisfying the requirement of the input.

Theorem 2.2. [GS22, Theorem 2.2] *For any two δ-separated finite sets $X^+, X^- \subset R\mathbb{B}_2^d$, define x_i and σ as in Algorithm 1 and (2.1). Consider the random affine function $h(z) = v^T z + t$ where $t \in \mathbb{R}, v \in \mathbb{R}^d$ are independent random variables, $t \sim \mathrm{Unif}([-\lambda, \lambda])$, and the distribution of v is specified below. For any $\gamma > 0$, let*

$$B_i := \{h(x_i) \geq \gamma\} \cap \bigcap_{j:j>i, \sigma(x_i) \neq \sigma(x_j)} \{h(x_j) \leq -\gamma\}.$$

for each $i \in \{1, 2, \ldots, N\}$. Let $p := \frac{R}{8(d-1)\lambda}(\frac{\delta}{8R^2})^d$ and recall q from (1.4). There exists a constant C depend only on δ and R in the definition (1.3) of k such that the following statements hold.

(1) For all i and $\gamma \leq \delta^2/8Rd$, $\mathbb{P}(B_i) \geq p$ if $v \sim \mathrm{Unif}\left(\mathbb{S}^{d-1}\right)$ and $\lambda \geq R$.
(2) For all i and $\gamma \leq \delta^2/8Rd$, $\mathbb{P}(B_i) \geq p/10$ if $v \sim \mathrm{N}(0, I_d)$ and $\lambda \geq 3R\sqrt{d}$.
(3) For all i and $\gamma \leq \delta^2/18R\sqrt{k}$, $\mathbb{P}(B_i) \geq q$ if $v \sim \mathrm{N}(0, I_d)$ and $\lambda \geq 9R\sqrt{k}/8$.

The details of the proof of this theorem can be found in the Appendix of [GS22]. For now, assuming its validity and thereby the existence of the hyperplanes h_i, we apply the ReLU activation

function to the hyperplanes with our neural network. The outcome is that

$$(\Phi(x_i))_j \begin{cases} = \mathrm{ReLU}(h_j(x_i)) \geq \gamma \text{ if } x \in X^+ \\ = 0 \text{ if } \sigma(x_j) \neq \sigma(x_i), j < i \\ \geq 0 \text{ otherwise.} \end{cases}$$

This property allows us to find a deterministic hyperplane $H(z) := \vec{a}^T(z) = \sum_{i=1}^N a_i z_i$ that separates $\Phi(X^+)$ and $\Phi(X^-)$ with desired margins by choosing $a_i := \sigma(x_i)\left(1 + \left|\sum_{j=i+1}^N a_j(\Phi(x_i))_j\right|\right)/\gamma$ recursively for $i = N, \ldots, 1$ (Algorithm 2 in [GS22]). This choice of $\vec{a}$ gives us that

$$\sigma(x_i)H(\Phi(x_i)) \geq 1$$

for all i after splitting $H(\Phi(x_i))$ into 4 different sums involving x_i, $j < i, \sigma(x_i)\neg\sigma(x_i)$ and $j < i, \sigma(x_i) = \sigma(x_i)$ and $j > i$ and using triangle inequality and leads to the following theorem.

Theorem 2.3. [GS22, Theorem 2.3] *For output Φ of Algorithm 1 on any sets X^+, X^- and parameter γ satisfying the input conditions, $\Phi(X^+)$ and $\Phi(X^-)$ are linearly separable with margin at least $\sqrt{N}\mathcal{M}(\gamma, N)$ where $\mathcal{M}(\gamma, N) := \sqrt{\frac{4R(R+\lambda)}{N(1+2R/\lambda)^{2N}-N}}$.*

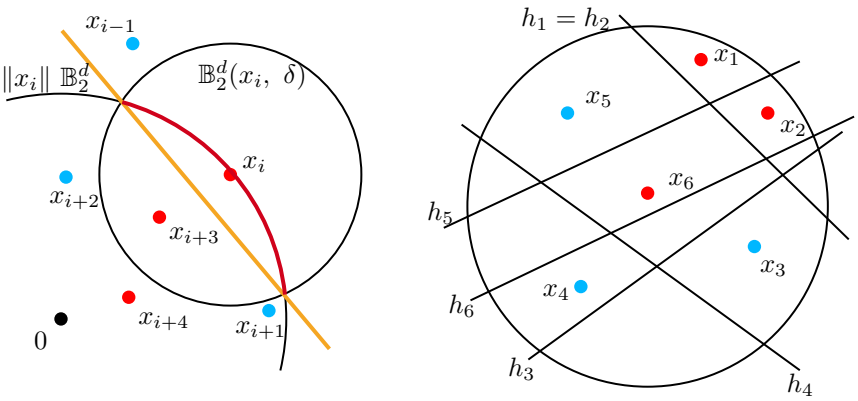


FIGURE 1. [GS22, Figure 1] **Illustration of Algorithm 1 and Theorem 2.3.** Given input sets X^+ and X^-, let x_i be the point with the i-th largest norm. The left picture illustrates how Algorithm 1 picks each h_i. See the proof sketch of Theorem 2.3. The right picture is a concrete example of this on a input with 6 points.

Remark 2.4. The above proof also suggests that other activation functions with similar properties, i.e. distinguishing between signage and preserving Euclidean distance for positive distance, could potentially achieve similar separability. We examined this idea experimentally in Section 3.5.

