Proceedings of Machine Learning Research vol 125:1–22, 2020 33rd Annual Conference on Learning Theory

Universal Approximation with Deep Narrow Networks

Patrick Kidger KIDGER@MATHS.OX.AC.UK
Terry Lyons TLYONS@MATHS.OX.AC.UK
Mathematical Institute, University of Oxford

Editors: Jacob Abernethy and Shivani Agarwal

Abstract

The classical Universal Approximation Theorem holds for neural networks of arbitrary width and bounded depth. Here we consider the natural 'dual' scenario for networks of bounded width and arbitrary depth. Precisely, let n be the number of inputs neurons, m be the number of output neurons, and let ρ be any nonaffine continuous function, with a continuous nonzero derivative at some point. Then we show that the class of neural networks of arbitrary depth, width $n + m + 2$, and activation function ρ, is dense in $C(K; \mathbb{R}^m)$ for $K \subseteq \mathbb{R}^n$ with K compact. This covers every activation function possible to use in practice, and also includes polynomial activation functions, which is unlike the classical version of the theorem, and provides a qualitative difference between deep narrow networks and shallow wide networks. We then consider several extensions of this result. In particular we consider nowhere differentiable activation functions, density in noncompact domains with respect to the L^p-norm, and how the width may be reduced to just $n + m + 1$ for 'most' activation functions.

Keywords: universal approximation, neural network, deep, narrow, bounded width
MSC (2020): 41A46, 41A63, 68T07

1. Introduction

Recall the classical Universal Approximation Theorem (Cybenko, 1989; Hornik, 1991; Pinkus, 1999).

Theorem 1.1 *Let $\rho\colon \mathbb{R} \to \mathbb{R}$ be any continuous function. Let $\mathcal{N}_n^\rho$ represent the class of feedforward neural networks with activation function ρ, with n neurons in the input layer, one neuron in the output layer, and one hidden layer with an arbitrary number of neurons. Let $K \subseteq \mathbb{R}^n$ be compact. Then $\mathcal{N}_n^\rho$ is dense in $C(K)$ if and only if ρ is nonpolynomial.*

Extending this result to any bounded number of hidden layers is easy, by simply requiring that the 'extra' hidden layers approximate the identity function. Thus the classical theorem addresses the case of arbitrary width and bounded depth.

This motivates a natural 'dual' scenario, in which the class of neural networks is of bounded width and arbitrary depth. We refer to networks of this type as *deep, narrow* networks. Natural questions are then what activation functions may be admitted, in what topologies density may be established, and how narrow the network may be made.

Notable existing work on this problem has been performed by Lu et al. (2017) and Hanin and Sellke (2017), however both of these studies only consider the ReLU activation function. In particular they rely on its explicit form and friendly algebraic properties.

The primary aim of this article is to addess this limitation, by considering essentially arbitrary activation functions. In particular we will find that polynomial activation functions are valid choices, meaning that deep and narrow networks behave distinctly differently to shallow and wide networks. We also provide some results on the choice of topology and the width of the network.

The rest of the paper is laid out as follows. Section 2 discusses existing work. Section 3 provides a summary of our results; these are then presented in detail in Section 4. Section 5 is the conclusion. A few proofs are deferred to the appendices.

2. Existing work

Some positive results have been established showing density of particular deep narrow networks.

Hanin and Sellke (2017) have shown that deep narrow networks with the ReLU activation function are dense in $C(K;\mathbb{R}^m)$ for $K \subseteq \mathbb{R}^n$ compact, and require only width $n+m$. Lu et al. (2017) have shown that deep narrow networks with the ReLU activation function are dense in $L^1(\mathbb{R}^n)$, with width $n+4$. Lin and Jegelka (2018) have shown that a particular description of residual networks, with the ReLU activation function, are dense in $L^1(\mathbb{R}^n)$.

We are not aware of any previously obtained positive results for activation functions other than the ReLU, or for the general case of $L^p(\mathbb{R}^n;\mathbb{R}^m)$ for $p \in [1,\infty)$ and $m \in \mathbb{N}$.

Moving on, some negative results have been established, about insufficiently wide networks.

Consider the case of a network with n input neurons and a single output neuron. For certain activation functions, Johnson (2019) shows that width n is insufficient to give density in $C(K)$. For the ReLU activation function, Lu et al. (2017) show that width n is insufficient to give density in $L^1(\mathbb{R}^n)$, and that width $n-1$ is insufficient in $L^1([-1,1]^n)$, whilst Hanin and Sellke (2017) show that width n is insufficient to give density in $C(K)$.

Everything discussed so far is in the most general case of approximating functions on Euclidean space. There has also been some related work for classification tasks (Beise et al., 2018; Szymanski and McCane, 2012; Rojas, 2003; Nguyen et al., 2018). There has also been some related work in the special case of certain finite domains; Sutskever and Hinton (2008); Le Roux and Bengio (2010) consider distributions on $\{0,1\}^n$. Montúfar (2014) consider distributions on $\{0,1,\dots,q-1\}^n$.

3. Summary of Results

Definition 3.1 *Let $\rho\colon \mathbb{R} \to \mathbb{R}$ and $n,m,k \in \mathbb{N}$. Then let $\mathcal{NN}^{\rho}_{n,m,k}$ represent the class of functions $\mathbb{R}^n \to \mathbb{R}^m$ described by feedforward neural networks with n neurons in the input layer, m neurons in the output layer, and an arbitrary number of hidden layers, each with k neurons with activation function ρ. Every neuron in the output layer has the identity activation function.*

Our main result is the following theorem.

Theorem 3.2 *Let $\rho\colon \mathbb{R} \to \mathbb{R}$ be any nonaffine continuous function which is continuously differentiable at at least one point, with nonzero derivative at that point. Let $K \subseteq \mathbb{R}^n$ be compact. Then $\mathcal{NN}^{\rho}_{n,m,n+m+2}$ is dense in $C(K;\mathbb{R}^m)$ with respect to the uniform norm.*

The key novelty here is the ability to handle essentially arbitrary activation functions, and in particular polynomials, which is a qualitative difference compared to shallow networks. In particular we have not relied on the explicit form of the ReLU, or on its favourable algebraic properties.

The technical condition is very weak; in particular it is satisfied by every piecewise-C^1 function not identically zero. Thus any activation function that one might practically imagine using on a computer must satisfy this property.

Theorem 3.2 is proved by handling particular classes of activation functions as special cases.

Proposition 4.9 *Let $\rho\colon \mathbb{R} \to \mathbb{R}$ be any continuous nonpolynomial function which is continuously differentiable at at least one point, with nonzero derivative at that point. Let $K \subseteq \mathbb{R}^n$ be compact. Then $\mathcal{NN}^{\rho}_{n,m,n+m+1}$ is dense in $C(K;\mathbb{R}^m)$ with respect to the uniform norm.*

Proposition 4.11 *Let $\rho\colon \mathbb{R} \to \mathbb{R}$ be any nonaffine polynomial. Let $K \subseteq \mathbb{R}^n$ be compact. Then $\mathcal{NN}^{\rho}_{n,m,n+m+2}$ is dense in $C(K;\mathbb{R}^m)$ with respect to the uniform norm.*

It is clear that Propositions 4.9 and 4.11 together imply Theorem 3.2. Note the slight difference in their required widths. Furthermore we will see that their manner of proofs are rather different.

As a related result, some of the techniques used may also be shown to extend to certain unfriendly-looking activation functions, that do not satisfy the technical condition of Theorem 3.2.

Proposition 4.15 *Let $w\colon \mathbb{R} \to \mathbb{R}$ be any bounded continuous nowhere differentiable function. Let $\rho(x) = \sin(x) + w(x)\mathrm{e}^{-x}$, which will also be nowhere differentiable. Let $K \subseteq \mathbb{R}^n$ be compact. Then $\mathcal{NN}^{\rho}_{n,m,n+m+1}$ is dense in $C(K;\mathbb{R}^m)$ with respect to the uniform norm.*

Moving on to a different related result, we consider the case of a noncompact domain.

Theorem 4.16 *Let ρ be the ReLU. Let $p \in [1,\infty)$. Then $\mathcal{NN}^{\rho}_{n,m,n+m+1}$ is dense in $L^p(\mathbb{R}^n;\mathbb{R}^m)$ with respect to the usual L^p norm.*

Whilst only about the ReLU, the novelty of this result is how it generalises (Lu et al., 2017, Theorem 1) in multiple ways: to a narrower width, multiple outputs, and L^p instead of just L^1.

As a final related result, we observe that the smaller width of $n+m+1$ also suffices for a large class of polynomials. Together with Proposition 4.9, this means that the smaller width of $n+m+1$ suffices for 'most' activation functions.

Proposition 4.17 *Let $\rho\colon \mathbb{R} \to \mathbb{R}$ be any polynomial for which there exists a point $\alpha \in \mathbb{R}$ such that $\rho'(\alpha) = 0$ and $\rho''(\alpha) \neq 0$. Let $K \subseteq \mathbb{R}^n$ be compact. Then $\mathcal{NN}^{\rho}_{n,m,n+m+1}$ is dense in $C(K;\mathbb{R}^m)$ with respect to the uniform norm.*

Remark 3.3 *Every proof in this article is constructive, and can in principle be traced so as to determine how depth changes with approximation error. We have elected not to present this, as depth-efficient versions of our constructions quickly become unclear to present. Furthermore this would require tracing the (constructive) proofs of the Stone–Weierstrass Theorem and the classical Universal Approximation Theorem, to which we appeal.*

4. Universal approximation

4.1. Preliminaries

A neuron is usually defined as an activation function composed with an affine function. For ease, we shall extend the definition of a neuron to allow it to represent a function of the form $\psi \circ \rho \circ \phi$, where ψ and ϕ are affine functions, and ρ is the activation function. This does not increase the

representational power of the network, as the new affine functions may be absorbed into the affine parts of the next layer, but it will make the neural representation of many functions easier to present. We refer to these as *enhanced neurons*. It is similarly allowable to take affine combinations of multiple enhanced neurons; we will use this fact as well.

One of the key ideas behind our constructions is that most reasonable activation functions can be taken to approximate the identity function. Indeed, this is essentially the notion that differentiability captures: that a function is locally affine. This makes it possible to treat neurons as 'registers', in which information may be stored and preserved through the layers. Thus our constructions have strong overtones of space-limited algorithm design in traditional computer science settings; in our proofs we will often think of 'storing' a value in a particular register.

Lemma 4.1 *Let $\rho\colon \mathbb{R} \to \mathbb{R}$ be any continuous function which is continuously differentiable at at least one point, with nonzero derivative at that point. Let $L \subseteq \mathbb{R}$ be compact. Then a single enhanced neuron with activation function ρ may uniformly approximate the identity function $\iota\colon \mathbb{R} \to \mathbb{R}$ on L, with arbitrarily small error.*

Proof By assumption, as ρ is *continuously* differentiable, there exists $[a, b] \subseteq \mathbb{R}$ with $a \neq b$, on some neighbourhood of which ρ is differentiable, and $\alpha \in (a, b)$ at which ρ' is continuous, and for which $\rho'(\alpha)$ is nonzero.

For $h \in \mathbb{R} \setminus \{0\}$, let $\phi_h(x) = hx + \alpha$, and let

$$\psi_h(x) = \frac{x - \rho(\alpha)}{h\rho'(\alpha)}.$$

Then $\iota_h = \psi_h \circ \rho \circ \phi_h$ is of the form that an enhanced neuron can represent. Then for all $u \in [a, b]$, by the Mean Value Theorem there exists ξ_u between u and α such that

$$\rho(u) = \rho(\alpha) + (u - \alpha)\rho'(\xi_u),$$

and hence

$$\begin{aligned}\iota_h(x) &= (\psi_h \circ \rho \circ \phi_h)(x) \\ &= \psi_h\left(\rho(\alpha) + hx\rho'(\xi_{hx+\alpha})\right) \\ &= \frac{x\rho'(\xi_{hx+\alpha})}{\rho'(\alpha)}\end{aligned}$$

for h sufficiently small that $\phi_h(L) \subseteq [a, b]$.

Now let ρ' have modulus of continuity ω on $[a, b]$. Let $\iota\colon \mathbb{R} \to \mathbb{R}$ represent the identity function. Then for all $x \in L$,

$$\begin{aligned}|\iota_h(x) - \iota(x)| &= |x|\left|\frac{\rho'(\xi_{hx+\alpha}) - \rho'(\alpha)}{\rho'(\alpha)}\right| \\ &\leqslant \frac{|x|}{|\rho'(\alpha)|}\omega(hx),\end{aligned}$$

and so $\iota_h \to \iota$ uniformly over L. ∎

Notation *Throughout the rest of this paper ι_h will be used to denote such an approximation to the identity function, where $\iota_h \to \iota$ uniformly as $h \to 0$.*

An enhanced neuron may be described as performing (for example) the computation $x \mapsto \iota_h(4x + 3)$. This is possible as the affine transformation $x \mapsto 4x + 3$ and the affine transformation ϕ_h (from the description of ι_h) may be combined together into a single affine transformation.

Now that we can approximate the identity funciton, another important ingredient will be a model that actually uses identity functions for us to approximate! As such we now consider the 'Register Model', which represents a simplification of a neural network.

Proposition 4.2 (Register Model) *Let $\rho\colon \mathbb{R} \to \mathbb{R}$ be any continuous nonpolynomial function. Let $\mathcal{I}^{\rho}_{n,m,n+m+1}$ represent the class of neural networks with n neurons in the input layer, m neurons in the output layer, and an arbitrary number of hidden layers, each with $n + m$ neurons with the identity activation function, and one neuron with activation function ρ. Let $K \subseteq \mathbb{R}^n$ be compact. Then $\mathcal{I}^{\rho}_{n,m,n+m+1}$ is dense in $C(K; \mathbb{R}^m)$.*

The Register Model is somewhat similar to the constructions used in (Lu et al., 2017; Hanin and Sellke, 2017), although their constructions are specific to the ReLU. As such we defer the proof to Appendix A.

Next we consider another different simplification, which we refer to as the 'Square Model'. Its proof is a little involved, so for clarity we present it in a new subsection.

4.2. Square Model

Lemma 4.3 *One layer of two enhanced neurons, with square activation function, may exactly represent the multiplication function $(x, y) \mapsto xy$ on $\mathbb{R}^2$.*

Proof Let the first neuron compute $\eta = (x + y)^2$. Let the second neuron compute $\zeta = (x - y)^2$. Then $xy = (\eta - \zeta)/4$. (This final affine transformation is allowed between enhanced neurons.) ■

Lemma 4.4 *Fix $L \subseteq \mathbb{R}^2$ compact. Three layers of two enhanced neurons each, with square activation function, may uniformly approximate $(x, y) \mapsto (x^2, y(x + 1))$ arbitrarily well on L.*

Proof Let $h, s \in \mathbb{R}\backslash\{0\}$. Let η_1, η_2, η_3 represent the first neuron in each layer; let $\zeta_1, \zeta_2, \zeta_3$ represent the second neuron in each layer. Let ι_h represent an approximation to the identity in the manner of Lemma 4.1. Using '$\approx$' as an informal notation to represent 'equal to up to approximation of the identity', assign values to η_1, η_2, η_3 and $\zeta_1, \zeta_2, \zeta_3$ as follows:

$$
\begin{aligned}
\eta_1 &= \iota_h(x) & \zeta_1 &= (x + sy + 1)^2 \\
&\approx x, & &= x^2 + 2sxy + s^2y^2 + 2x + 2sy + 1, \\
\eta_2 &= (\eta_1)^2 & \zeta_2 &= \iota_h(\zeta_1 - 2\eta_1 - 1) \\
&\approx x^2, & &\approx x^2 + 2sxy + s^2y^2 + 2sy, \\
\eta_3 &= \iota_h(\eta_2) & \zeta_3 &= \iota_h((\zeta_2 - \eta_2)/2s) \\
&\approx x^2, & &\approx xy + y + sy^2/2.
\end{aligned}
$$

And so η_3 may be taken arbitrarily close to x^2 and ζ_3 may be taken arbitrarily close to $y(x+1)$, with respect to $\|\cdot\|_\infty$ on L, by first taking s arbitrarily small, and then taking h arbitrarily small. ■

Lemma 4.5 *Fix $L \subseteq (0,2)$ compact. Then multiple layers of two enhanced neurons each, with square activation function, may uniformly approximate $x \mapsto 1/x$ arbitrarily well on L.*

Proof First note that

$$\prod_{i=0}^{n}(1+x^{2^i}) \to \frac{1}{1-x}$$

as $n \to \infty$, uniformly over compact subsets of $(-1,1)$. Thus,

$$(2-x)\prod_{i=1}^{n}(1+(1-x)^{2^i}) = \prod_{i=0}^{n}(1+(1-x)^{2^i}) \to \frac{1}{x}$$

uniformly over L.