2.4. **Generalizations to Arbitrary Sets and Random Initialization.** Next up, we generalize Theorem 2.3 to arbitrary sets X^+ and X^- which may not be finite. By carefully varying margin parameter γ and mutual complexity parameters, the next theorem shows that the output of Algorithm 1 separates the entire sets X^+ and X^- if it separates the centers in Definition 1.4.

Theorem 2.5. [GS22, Theorem 2.4] *Let $X^+, X^- \subset R\mathbb{B}_2^d$ be any two δ-separated sets and k as defined in (1.3). For any parameter $\gamma \in [0, \max(\delta^2/8Rd, \delta^2/18R\sqrt{k})]$, let N be the $(\delta, 8R^2/\gamma)$-mutual complexity of X^+ and X^- and let centers C^+ and C^- be as guaranteed by Definition 1.4. Then*

(1) The centers C^+ and C^- with parameter γ satisfy the input conditions of Algorithm 1.
(2) Let Φ be the output of Algorithm 1 on C^+ and C^- with parameter γ. Then, $\Phi(X^+)$ and $\Phi(X^-)$ are linearly separable with margin at least $\sqrt{N}\mathcal{M}(\gamma/2, N)$ where $\mathcal{M}$ is as in (??).

Proof of the above theorem follows from the proof of Theorem 2.3 applying to the centers C^+ and C^- and Definition 1.4, which controls the closeness of the clusters of points to the centers. The details of the proof can be found in the Appendix of [GS22] and this result sets us up for the main result in Theorem 1.6, which generalizes Theorem 2.3 to one-layer RINN Φ.

Invoking Lemma 2.1, for Theorem 1.6, it suffices to show that with sufficient width, for every i, there exists a node in Φ that satisfies the conditions for h_i in Algorithm 1 with high probability. Collectively, by Lemma 2.1, these nodes make X^+ and X^- separable, which will be enough for Φ to also make them separable. Crucially, this means that

(1) we need to find a random node that satisfies the condition for each h_i with high probability, which is precisely Theorem 2.2;
(2) the remaining hyperplanes do not hurt separation, which is the content of Lemma 2.1.

Based on the above analysis, we sketch our main result (Theorem 1.6) that a one-layer RINN separates sets with high probability.

Proof Sketch of Theorem 1.6. We first prove the theorem for finite X^+ and X^- before generalizing to arbitrary sets.

In the finite X^+, X^- case, let B_i^ℓ, $1 \le i \le N$, $1 \le \ell \le n$ denote the event that the ℓ-th random hyperplane H_ℓ of Φ satisfies the condition for h_i on line 5 of Algorithm 1.

Claim: *$\Phi(X^+)$ and $\Phi(X^-)$ are separable with the desired margin if, for every i, B_i^ℓ holds for some ℓ.*

Assuming the claim, we can find some hyperplanes in Φ that form an output of Algorithm 1 on X^+, X^-, and γ and show that these hyperplanes make X^+ and X^- separable via a similar argument as Theorem 2.3. By Lemma 2.1, Φ makes X^+ and X^- separable with the desired margin, thus proving the claim. The case that the same H_ℓ may satisfy the conditions of multiple h_i is addressed in the Appendix of [GS22], which shrinks the separation margin by a factor of $\sqrt{N}$ compared to Theorems 2.3 and 2.5. The remaining task is to bound $\mathbb{P}((\forall i)(\exists \ell)B_i^\ell)$. By Theorem 2.2, for all i and ℓ, $\mathbb{P}(B_i^\ell) \ge p/10$ or q depending on the γ when j exists on line 5 of Algorithm 1. The case when no such j exists is addressed in the Appendix of [GS22]. By a union bound over i and independence of ℓ

$$\mathbb{P}(\Phi(X^+), \Phi(X^-) \text{ are separable}) \ge \mathbb{P}\left((\forall i)(\exists \ell)B_i^\ell\right) \ge 1 - \sum_{i=1}^{N} \mathbb{P}\left((\forall \ell)(B_i^\ell)^C\right) \ge 1 - N(1-q)^n$$

which is at least $1 - \eta$ if $n \ge \log(N/\eta)/q$.

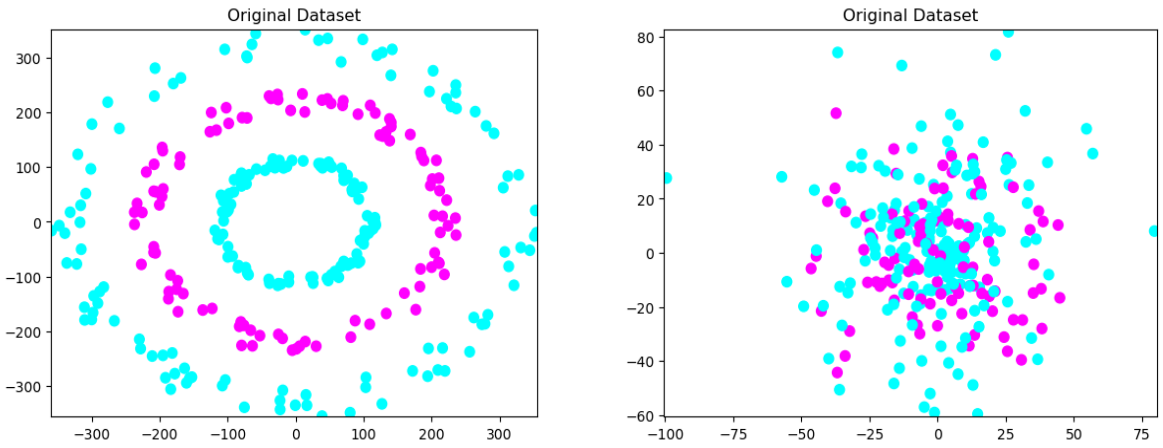
For arbitrary sets X^+ and X^-, let B_i^ℓ be the same event based on the centers C^+ and C^- as guaranteed by Definition 1.4. Then, the argument above combined with Theorem 2.5 follows similarly. □

3. Experimental results

We have reproduced the experimental results from [GS22] , and further extended the experiment in the setting of deeper networks, wider networks, networks with changing biases, networks with skewed supports, and networks with different activation functions. In general, we conclude that for a simpler synthetic data set like ours, the randomly-initialised one-layer neural network performs the best in terms of high separation percentage, but with skewed supports for the bias vector, deeper networks tend to outperform shallower networks. However, as the complexity of the data sets increases, the picture could look dramatically different.

3.1. **Experimental Setup.** Unless stated otherwise, our experimental results are obtained by running many (1000+) trials of randomly initialized neural networks and testing for metrics of accuracy and separability without training the networks via backpropagation. Unless stated otherwise, for each layer, the weight matrices are drawn from a standard Gaussian distribution, and the bias vectors are drawn from a uniform distribution on $[-\lambda, \lambda]$, where λ is to be specified. The data sets used were synthetic separated concentric circular rings with alternating labels. The exact location of the data points were disturbed from the ring by random Gaussian noise.

For the low-dimensional case, we consider concentric circular rings in $\mathbb{R}^2$, which is visualized below. For the high-dimensional case, we used concentric circular rings in $\mathbb{R}^{100}$, which is visualized here in $\mathbb{R}^2$ using t-distributed stochastic neighbor embedding (t-SNE).



(A) Concentric circles in $\mathbb{R}^2$ (B) Concentric circles in $\mathbb{R}^{100}$

Figure 2. Visualization of datasets

During the experiment, we use a Support Vector Machine (SVM) to test the linear separability of the neural network classifier. After 100,000 iterations of the SVM, the percentage of points correctly classified by the Support Vector Machine is referred to as the accuracy of a trial. For each experiment, we run 1000 trials of architectures of newly sampled parameters and define the

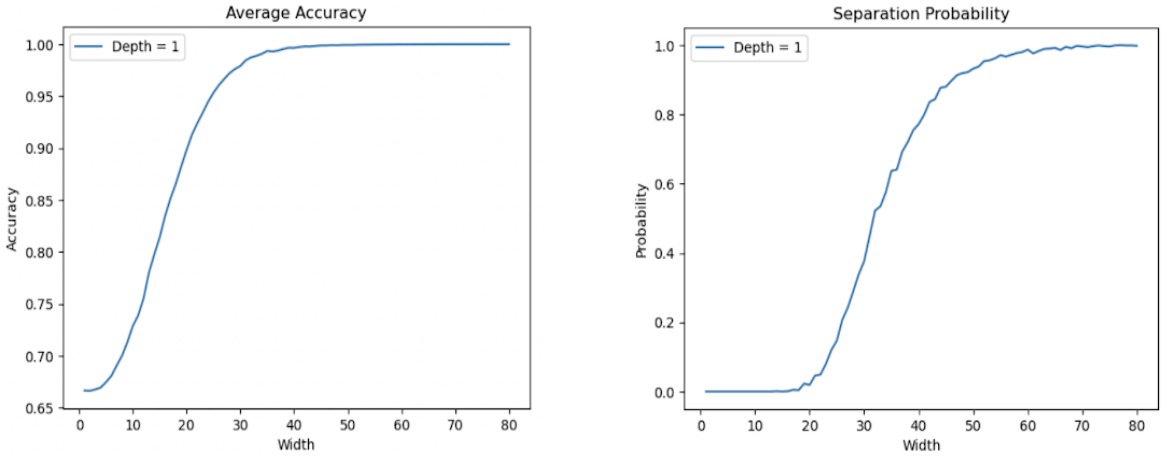


FIGURE 3. Performance of shallow RINN on low-dimensional data

mean as the average accuracy for a given combination of depth and width. Since out initial dataset contains 2/3 of its points in one label and the remaining 1/3 points in the alternative label, the average accuracy of every experiment has a lower bound of 2/3 by construction. The other metric, separation probability, represents the percentage of trials in which the SVM achieves 100 percent accuracy, effectively measuring the probability of a network being fully linearly separable. Unlike average accuracy, separation probability is not lower bounded by any fixed level above zero.