This has the following neural approximation: let $\eta_1 = (1-x)^2$ and $\zeta_1 = \iota_h(2-x)$ be the neurons in the first layer, where ι_h is some approximation of the identity as in Lemma 4.1. Let κ_h represent an approximation to $(x,y) \mapsto (x^2, y(x+1))$ in the manner of Lemma 4.4, with error made arbitrarily small as $h \to 0$. Now for $i \in \{1,4,7,10,\ldots,3n-2\}$, recursively define $(\eta_{i+3},\zeta_{i+3}) = \kappa_h(\eta_i,\zeta_i)$, where we increase the index by three to represent the fact that three layers are used to perform this operation. So up to approximation, $\eta_{i+3} \approx (\eta_i)^2$, and $\zeta_{i+3} \approx \zeta_i(\eta_i+1)$.

So $\zeta_{3n+1} \to (2-x)\prod_{i=1}^{n}(1+(1-x)^{2^i})$ uniformly over L as $h \to 0$. Thus the result is obtained by taking first n large enough and then h small enough. ∎

Proposition 4.6 (Square Model) *Let $\rho(x) = x^2$. Let $K \subseteq \mathbb{R}^n$ be compact. Then $\mathcal{NN}^{\rho}_{n,m,n+m+1}$ is dense in $C(K;\mathbb{R}^m)$.*

Proof Fix $f = (f_1,\ldots,f_m) \in C(K;\mathbb{R}^m)$. Fix $\varepsilon > 0$. By precomposing with an affine function, which may be absorbed into the first layer of the network, assume without loss of generality that

$$K \subseteq (1,2)^n. \tag{1}$$

By the Stone–Weierstrass Theorem there exist polynomials $g_1,\ldots,g_m$ in the variables $x_1,\ldots x_n$ approximating $f_1,\ldots,f_m$ to within $\varepsilon/3$ with respect to $\|\cdot\|_\infty$.

We will construct a network in $\mathcal{NN}^{\rho}_{n,m,n+m+1}$ approximating $g_1,\ldots,g_n$. There will be a total of $n+m+1$ neurons in each hidden layer. In each hidden layer, for each $i \in \{1,\ldots,n\}$, we associate an input x_i with a neuron, which we shall refer to as the x_i-in-register neuron. Similarly for each $i \in \{1,\ldots,m\}$, we will associate an output g_i and a neuron, which we shall refer to as the g_i-out-register neuron. The final neuron in each layer will be referred to as the computation neuron.

Our proof will progress by successively adding layers to the network.

We begin by constructing approximations to $g_2,\ldots,g_m$. (Constructing the approximation to the final g_1 will be more challenging.) These next few paragraphs may be skipped if $m=1$.

In each hidden layer, for $i \in \{1,\ldots,n\}$, have the x_i-in-register neuron apply the approximate identity function in the manner of Lemma 4.1 to the x_i-in-register neuron of the previous layer, or to the input x_i if it is the first hidden layer. In this way the in-register neurons will preserve the inputs to the network, so that the values of x_i are accessible by the other neurons in every layer.

(At least, up to an arbitrarily good approximation of the identity. For the sake of sanity of notation, we shall suppress this detail in our notation, and refer to our neurons in later layers as having e.g. 'x_1' as an input to them.)

Now write $g_2 = \sum_{j=1}^{N} \delta_j$, where each δ_j is a monomial. Using just the computation neuron and the g_1-out-register neuron in multiple consecutive layers, perform successive multiplications in the manner of Lemma 4.3 to compute the value of δ_1. For example, if $\delta_1 = x_1^2 x_2 x_3$, then a suitable chain of multiplications is $x_1(x_1(x_2x_3))$. In each layer, each x_i is available as an input because it is stored in the in-register neurons, and the intermediate partial products are available as they have just been computed in the preceding layer. The value for δ_1 is then stored in the g_2-out-register neuron and kept through the subsequent layers via approximate identity functions.

The previous paragraph is somewhat wordy, so it is worth pausing to make clear what the appropriate mental model is for validating that this construction is correct. It is simply that at each layer, we have at most $n + m + 1$ values available from the preceding layer, and we must now allocate a budget of at most $n + m + 1$ enhanced neurons, describing how to obtain the at most $n + m + 1$ outputs from the layer. Values may be copied, moved and added between neurons using the affine part of a layer, and preserved between layers using ι_h. We have not yet needed our full budget of neurons, so far.

This process is then repeated for δ_2, again using just the computation neuron and the g_1-out-register neuron. The result is then added on to the g_2-out-register neuron, via the affine part of the operation of this neuron. Repeat for all j until all of the δ_j have been computed and added on, so that the g_2-out-register neuron stores an approximation to g_2.

This is only an approximation in that we have used approximations to the identity function; other than that it is exact. As such, by taking sufficiently good approximations of the identity function, this will be a uniform approximation to g_2 over K. For the remaining layers of the network, have the g_2-out-register neurons just preserve this value with approximate identity functions.

Now repeat this whole process for g_i and the g_i-out-register neurons, for $i \in \{3, \dots, m\}$. Let the computed values be denoted $\widehat{g}_2, \dots, \widehat{g}_m$. (With the 'hat' notation because of the fact that these are not the values $g_2, \dots, g_m$, due to the approximate identity functions in between.)

The difficult bit is computing an approximation to g_1, as it must be done without the 'extra' g_1-in-register neuron. Going forward we now only have $n+2$ neurons available in each layer: the n in-register neurons (which have so far been storing the inputs $x_1, \dots x_n$), the computation neuron, and the g_1-out-register neuron.

Written in terms of monomials, let $g_1 = \sum_{j=1}^{M} \gamma_j$. Then g_1 may be written as

$$g_1 = \gamma_1 \left(1 + \frac{\gamma_2}{\gamma_1}\left(1 + \frac{\gamma_3}{\gamma_2}\left(\cdots\left(1 + \frac{\gamma_{M-1}}{\gamma_{M-2}}\left(1 + \frac{\gamma_M}{\gamma_{M-1}}\right)\right)\cdots\right)\right)\right).$$

Note that this description is defined over K, as K is bounded away from the origin by equation (1).

Now write $\gamma_j = \prod_{k=1}^{n} x_k^{\theta_{j,k}}$, for $\theta_{j,k} \in \mathbb{N}_0$. Substituting this in,

$$g_1 = \left[\prod_{k=1}^{n} x_k^{\theta_{1,k}}\right]\left(1 + \frac{\prod_{k=1}^{n} x_k^{\theta_{2,k}}}{\prod_{k=1}^{n} x_k^{\theta_{1,k}}}\left(1 + \frac{\prod_{k=1}^{n} x_k^{\theta_{3,k}}}{\prod_{k=1}^{n} x_k^{\theta_{2,k}}}\left(\cdots\right.\right.\right.$$
$$\left.\left.\left.\left(1 + \frac{\prod_{k=1}^{n} x_k^{\theta_{M-1,k}}}{\prod_{k=1}^{n} x_k^{\theta_{M-2,k}}}\left(1 + \frac{\prod_{k=1}^{n} x_k^{\theta_{M,k}}}{\prod_{k=1}^{n} x_k^{\theta_{M-1,k}}}\right)\right)\cdots\right)\right)\right).$$

Now let $\sup K$ be defined by

$$\sup K = \sup\{x_i \,|\, (x_1, \dots, x_n) \in K\},$$

so that $1 < \sup K < 2$. Let r be an approximation to $x \mapsto 1/x$ in the manner of Lemma 4.5, with the L of that proposition given by

$$L = [(\sup K)^{-1} - \alpha, \sup K + \alpha] \subseteq (0, 2), \tag{2}$$

where $\alpha > 0$ is taken small enough that the inclusion holds.

Let r^a denote r composed a times. By taking r to be a suitably good approximation, we may ensure that $\widetilde{g}_1$ defined by

$$\begin{aligned}\widetilde{g}_1 = \left[\prod_{k=1}^{n} r^{2M-2}(x_k)^{\theta_{1,k}}\right] \Bigg(1 + \left[\prod_{k=1}^{n} r^{2M-3}(x_k)^{\theta_{1,k}}\right] \left[\prod_{k=1}^{n} r^{2M-4}(x_k)^{\theta_{2,k}}\right] \\ \Bigg(1 + \left[\prod_{k=1}^{n} r^{2M-5}(x_k)^{\theta_{2,k}}\right] \left[\prod_{k=1}^{n} r^{2M-6}(x_k)^{\theta_{3,k}}\right] \qquad (3) \\ \Bigg(\cdots \\ \Bigg(1 + \left[\prod_{k=1}^{n} r^{3}(x_k)^{\theta_{M-2,k}}\right] \left[\prod_{k=1}^{n} r^{2}(x_k)^{\theta_{M-1,k}}\right] \\ \Bigg(1 + \left[\prod_{k=1}^{n} r(x_k)^{\theta_{M-1,k}}\right] \left[\prod_{k=1}^{n} x_k^{\theta_{M,k}}\right]\Bigg)\Bigg) \\ \cdots\Bigg)\Bigg)\Bigg) \qquad (4)\end{aligned}$$

is an approximation to g_1 in K, to within $\varepsilon/3$, with respect to $\|\cdot\|_\infty$. This is possible by equations (1) and (2); in particular the approximation should be sufficiently precise that

$$r^{2M-2}([(\sup K)^{-1}, \sup K]) \subseteq L,$$

which is possible due to the margin $\alpha > 0$. Note how r^2, and thus r^4, r^6, …, r^{2M-2}, are approximately the identity function on L.

This description of $\widetilde{g}_1$ is now amenable to representation with a neural network. The key fact about this description of $\widetilde{g}_1$ is that, working from the most nested set of brackets outwards, the value of $\widetilde{g}_1$ may be computed by performing a single chain of multiplications and additions, along with occasionally taking the reciprocal of all of the input values. Thus unlike our earlier computations involving the monomial δ_j, we do not need to preserve an extra partially-computed piece of information between layers.

So let the computation neuron and the g_1-out-register neuron perform the multiplications, layer-by-layer, to compute $\prod_{k=1}^{n} x_k^{\theta_{M,k}}$, in the manner of Lemma 4.3. Store this value in the g_1-out-register neuron.

Now use the computation neuron and the x_1-in-register neuron, across multiple layers, to compute $r(x_1)$, in the manner of Lemma 4.5. Eventually the x_1-in-register neuron will be storing $r(x_1) \approx 1/x_1$. Repeat for the other in-register neurons, so that they are collectively storing $r(x_1), \ldots, r(x_n)$.

Now the computation neuron and the g_1-out-register neuron may start multipying $r(x_1), \ldots, r(x_n)$ on to $\prod_{k=1}^{n} x_k^{\theta_{M,k}}$ (which is the value presently stored in the g_1-out-register neuron) the appropriate number to times to compute $\left[\prod_{k=1}^{n} r(x_k)^{\theta_{M-1,k}}\right]\left[\prod_{k=1}^{n} x_k^{\theta_{M,k}}\right]$, by Lemma 4.3. Store this value in the out-register neuron. Then add one (using the affine part of a layer). The out-register neuron has now computed the expression in the innermost bracket in equation (4).

The general pattern is now clear: apply r to all of the in-register neurons again to compute $r^2(x_i)$, multiply them on to the value in the out-register neuron, and so on. Eventually the g_1-out-register neuron will have computed an approximation to $\widetilde{g}_1$. It will have computed some approximation $\widehat{g}_1$ to this value, because of the identity approximations involved.

Thus the out-register neurons have computed $\widehat{g}_1, \ldots, \widehat{g}_m$. Now simply have the output layer copy the values from the out-register neurons.

Uniform continuity preserves uniform convergence, compactness is preserved by continuous functions, and a composition of two uniformly convergent sequences of functions with uniformly continuous limits is again uniformly convergent. So by taking all of the (many) identity approximations throughout the network to be suitably precise, then $\widetilde{g}_1$ and $\widehat{g}_1$ may be taken within $\varepsilon/3$ of each other, and the values of $\widehat{g}_2, \ldots, \widehat{g}_m$ and $g_2, \ldots, g_m$ may be taken within $2\varepsilon/3$ of each other, in each case with respect to $\|\cdot\|_\infty$ on K.

Thus $(\widehat{g}_1, \ldots, \widehat{g}_m)$ approximates f with total error no more than ε, and the proof is complete. ■

Remark 4.7 *Lemma 4.5 is key to the proof of Proposition 4.6. It was fortunate that the reciprocal function may be approximated by a network of width two - note that even if Proposition 4.6 were already known, it would have required a network of width three. It remains unclear whether an arbitrary-depth network of width two, with square activation function, is dense in $C(K)$.*

Remark 4.8 *Note that allowing a single extra neuron in each layer would remove the need for the trick with the reciprocal, as it would allow g_1 to be computed in the same way as $g_2, \ldots, g_m$. Doing so would dramatically reduce the depth of the network. We are thus paying a heavy price in depth in order to reduce the width by a single neuron.*

4.3. Key results

Proposition 4.9 *Let $\rho\colon \mathbb{R} \to \mathbb{R}$ be any continuous nonpolynomial function which is continuously differentiable at at least one point, with nonzero derivative at that point. Let $K \subseteq \mathbb{R}^n$ be compact. Then $\mathcal{NN}^{\rho}_{n,m,n+m+1}$ is dense in $C(K;\mathbb{R}^m)$ with respect to the uniform norm.*

Proof Let $f \in C(K;\mathbb{R}^m)$ and $\varepsilon > 0$. Set up a neural network as in the Register Model (Proposition 4.2), approximating f to within $\varepsilon/2$. Every neuron requiring an identity activation function in the Register Model will instead approximate the identity with ι_h, in the manner of Lemma 4.1.

Uniform continuity preserves uniform convergence, compactness is preserved by continuous functions, and a composition of two uniformly convergent sequences of functions with uniformly continuous limits is again uniformly convergent. Thus the new model can be taken within $\varepsilon/2$ of the Register Model, with respect to $\|\cdot\|_\infty$ in K, by taking h sufficiently small. ■

Remark 4.10 *This of course implies approximation in $L^p(K, \mathbb{R}^m)$ for $p \in [1, \infty)$. However, when ρ is the ReLU activation function, then Theorem 4.16, later, shows that in fact the result may be generalised to noncompact domains.*

Moving on, we consider polynomial activation functions. We now show that it is a consequence of Proposition 4.6 that any (polynomial) activation function which can approximate the square activation function, in a suitable manner, is also capable of universal approximation.

Proposition 4.11 *Let $\rho\colon \mathbb{R} \to \mathbb{R}$ be any nonaffine polynomial. Let $K \subseteq \mathbb{R}^n$ be compact. Then $\mathcal{NN}^{\rho}_{n,m,n+m+2}$ is dense in $C(K; \mathbb{R}^m)$ with respect to the uniform norm.*

Proof Fix $\alpha \in \mathbb{R}$ such that $\rho''(\alpha) \neq 0$, which exists as ρ is nonaffine. Now let $h \in (0, \infty)$. Define $\sigma_h\colon \mathbb{R} \to \mathbb{R}$ by

$$\sigma_h(x) = \frac{\rho(\alpha + hx) - 2\rho(\alpha) + \rho(\alpha - hx)}{h^2\rho''(\alpha)}.$$

Then Taylor expanding $\rho(\alpha + hx)$ and $\rho(\alpha - hx)$ around α,

$$\begin{aligned}\sigma_h(x) &= \frac{\rho(\alpha) + hx\rho'(\alpha) + h^2x^2\rho''(\alpha)/2 + \mathcal{O}(h^3x^3)}{h^2\rho''(\alpha)} - \frac{2\rho(\alpha)}{h^2\rho''(\alpha)} + \\ &\qquad \frac{\rho(\alpha) - hx\rho'(\alpha) + h^2x^2\rho''(\alpha)/2 + \mathcal{O}(h^3x^3)}{h^2\rho''(\alpha)} \\ &= x^2 + \mathcal{O}(hx^3).\end{aligned}$$

Observe that σ_h needs precisely two operations of ρ on (affine transformations of) x, and so may be computed by two enhanced neurons with activation function ρ. Thus the operation of a single enhanced neuron with square activation function may be approximated by two enhanced neurons with activation function ρ.

Let N be any network as in the Square Model (Proposition 4.6). Let ℓ be any hidden layer of N; it contains $n + m + 1$ neurons. Let η be a vector of the values of the neurons of the previous layer. Let ϕ_i be the affine part of the ith neuron of ℓ, so that ℓ computes $\phi_1(\eta)^2, \ldots, \phi_{n+m+1}(\eta)^2$. Then this may equivalently be calculated with $n + m + 1$ layers of $n + m + 1$ neurons each, with $n + m$ of the neurons in each of these new layers using the identity function, and one neuron using the square activation function. This is done by having the first of these new layers apply the ϕ_i, and having the ith layer square the value of the ith neuron. See Figure 1.

Apply this procedure to every layer of N; call the resulting network $\widetilde{N}$. It will compute exactly the same function as N, and will have $n + m + 1$ times as many layers, but will use only a single squaring operation in each layer.

Create a copy of $\widetilde{N}$, call it $\widetilde{N}_h$. Replace its identity activation functions with approximations in the manner of Lemma 4.1, using activation function ρ. Replace its square activation functions (one in each layer) by approximations in the manner described above with σ_h; this requires an extra neuron in each hidden layer, so that the network is now of width $n + m + 2$. Thus $\widetilde{N}_h$ uses the activation function ρ throughout.