3.2. **Testing the Impact of Depth for Networks of Varying Width.** The experimentation of the source papers test the linear separability for networks with 1 or 2 hidden layers. In the case of low-dimensional data, we extend this experimentation to a 3rd hidden layer, while computational bottlenecks inhibited our analysis of deeper networks when applying them to the high-dimensional data. We see that 3-layer neural networks generally under perform compared to 1-layer and 2-layer networks. We also observe that despite the average accuracy of 1-layer network dominating that of 2-layer networks across all choices of width, the separation probability of a 2-layer neural network is higher than that of a 1-layer neural network for widths of at least 33 neurons per layer. We propose that this is due to the low-dimensional structure of the original data set. For a 1 dimensional neural net with width d, the data is being mapped from $\mathbb{R}^2$ to $\mathbb{R}^d$, while for a neural net with the same width d and 1 hidden layer, the data is being mapped from $\mathbb{R}^2$ to $\mathbb{R}^d$ to $\mathbb{R}^d$. During the last mapping step, the data is from $\mathbb{R}^d$ to $\mathbb{R}^d$, and hence when the dimension d is considerably large compared to 2, the input in the last mapping is sufficiently sparse, hence there is a higher probability of total separation, as compared to from 2 dimension to d dimension in the 1-layer case. This scenario is not observed in other cases, and is up to further investigation. These results are counterintuitive, and in a later section, we see that changing the support for the distribution from which the bias vectors are sampled from leads to different results.

For the high-dimensional dataset, similar results are observed in average accuracy. However, the separation probability of a 1-layer network dominates the separation probability of 2-layer network across all tested width. This observation aligns with our initial proposed explanation in the low-dimensional case – since the original data set is 100 dimension, the mapping in the 1-layer

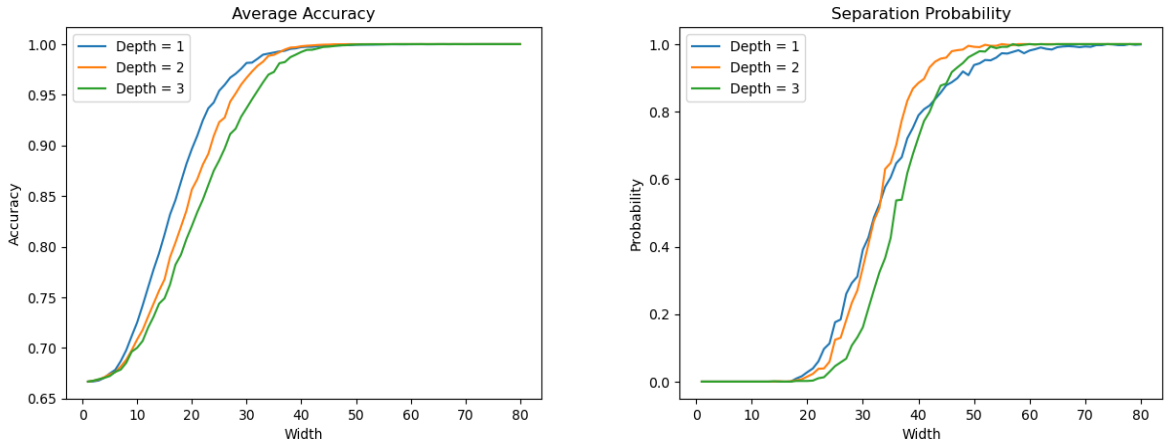


FIGURE 4. Performance of RINNs of varying depth on low-dimensional data

network already has a sufficiently sparse input, and a large depth no longer provides improvement in separation probability through increasing the sparsity of input data in the last mapping of the neural net.

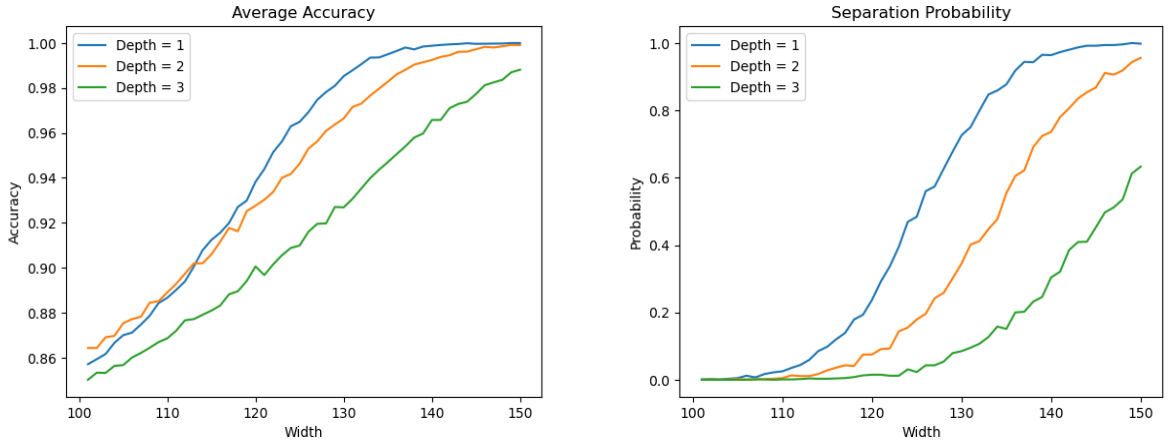


FIGURE 5. Performance of RINNs of varying depth on high-dimensional data

3.3. **Testing the Impact of Bias Across Different Depth of Neural Network.** We further investigate the impact of maximum bias on the separation probability of the randomly initialized neural networks.