Uniform continuity preserves uniform convergence, compactness is preserved by continuous functions, and a composition of two uniformly convergent sequences of functions with uniformly continuous limits is again uniformly convergent. Thus the difference between $\widetilde{N}_h$ and $\widetilde{N}$, with respect to $\|\cdot\|_\infty$ on K, may be taken arbitrarily small by taking h arbitrarily small. ∎

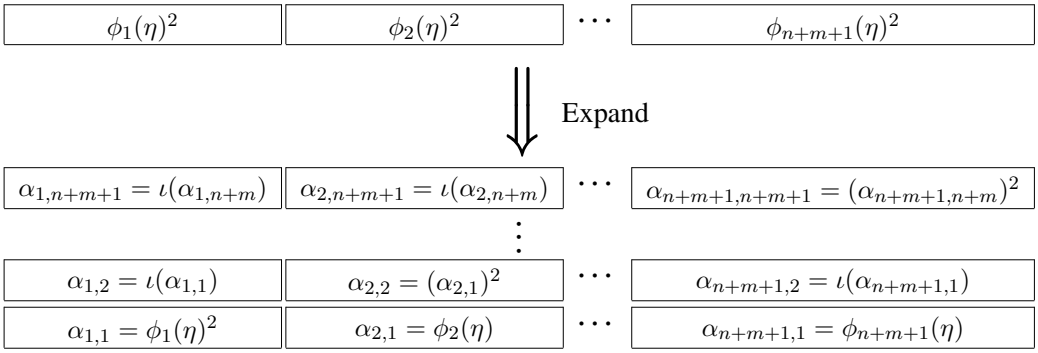

Figure 1: A layer with square activation functions is equivalent to multiple layers with only a single square activation function in each layer. The other neurons use the identity activation function, denoted ι.

Remark 4.12 *It is possible to construct shallower networks analogous to $\widetilde{N}$. This is because the proof of Proposition 4.6 actually uses many of the network's neurons to approximate the identity, so the identity activation functions of $\widetilde{N}$ may be used directly.*

Remark 4.13 *That ρ is polynomial is never really used in the proof of Proposition 4.11. It is simply that a certain amount of differentiability is required, and all such nonpolynomial functions are already covered by Proposition 4.9, as a nonzero second derivative at α implies a nonzero first derivative somewhere close to α. Thus in principle this provides another possible construction by which certain networks may be shown to exhibit universal approximation.*

Given both Proposition 4.9 and Proposition 4.11, then Theorem 3.2 follows immediately.

4.4. Extensions of these results to other special cases

We begin by showing how the Register Model may also be used to handle the case of certain nowhere-differentiable activation functions. Our manner of proof may be extended to other non-differentiable activation functions as well.

Lemma 4.14 *Let $w\colon \mathbb{R} \to \mathbb{R}$ be any bounded continuous nowhere differentiable function. Let $\rho(x) = \sin(x) + w(x)\mathrm{e}^{-x}$. Let $L \subseteq \mathbb{R}$ be compact. Then a single enhanced neuron with activation function ρ may uniformly approximate the identity function $\iota\colon \mathbb{R} \to \mathbb{R}$ on L, with arbitrarily small error.*

Proof For $h \in \mathbb{R} \setminus \{0\}$ and $A \in 2\pi\mathbb{N}$, let $\phi_{h,A}(x) = hx + A$, and let $\psi(x) = x/h$. Let

$$\iota_{h,A} = \psi_h \circ \rho \circ \phi_{h,A},$$

which is of the form that an enhanced neuron can represent. Then jointly taking h small enough and A large enough it is clear that $\iota_{h,A}$ may be taken uniformly close to ι on L. ∎

Proposition 4.15 *Let $w\colon \mathbb{R} \to \mathbb{R}$ be any bounded continuous nowhere differentiable function. Let $\rho(x) = \sin(x) + w(x)\mathrm{e}^{-x}$, which will also be nowhere differentiable. Let $K \subseteq \mathbb{R}^n$ be compact. Then $\mathcal{NN}^{\rho}_{n,m,n+m+1}$ is dense in $C(K;\mathbb{R}^m)$ with respect to the uniform norm.*

Proof As the proof of Proposition 4.9, except substituting Lemma 4.14 for Lemma 4.1. ■

Next, we discuss how we may establish a result over a noncompact domain, by exploiting the nice properties of the ReLU.

Theorem 4.16 *Let ρ be the ReLU. Let $p \in [1, \infty)$. Then $\mathcal{NN}^{\rho}_{n,m,n+m+1}$ is dense in $L^p(\mathbb{R}^n; \mathbb{R}^m)$ with respect to the usual L^p norm.*

The full proof is deferred to Appendix B due to space, but may be sketched as follows. Given some $f \in L^p(\mathbb{R}^n; \mathbb{R}^m)$, choose a compact set $K \subseteq \mathbb{R}^n$ on which f places most of its mass, and find a neural approximation to f on K in the manner of Proposition 4.9. Once this is done, a cut-off function is applied outside the set, so that the network takes the value zero in $\mathbb{R}^n \setminus K$. The interesting bit is finding a neural representation of such cut-off behaviour.

In particular the usual thing to do – multiply by a cut-off function – does not appear to have a suitable neural representation, as merely approximating the multiplication operation is not necessarily enough on an unbounded domain. Instead, the strategy is to take a maximum and a minimum with multiples of the cut-off function, which may be performed exactly.

Moving on, our final result is that Proposition 4.11 may be improved upon, provided the activation function satisfies a certain condition.

Proposition 4.17 *Let $\rho\colon \mathbb{R} \to \mathbb{R}$ be any polynomial for which there exists a point $\alpha \in \mathbb{R}$ such that $\rho'(\alpha) = 0$ and $\rho''(\alpha) \neq 0$. Let $K \subseteq \mathbb{R}^n$ be compact. Then $\mathcal{NN}^{\rho}_{n,m,n+m+1}$ is dense in $C(K; \mathbb{R}^m)$ with respect to the uniform norm.*

The proof is similar to that of Proposition 4.11, so it is deferred to Appendix C. Together with Proposition 4.9, this means that 'most' activation functions require a width of only $n + m + 1$.

5. Conclusion

There is a large literature on theoretical properties of neural networks, but much of it deals only with the ReLU.[1] However how to select an activation function remains a poorly understood topic, and many other options have been proposed: leaky ReLU, PReLU, RRelu, ELU, SELU and other more exotic activation functions as well.[2]

Our central contribution is to provide results for universal approximation using general activation functions (Theorem 3.2 and Propositions 4.9 and 4.11). In contrast to previous work, these results do not rely on the nice properties of the ReLU, and in particular do not rely on its explicit description. The techniques we use are straightforward, and robust enough to handle even the pathological case of nowhere-differentiable activation functions (Proposition 4.15).

We also consider approximation in L^p norm (Remark 4.10), and generalise previous work to smaller widths, multiple output neurons, and $p \geqslant 1$ in place of $p = 1$ (Theorem 4.16).

In contrast to much previous work, every result we show also handles the general case of multiple output neurons.

1. See for example Hanin and Sellke (2017); Petersen and Voigtlaender (2018); Gühring et al.; Daubechies et al. (2019); Arora et al. (2018); Yun et al. (2019); Chen et al. (2019a); Williams et al. (2019); Elbrächter et al. (2019).
2. See Maas et al. (2013); He et al. (2015); Xu et al. (2015); Clevert et al. (2016); Klambauer et al. (2017); Krizhevsky (2012); Ramachandran et al. (2017); Chen et al. (2019b); Boullé et al. (2020); Molina et al. (2020) respectively.

Acknowledgments

This work was supported by the Engineering and Physical Sciences Research Council [EP/L015811/1].

References

R. Arora, A. Basu, P. Mianjy, and A. Mukherjee. Understanding Deep Neural Networks with Rectified Linear Units. In *International Conference on Learning Representations*, 2018.

H.-P. Beise, S. D. Da Cruz, and U. Schröder. On decision regions of narrow deep neural networks. *CoRR*, arXiv:1807.01194, 2018.

N. Boullé, Y. Nakatsukasa, and A. Townsend. Rational neural networks. arXiv:2004.01902, 2020.

M. Chen, H. Jiang, W. Liao, and T. Zhao. Efficient Approximation of Deep ReLU Networks for Functions on Low Dimensional Manifolds. In *Advances in Neural Information Processing Systems 32*, pages 8174–8184. Curran Associates, Inc., 2019a.

R. T. Q. Chen, J. Behrmann, D. K. Duvenaud, and J.-H. Jacobsen. Residual Flows for Invertible Generative Modeling. In *Advances in Neural Information Processing Systems 32*, pages 9916–9926. Curran Associates, Inc., 2019b.

D.-A. Clevert, T. Unterthiner, and S. Hochreiter. Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs). In *International Conference on Learning Representations*, 2016.

G. Cybenko. Approximation by superpositions of a sigmoidal function. *Math. Control Signals Syst.*, 2(4):303–314, 1989.

I. Daubechies, R. DeVore, S. Foucart, B. Hanin, and G. Petrova. Nonlinear Approximation and (Deep) ReLU Networks. arXiv:1905.02199, 2019.

D. M. Elbrächter, J. Berner, and P. Grohs. How degenerate is the parametrization of neural networks with the ReLU activation function? In *Advances in Neural Information Processing Systems 32*, pages 7790–7801. Curran Associates, Inc., 2019.

I. Gühring, G. Kutyniok, and P. Petersen. Error bounds for approximations with deep ReLU neural networks in $W^{s,p}$ norms. *Proceedings of the AMS*. In press.

B. Hanin and M. Sellke. Approximating Continuous Functions by ReLU Nets of Minimal Width. arXiv:1710.11278, 2017.

K. He, X. Zhang, S. Ren, and J. Sun. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. In *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, ICCV '15, pages 1026–1034, Washington, DC, USA, 2015. IEEE Computer Society.

K. Hornik. Approximation capabilities of multilayer feedforward networks. *Neural Netw.*, 4(2): 251–257, 1991.

J. Johnson. Deep, Skinny Neural Networks are not Universal Approximators. In *International Conference on Learning Representations*, 2019.

G. Klambauer, T. Unterthiner, A. Mayr, and S. Hochreiter. Self-normalizing neural networks. In *Advances in Neural Information Processing Systems 30*, pages 971–980. Curran Associates, Inc., 2017.

A. Krizhevsky. Convolutional Deep Belief Networks on CIFAR-10. 2012.

N. Le Roux and Y. Bengio. Deep Belief Networks are Compact Universal Approximators. *Neural Comput.*, 22(8):2192–2207, 2010.

H. Lin and S. Jegelka. ResNet with one-neuron hidden layers is a Universal Approximator. In *Advances in Neural Information Processing Systems 31*, pages 6169–6178. Curran Associates, Inc., 2018.

Z. Lu, H. Pu, F. Wang, Z. Hu, and L. Wang. The Expressive Power of Neural Networks: A View from the Width. In *Advances in Neural Information Processing Systems 30*, pages 6231–6239. Curran Associates, Inc., 2017.

A. Maas, A. Hannun, and A. Ng. Rectifier Nonlinearities Improve Neural Network Acoustic Models. In *International Conference on Learning Representations*, 2013.

A. Molina, P. Schramowski, and K. Kersting. Padé Activation Units: End-to-end Learning of Flexible Activation Functions in Deep Networks. In *International Conference on Learning Representations*, 2020.

G. F. Montúfar. Universal Approximation Depth and Errors of Narrow Belief Networks with Discrete Units. *Neural Comput.*, 26(7):1386–1407, 2014. ISSN 0899-7667.

Q. Nguyen, M. C. Mukkamala, and M. Hein. Neural Networks Should Be Wide Enough to Learn Disconnected Decision Regions. In *Proceedings of the 35th International Conference on Machine Learning*. PMLR 80, Stockholm, Sweden, 2018.

P. Petersen and F. Voigtlaender. Optimal approximation of piecewise smooth functions using deep ReLU neural networks. *Neural Netw.*, 108:296–330, 2018.

A. Pinkus. Approximation theory of the MLP model in neural networks. *Acta Numer.*, 8:143–195, 1999.

P. Ramachandran, B. Zoph, and Q. V. Le. Searching for Activation Functions. arXiv:1710.05941, 2017.

R. Rojas. Networks of width one are universal classifiers. In *Proceedings of the International Joint Conference on Neural Networks*, volume 4, pages 3124–3127, 2003.

I. Sutskever and G. E. Hinton. Deep, Narrow Sigmoid Belief Networks Are Universal Approximators. *Neural Comput.*, 20(11):2629–2636, 2008. ISSN 0899-7667. doi: 10.1162/neco.2008.12-07-661.

L. Szymanski and B. McCane. Deep, super-narrow neural network is a universal classifier. In *The 2012 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, 2012.

F. Williams, M. Trager, D. Panozzo, C. Silva, D. Zorin, and J. Bruna. Gradient Dynamics of Shallow Univariate ReLU Networks. In *Advances in Neural Information Processing Systems 32*, pages 8378–8387. Curran Associates, Inc., 2019.

B. Xu, N. Wang, T. Chen, and M. Li. Empirical Evaluation of Rectified Activations in Convolutional Network. arXiv:1505.00853, 2015.

C. Yun, S. Sra, and A Jadbabaie. Small ReLU networks are powerful memorizers: a tight analysis of memorization capacity. In *Advances in Neural Information Processing Systems 32*, pages 15558–15569. Curran Associates, Inc., 2019.

Appendix A. Proof of the Register Model (Proposition 4.2)

First, we recall the classical Universal Approximation Theorem (Pinkus, 1999):

Theorem 1.1 *Let $\rho\colon \mathbb{R} \to \mathbb{R}$ be any continuous function. Let $\mathcal{N}_n^\rho$ represent the class of feedforward neural networks with activation function ρ, with n neurons in the input layer, one neuron in the output layer, and one hidden layer with an arbitrary number of neurons. Let $K \subseteq \mathbb{R}^n$ be compact. Then $\mathcal{N}_n^\rho$ is dense in $C(K)$ if and only if ρ is nonpolynomial.*

The Register Model is created by suitably reorganising the neurons from a collection of such shallow networks.

Proposition 4.2 (Register Model) *Let $\rho\colon \mathbb{R} \to \mathbb{R}$ be any continuous nonpolynomial function. Let $\mathcal{I}^\rho_{n,m,n+m+1}$ represent the class of neural networks with n neurons in the input layer, m neurons in the output layer, and an arbitrary number of hidden layers, each with $n+m$ neurons with the identity activation function, and one neuron with activation function ρ. Let $K \subseteq \mathbb{R}^n$ be compact. Then $\mathcal{I}^\rho_{n,m,n+m+1}$ is dense in $C(K;\mathbb{R}^m)$.*

Proof Fix $f \in C(K;\mathbb{R}^m)$. Let $f = (f_1, \dots, f_m)$. Fix $\varepsilon > 0$. By Theorem 1.1, there exist single-hidden-layer neural networks $g_1, \dots, g_m \in \mathcal{N}_n^\rho$ with activation function ρ approximating $f_1, \dots, f_m$ respectively. Each approximation is to within error ε with respect to $\|\cdot\|_\infty$ on K. Let each g_i have β_i hidden neurons. Let $\sigma_{i,j}$ represent the operation of its jth hidden neuron, for $j \in \{1, \dots, \beta_i\}$. In keeping with the idea of enhanced neurons, let each $\sigma_{i,j}$ include the affine function that comes after it in the output layer of g_i, so that $g_i = \sum_{j=1}^{\beta_i} \sigma_{i,j}$. Let $M = \sum_{i=1}^m \beta_i$.

We seek to construct a neural network $N \in \mathcal{I}^\rho_{n,m,n+m+1}$. Given input $(x_1, \dots, x_n) \in \mathbb{R}^n$, it will output $(G_1, \dots, G_m) \in \mathbb{R}^m$, such that $G_i = g_i(x_1, \dots, x_n)$ for each i. That is, it will compute all of the shallow networks $g_1, \dots, g_m$. Thus it will approximate f to within error ε with respect to $\|\cdot\|_\infty$ on K.

The construction of N is mostly easily expressed pictorially; see Figure 2. In each cell, representing a neuron, we define its value as a function of the values of the neurons in the previous layer. In every layer, all but one of the neurons uses the identity activation function $\iota\colon \mathbb{R} \to \mathbb{R}$, whilst one neuron in each layer performs a computation of the form $\sigma_{i,j}$.

The construction can be summed up as follows.

Each layer has $n + m + 1$ neurons, arranged into a group of n neurons, a group of a single neuron, and a group of m neurons.

The first n neurons in each layer simply record the input $(x_1, \ldots, x_n)$, by applying an identity activation function. We refer to these as the 'in-register neurons'.

Next we consider $g_1, \ldots, g_m$, which are all shallow networks. The neurons in the hidden layers of $g_1, \ldots, g_m$ are arranged 'vertically' in our deep network, one in each layer. This is the neuron in each layer that uses the activation function ρ. We refer to these as the 'computation neurons'. Each computation neuron performs its computation based off of the inputs preserved in the in-register neurons.

The final group of m neurons also use the identity activation function; their affine parts gradually sum up the results of the computation neurons. We refer to these as the 'out-register neurons'. The ith out-register neuron in each layer will sum up the results of the computation neurons computing $\sigma_{i,j}$ for all $j \in \{1, \ldots, \beta_i\}$.