As we discussed earlier, the bias vectors are drawn from a uniform distribution on $[-\lambda, \lambda]$, and we would like to investigate the impact of choice of λ on the linear separability of the network. For networks with hidden layers, we further examine the impact of changing the maximum bias λ across different layers based on the proposed formula

$$\lambda_n = \sqrt{\lambda_{n-1} + R^2/3}$$

which is derived from the estimate

$$\|\Phi^n(x)\|_2^2 \approx \|\Phi^{n-1}(x)\|_2^2 + \lambda^2/3$$

featured in [DS21, Theorem 19], where $\Phi^n(x) = (\Phi_n \circ \Phi_{n-1} \circ \ldots \circ \Phi_1)(x)$ and $\Phi^0(x) = x$. We accept this estimate without outlining the full proof. The essence of the proof involves computing the expected value of $\langle \Phi(x^+), \Phi(x^-) \rangle$ for arbitrary points x^+, x^- in the ball of radius R and showing that $\langle \Phi(x^+), \Phi(x^-) \rangle$ is uniformly concentrated about its expected value.

In the performance plots, we use the term 'suboptimal bias' to indicate that the maximal bias of deeper layers are equal to that of the first layer, whereas plots not labeled with this term feature maximal biases that follow the aforementioned rule for each layer. As we can see from the graph, for fixed maximum bias across layers, the performance generally grows as the maximum bias grows, peaking at around 350 for low-dimension and 500 for high-dimension before dropping for larger maximal biases. We conclude that for small maximal bias, increasing λ improves linear separability.

However, this is not the case for deeper networks. As we inspect the formula more closely,

$$\lambda_n = \sqrt{\lambda_{n-1} + R^2/3}$$

we observe that the maximal bias in deeper layers is lower-bounded by R, which is a fixed constant close to 359 determined by the synthesis of the dataset. With this lower-bound, we do not observe an improvement in linear separability through increasing the maximal bias. However, as the maximal bias grows larger and larger, we do observe that the performances of multi-layer networks with fixed bias and varied bias converge. We also continue to observe that 1-layer networks outperform 2-layer networks, and both of them outperform 3-layer networks.

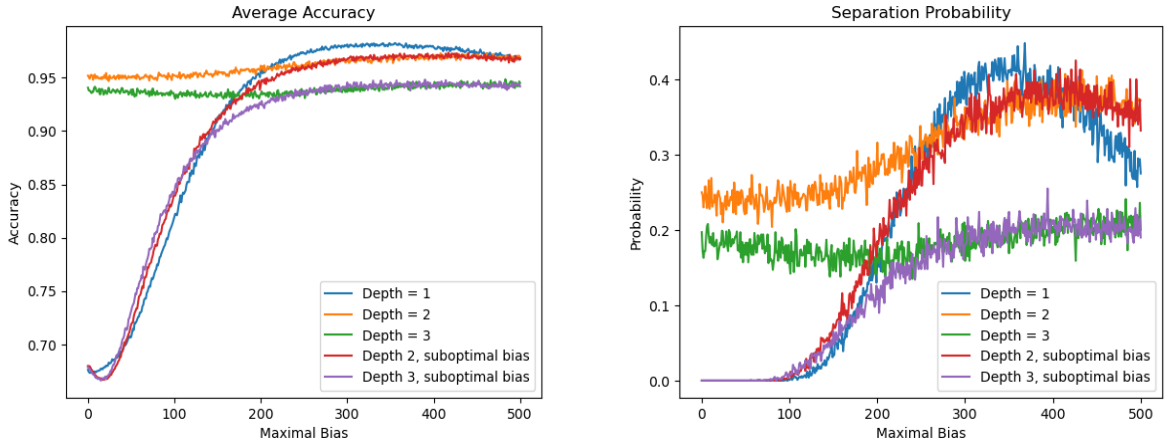


FIGURE 6. Performance of RINNs of varying depth and maximal bias on low-dimensional data

3.4. **Testing Skewed Supports for Bias Initialization.** The theoretical results of our source papers rely on the bias vector being sampled from a uniform distribution symmetric about 0. That is, in our formulation of the problem, we sample many neural networks with standard Gaussian weights and bias vectors that are uniform on $[-\lambda, \lambda]$, and the results we report are thus the average results across many models that are untrained. In this section of our report, we ask whether training the network would produce bias vectors that agree with this prior distribution for bias.