Finally, the neurons in the output layer of the network are connected to the out-register neurons of the final hidden layer. As each of the neurons in the output layer has, as usual, the identity activation function, they will now have computed the desired results. ■

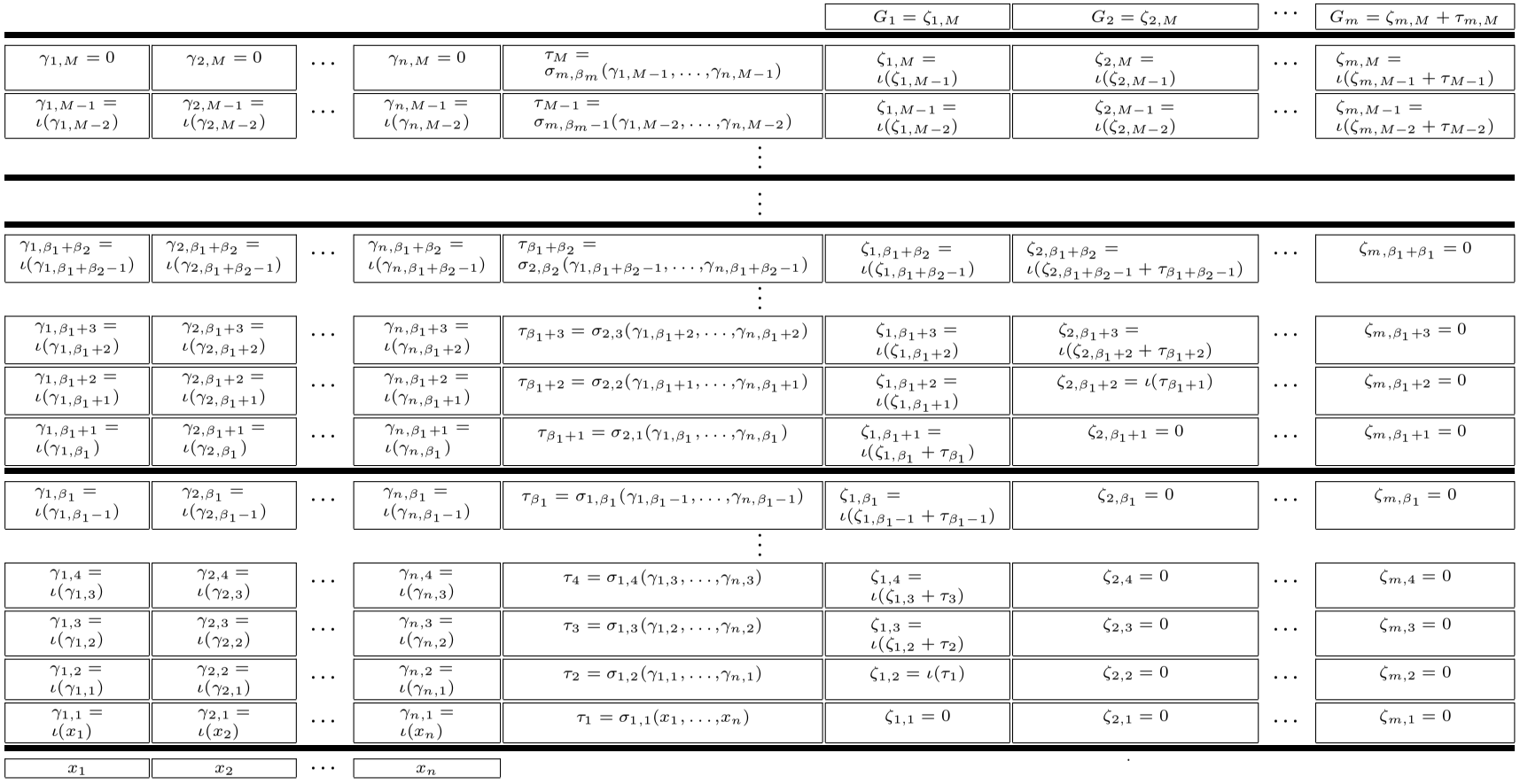

Figure 2: The thick lines delimit groups of layers; the ith group computes $\sigma_{i,1}, \ldots, \sigma_{i,\beta_i}$. The inputs to the network are $x_1, \ldots, x_n$, depicted at the bottom. The outputs from the network are $G_1, \ldots, G_m$, depicted at the top. The identity activation function $\mathbb{R} \to \mathbb{R}$ is denoted ι.

Appendix B. Proof of Theorem 4.16

Lemma B.1 *Let $a, b, c, d \in \mathbb{R}$ be such that $a < b < c < d$. Let $U_{a,b,c,d} \colon \mathbb{R} \to \mathbb{R}$ be the unique continuous piecewise affine function which is one on $[b, c]$ and zero on $(-\infty, a] \cup [d, \infty)$. Then two layers of two enhanced neurons each, with ReLU activation function, may exactly represent the function $U_{a,b,c,d}$.*

Proof Let $x \in \mathbb{R}$ be the input. Let $m_1 = 1/(b-a)$. Let $m_2 = 1/(d-c)$. Let η_1, η_2 represent the first neuron in each layer, and ζ_1, ζ_2 represent the second neuron in each layer. We assign them values as follows.

$$\begin{aligned} \eta_1 &= \max\{0, m_1(x-a)\}, & \zeta_1 &= \max\{0, m_2(x-c)\}, \\ \eta_2 &= \max\{0, 1-\eta_1\}, & \zeta_2 &= \max\{0, 1-\zeta_1\}. \end{aligned}$$

Then $U_{a,b,c,d}(x) = \zeta_2 - \eta_2$. ■

Lemma B.2 *One layer of two enhanced neurons, with ReLU activation function, may exactly represent the function $(x, y) \mapsto \min\{x, y\}$ on $[0, \infty)^2$.*

Proof Let the first neuron compute $\eta = \max\{0, x-y\}$. Let the second neuron compute $\zeta = \max\{0, x\}$. Then $\min\{x, y\} = \zeta - \eta$. ■

Theorem 4.16 *Let ρ be the ReLU. Let $p \in [1, \infty)$. Then $\mathcal{NN}^{\rho}_{n,m,n+m+1}$ is dense in $L^p(\mathbb{R}^n; \mathbb{R}^m)$ with respect to the usual L^p norm.*

Proof Let $f \in L^p(\mathbb{R}^n; \mathbb{R}^m)$ and $\varepsilon > 0$. For simplicity assume that $\mathbb{R}^m$ is endowed with the $\|\cdot\|_\infty$ norm; other norms are of course equivalent. Let $\widehat{f} = (\widehat{f}_1, \dots, \widehat{f}_m) \in C_c(\mathbb{R}^n; \mathbb{R}^m)$ be such that

$$\left\| f - \widehat{f} \right\|_p < \varepsilon/3. \tag{5}$$

Let

$$C = \sup_{x \in \mathbb{R}^n} \max_i \widehat{f}_i(x) + 1 \tag{6}$$

and

$$c = \inf_{x \in \mathbb{R}^n} \min_i \widehat{f}_i(x) - 1 \tag{7}$$

Pick $a_1, b_1, \dots, a_n, b_n \in \mathbb{R}$ such that J defined by

$$J = [a_1, b_1] \times \cdots \times [a_n, b_n]$$

is such that $\operatorname{supp} \widehat{f} \subseteq J$. Furthermore, for $\delta > 0$ that we shall fix in a moment, let

$$\begin{aligned} A_i &= a_i - \delta, \\ B_i &= b_i + \delta, \end{aligned}$$

and let K be defined by

$$K = [A_1, B_1] \times \cdots \times [A_n, B_n].$$

Fix δ small enough that

$$|K \setminus J|^{1/p} \cdot \max\{|C|, |c|\} < \frac{\varepsilon}{6}. \tag{8}$$

Let $g = (g_1, \ldots, g_m) \in \mathcal{NN}^{\rho}_{n,m,n+m+1}$ be such that

$$\sup_{x \in K} \left| \widehat{f}(x) - g(x) \right| < \min\left\{ \frac{\varepsilon}{3|J|^{1/p}}, 1 \right\}, \tag{9}$$

which exists by Proposition 4.9. Note that g is defined on all of $\mathbb{R}^n$; it simply happens to be close to $\widehat{f}$ on K. In particular it will takes values close to zero on $K \setminus J$, and may take arbitrary values in $\mathbb{R}^n \setminus K$. By equations (6), (7), (9), it is the case that

$$\begin{aligned} C &\geqslant \sup_{x \in K} \max_i g_i(x), \\ c &\leqslant \inf_{x \in K} \min_i g_i(x). \end{aligned} \tag{10}$$

Now consider the network describing g; we will modify it slightly. The goal is to create a network which takes value g on J, zero in $\mathbb{R}^n \setminus K$, and moves between these values in the interface region $K \setminus J$. Such a network will provide a suitable approximation to $\widehat{f}$.

This will be done by first constructing a function which is approximately the indicator function for J, with support in K; call such a function U. The idea then is to construct a neural representation of G_i defined by

$$G_i = \min\{\max\{g_i, cU\}, CU\}.$$

Provided $|K \setminus J|$ is small enough then $G = (G_1, \ldots, G_m)$ will be the desired approximation; this is proved this below.

We move on to presenting the neural representation of this construction.

First we observe that because the activation function is the ReLU, then the identity approximations used in the proof of Proposition 4.9 may in fact exactly represent the identity function on some compact set: $x \mapsto \max\{0, x + N\} - N$ is exactly the identity function, for suitably large N, and is of the form that an enhanced neuron may represent. This observation isn't strictly necessary for the proof, but it does simplify the presentation somewhat, as the values preserved in the in-register neurons of g are now exactly the inputs $x = (x_1, \ldots, x_n)$ for $x \in K$. For sufficiently negative x_i, outside of K, they will take the value $-N$ instead, but by insisting that is N sufficiently large that

$$-N < \min_{i \in \{1, \ldots, n\}} A_i, \tag{11}$$

then this will not be an issue for the proof.

So take the network representing g, and remove the output layer. (If the output layer is performing any affine transformations then treat them as being part of the final hidden layer, in the manner of enhanced neurons. Thus the output layer that is being removed is just applying the identity function to the out-register neurons.) Some more hidden layers will be placed on top, and then a new output layer will be placed on top. In the following description, all neurons not otherwise specified will be performing the identity function, so as to preserve the values of the corresponding neurons in the preceding layer. As all functions involved are continuous and K is compact, and compactness is preserved by continuous functions, and continuous functions are bounded on compact sets, then this is possible for all $x \in K$ by taking N large enough.

The first task is to modify the value stored in the x_1-in-register neuron. At present it stores the value x_1; by using the x_1-in-register neuron and the computation neuron in two extra layers, its value may be replaced with $U_{A_1,a_1,b_1,B_1}(x_1)$, via Lemma B.1. Place another two layers on top, and use them to replace the value of x_2 in the x_2-in-register neuron with $U_{A_2,a_2,b_2,B_2}(x_2)$, and so on. The in-register neurons now store the values

$$U_{A_1,a_1,b_1,B_1}(x_1), \ldots, U_{A_n,a_n,b_n,B_n}(x_n).$$

Once this is complete, place another layer on top and use the first x_1-in-register neuron and the x_2-in-register neuron to compute the minimum of their values, in the manner of Lemma B.2, thus computing

$$\min\{U_{A_1,a_1,b_1,B_1}(x_1), U_{A_2,a_2,b_2,B_2}(x_2)\}.$$

Place another layer on top and use another two in-register neurons to compute the minimum of this value and the value presently stored in the x_3-in-register neuron, that is $U_{A_3,a_3,b_3,B_3}(x_3)$, so that

$$\min\{U_{A_1,a_1,b_1,B_1}(x_1), U_{A_2,a_2,b_2,B_2}(x_2), U_{A_3,a_3,b_3,B_3}(x_3)\}$$

has now been computed. Continue to repeat this process until the in-register neurons have computed[3]

$$U = \min_{i\in\{1,\ldots,n\}} U_{A_i,a_i,b_i,B_i}(x_i).$$

Observe how U represents an approximation to the indicator function for J, with support in K, evaluated at $(x_1, \ldots, x_n)$.

(Note how the small foible regarding how an in-register neuron would only record $-N$ instead of x_i, for $x_i < -N$, is not an issue. This is because of equation (11), which implies that $U_{A_i,a_i,b_i,B_i}(x_i) = 0 = U_{A_i,a_i,b_i,B_i}(-N)$, thus leaving the value of U unaffected.)

This is a highly destructive set of operations: the network no longer remembers the values of its inputs. Thankfully, it no longer needs them.

The out-register neurons presently store the values $g_1, \ldots, g_m$, where $g_i = g_i(x_1, \ldots, x_n)$. Now add another layer. Let the value of its out-register neurons be $\theta_1, \ldots, \theta_m$, where

$$\theta_i = \max\{0, g_i - cU\}.$$

Add one more hidden layer. Let the value of its out-register neurons be $\lambda_1, \ldots, \lambda_m$, where

$$\lambda_i = \max\{0, -\theta_i + (C - c)U\}.$$

Finally place the output layer on top. Let the value of its neurons be $G_1, \ldots, G_m$, where

$$G_i = -\lambda_i + CU.$$

Then in fact

$$G_i = \min\{\max\{g_i, cU\}, CU\} \tag{12}$$

as desired.

3. It doesn't matter which of the in-register neurons records the value of U.

All that remains to show is that $G = (G_1, \ldots, G_m)$ of this form is indeed a suitable approximation. First, as G and g coincide in J, and by equation (9),

$$\begin{aligned}\left(\int_J \left|\widehat{f}(x) - G(x)\right|^p \mathrm{d}x\right)^{1/p} &\leqslant |J|^{1/p} \sup_{x \in J} \left|\widehat{f}(x) - G(x)\right| \\ &= |J|^{1/p} \sup_{x \in J} \left|\widehat{f}(x) - g(x)\right| \\ &< \frac{\varepsilon}{3}. \end{aligned} \tag{13}$$

Secondly, by equations (6), (7), (10), (12) and then equation (8),

$$\begin{aligned}\left(\int_{K \setminus J} \left|\widehat{f}(x) - G(x)\right|^p \mathrm{d}x\right)^{1/p} &\leqslant |K \setminus J|^{1/p} \sup_{x \in K \setminus J} \left|\widehat{f}(x) - G(x)\right| \\ &\leqslant |K \setminus J|^{1/p} \cdot 2 \max\{|C|, |c|\} \\ &< \frac{\varepsilon}{3}. \end{aligned} \tag{14}$$

Thirdly,

$$\left(\int_{\mathbb{R}^n \setminus K} \left|\widehat{f}(x) - G(x)\right|^p \mathrm{d}x\right)^{1/p} = 0, \tag{15}$$

as both $\widehat{f}$ and G have support in K.

So by equations (5), (13), (14) and (15),

$$\|f - G\|_p \leqslant \left\|f - \widehat{f}\right\|_p + \left\|\widehat{f} - G\right\|_p < \varepsilon$$

■

Appendix C. Proof of Proposition 4.17

Proposition 4.17 *Let $\rho\colon \mathbb{R} \to \mathbb{R}$ be any polynomial for which there exists a point $\alpha \in \mathbb{R}$ such that $\rho'(\alpha) = 0$ and $\rho''(\alpha) \neq 0$. Let $K \subseteq \mathbb{R}^n$ be compact. Then $\mathcal{NN}^{\rho}_{n,m,n+m+1}$ is dense in $C(K; \mathbb{R}^m)$ with respect to the uniform norm.*

Proof Let $h \in \mathbb{R} \setminus \{0\}$. Define $\rho_h\colon \mathbb{R} \to \mathbb{R}$ by

$$\rho_h(x) = \frac{\rho(\alpha + hx) - \rho(\alpha)}{h^2 \rho''(\alpha)/2}.$$

Then, taking a Taylor explansion around α,

$$\begin{aligned}\rho_h(x) &= \frac{\rho(\alpha) + hx\rho'(\alpha) + h^2x^2\rho''(\alpha)/2 + \mathcal{O}(h^3x^3) - \rho(\alpha)}{h^2\rho''(\alpha)/2} \\ &= x^2 + \mathcal{O}(hx^3).\end{aligned}$$

Let $s(x) = x^2$. Then $\rho_h \to s$ uniformly over any compact set as $h \to 0$.

Now set up a network as in the Square Model (Proposition 4.6), with every neuron using the square activation function. Call this network N. Create a network N_h by copying N and giving every neuron in the network the activation function ρ_h instead.

Uniform continuity preserves uniform convergence, compactness is preserved by continuous functions, and a composition of two uniformly convergent sequences of functions with uniformly continuous limits is again uniformly convergent. Thus the difference between N and N_h, with respect to $\|\cdot\|_\infty$ on K, may be taken arbitrarily small by taking h arbitrarily small.

Furthermore note that ρ_h is just ρ pre- and post-composed with affine functions. (Note that there is only one term in the definition of $\rho_h(x)$ which depends on x.) This means that any network which may be represented with activation function ρ_h may be precisely represented with activation function ρ, by combining the affine transformations involved. ■