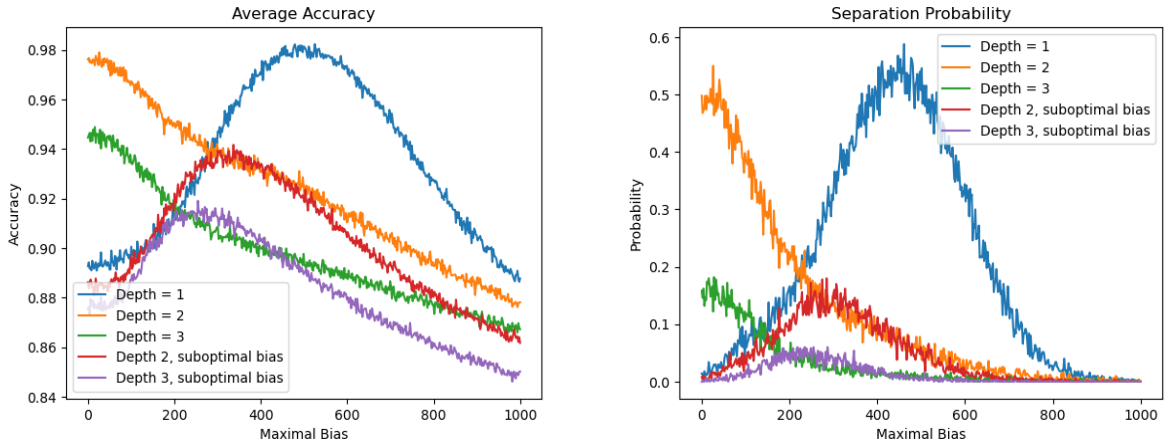


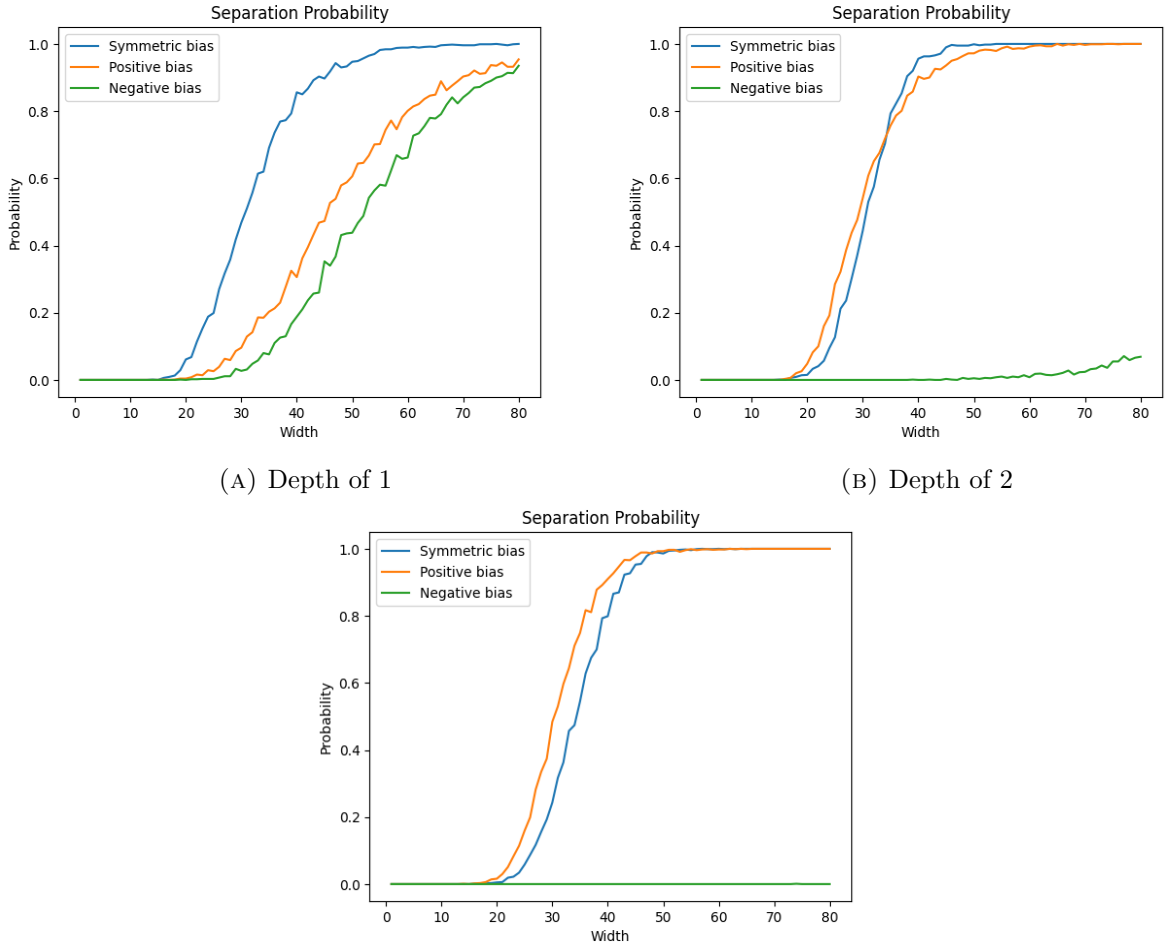
FIGURE 7. Performance of RINNs of varying depth and maximal bias on high-dimensional data

To explore this question, we test different prior distributions for bias and examine the resulting linear separability of those networks. Namely, we examine networks whose bias vectors who are sampled from a positively skewed uniform distribution on $[0, 2\lambda]$ and networks whose bias vectors are sampled from a negatively skewed uniform distribution on $[-2\lambda, 0]$. For each case, we maintain the sampling of weight matrices that come from a standard Gaussian distribution.

We find that when bias is strictly positive, deeper networks separate low-dimensional data with higher probability. This suggests a potential reasoning for the counterintuitive results presented earlier, that increasing depth does not necessarily increase separability when the bias vector is sampled uniformly on $[-\lambda, \lambda]$. Moreover, the plots reveal that when bias is strictly negative, linear separability is severly diminished in deeper networks. In essence, when we sample many architectures whose bias distribution is symmetric about 0, the trials whose biases are predominantly negative leads to average accuracy results that are subpar in deeper networks. The experimental result that positive bias improves average accuracy and separation probability suggests that fully training the network rather than testing untrained networks would tune bias in a positive direction. Further study can be conducted to examine whether this trend continues in even deeper networks and for high-dimensional data.

3.5. **Testing Different Activation Functions.** We also explored the effect different activation functions have on the linear separability of randomly initialized neural networks.

Recall that $h_i(x) = W_i^T x + b_i$ measures the signed distance between x and the hyperplane $h_i = 0$, so a ReLU activation applied to $h_i(x)$ measures the distance from x to h_i when x is on the positive side of h_i and 0 otherwise. That is, the motivation for ReLU networks in the source material is that a neuron is activated proportionally to its signed distance from the hyperplane that separates the data. So, a neuron is active if an input is positive and is inactive otherwise. Effectively, this allows the network to preserve the Euclidean distances captured by the inputs when a point is on the positive side of the hyperplane.



(A) Depth of 1

(B) Depth of 2

(C) Depth of 3

FIGURE 8. Separation probability of RINNs of varying depth and bias support on low-dimensional data. We fix the choice of λ and vary the width of each network. We clearly see that in deeper networks, sampling bias vectors from a uniform distribution of $[0, 2\lambda]$ leads to improved results.

For the different activation functions we experiment with, we consider the performance of a leaky ReLU function $f(h_i(x), \alpha) = \max\{h_i(x), \alpha h_i(x)\}$ with varying leak rates α, a step function $f(h_i(x) = \mathbb{1}_{h_i(x)>0}$, and a linear activation $f(h_i(x)) = h_i(x)$.

Through the experiments, we see that changing the activation function can distort the preserved distance property that the ReLU activation offers and in turn diminish or tarnish the linear separability of the network.

Expectedly, a leaky ReLU activation becomes less effective as it becomes more linear, but it interestingly remains competitive with the ReLU activation even with a leak rate of 0.7. When the activation is completely linear, however, accuracy and separation probability rests at the lower bound regardless of the width imposed on the network. That is, a linear activation leads to the network classifying all points as coming from one group. Although this function preserves

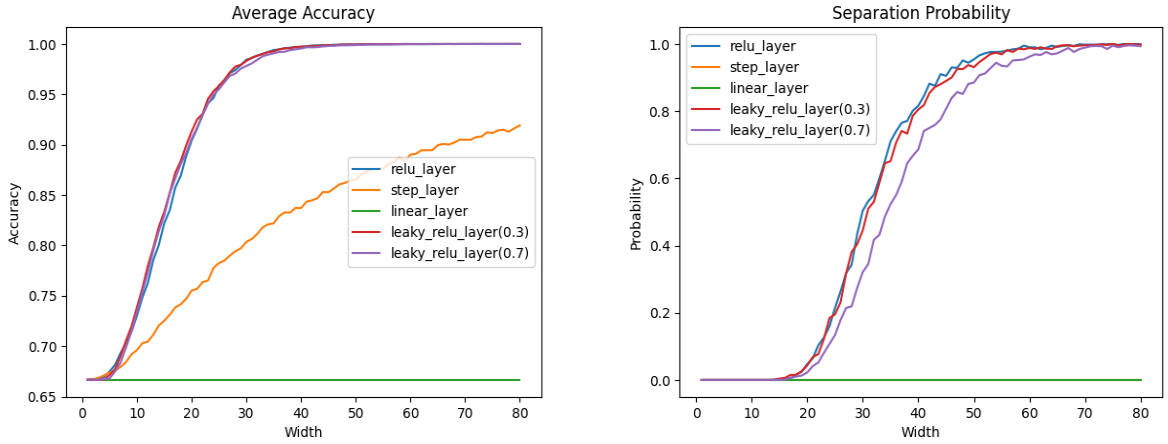


FIGURE 9. Performance of RINNs with varying activation functions. We fix the depth to 1 and vary the width of the network. Bias vectors are sampled from a uniform distribution symmetric about 0 and we fix the choice of λ.

signed distance, it introduces no non-linearity to indicate a point being on the negative side of the hyperplane.

And lastly, the step function is unable to separate the datasets with 100% probability regardless of the width imposed, although it achieves accuracy measurements that mark an improvement over the linear activation. Unlike the linear activation, this function introduces non-linearity to the network and indicates the side of a hyperplane an input rests on, but the notion of Euclidean distance is lost. If this network were to extend further in depth, it is expected that the metrics would be even worse since $h_i(x) = W_i^T + b_i$ would be operating on a vector of binary values, so expressivity is particularly sensitive to the choice of W and b.

In summary, the ReLU activation offers two properties that are key for the classification problem: preservation of distance and indication of class labeling via its non-linearity.

3.6. **The Impact of Random Initialization.** Lastly, we touch on the nature of our network being randomly initialized. By randomly initializing weights and biases and running many trials, we create a swath of architectures that can span the space of possible combinations. This effectively sidesteps the added time complexity of training a network while matching the performance of a trained network. Hence, setting a bias vector deterministically, for example, restricts the network from covering a larger space in the same number of trials. This effect is demonstrated in the figure below.

In the experiments we repeat the tests conducted for different supports on the bias vector sampling that was outlined in Section 3.4. We fix the depth of the network to a single hidden layer. For the networks whose bias vectors are deterministic, we set bias to 0 for the symmetric case, λ for the positive case, and $-\lambda$ for the negative case the reflect the mean of each distribution. Through the experimentation, we clearly observe that sampling the bias vectors randomly leads to greater linear separability in terms of average accuracy and separation probability. Moreover, we see that networks of 0 bias perform the worse, implying that some perturbation is needed to break symmetry when initializing the networks.

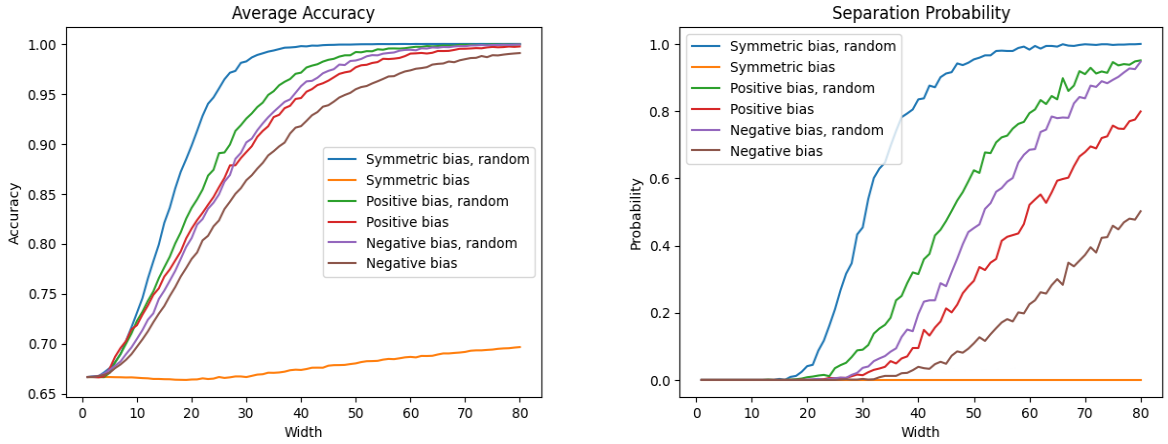


FIGURE 10. Performance of RINNs with deterministic and random bias vectors. We fix the depth to 1 and vary the width of the network. We see that randomization separates the data with higher probability for each bias distribution.

In summary, randomly initializing the networks and testing many trials allows for trials to explore a larger space of architectures and achieve greater linear separability in lieu of training. A future experiment that we could explore is finding bounds on how many RINNs would be needed to match the performance of a trained network, and whether it still saves time.

4. CONCLUSION

In this report, we explore the capacity of randomly initialized feedforward networks to separate data into two classes. Our experiments explore a swath of architectures by randomly sampling parameter values and compare average results for different tests. This approach sidesteps the added time complexity of fully training a network and allows us to explore probabilistic properties of the unsupervised learning problem theoretically and experimentally.

The theoretical justification for the method is outlined in Section 2. Namely, the problem boils down to finding a parameterization for a neural network that effectively maps points of different classes to opposite sides of the hyperplane.

In Section 3, we conduct a group of experiments to explore the problem empirically, and we extend experimental results of previous papers to networks with a third hidden layer. We find that with our synthetic data sets, when weights are sampled from a standard Gaussian distribution and biases are sampled from a uniform distribution symmetric about 0, adding depth to the network does not necessarily improve linear separability. However, when the biases are drawn from a distribution that is strictly positive, adding depth does improve linear separability. The latter of these results suggests that fully training the networks rather than testing untrained architectures would produce biases that are positive.

We also examined the efficacy of the rectified linear unit (ReLU) activation function in this particular problem. Applying the ReLU to all neurons leads to the best results among the activation functions we tested, and we suggest that this is due to a pair of properties unique to ReLU. In particular, ReLU allows the network to preserve the Euclidean distance to the hyperplane that is

to be estimated, and the function's break in linearity allows the network to determine which side of the hyperplane that an input lies on. Other activation functions violate at least one of these two properties and thus produce results that are not competitive.

Looking ahead, an important experiment to perform next would be examining the impact of the neural network architecture on a dataset with increased complexity. The advantage of neural networks at disentangling complex geometry of datasets makes us believe that for a more complex low-dimensional dataset, deeper network will perform better. However, the case for high-dimensional datasets remains intriguing.

We give thanks to the authors of [GS22] for being so gracious as to provide us with their source code.

References

[DS21] Genzel M. Jacques L. Dirksen, S. and A. Stollenwerk. The separation capacity of random neural networks. *arXiv preprint arXiv:2108.00207*, 2021.

[GC16] Bengio Y. Goodfellow, I. and A. Courville. *Deep Learning*. MIT Press, Cambridge, MA, USA, 2016. http://www.deeplearningbook.org.

[GS22] Mahankali S. Ghosal, P. and Y. Sun. Randomly initialized one-layer neural networks make data linearly separable. *arXiv preprint arXiv:2205.11716*, 2022.

[Ver18] R. Vershynin. *High-dimensional probability*, volume 47 of *Cambridge Series in Statistical and Probabilistic Mathematics*. Cambridge University Press, Cambridge, 2018. An introduction with applications in data science, With a foreword by Sara van de Geer.

A. Cutuli, Department of Industrial Engineering and Operations Research, Columbia University, 500 W 120th St 315, New York, NY 10027 USA

Email address: ajc2312@columbia.edu

H. Miao, Department of Industrial Engineering and Operations Research, Columbia University, 500 W 120th St 315, New York, NY 10027 USA

Email address: hm2935@columbia.edu

W. Zhu, Department of Mathematics, Columbia University, 2990 Broadway, New York, NY 10027 USA

Email address: wz2453@columbia.edu