A REPORT ON "THE EXPRESSIVE POWER OF NEURAL NETWORKS: A VIEW FROM THE WIDTH"

S&DS 670

NOVEMBER 1, 2017

DAVID BRANDFONBRENER

1. Introduction

In a recent paper that has been accepted at NIPS [3], Lu et al. present some new results about deep neural networks with bounded width and ReLU activations. Their central question is whether a network with fixed width (relatively near the input dimension) and unbounded depth can achieve the same results as a network with fixed depth and unbounded width. They formalize their response to this question into four main theorems and one open question to which they provide some experimental evidence.

Here we will summarize and critique their findings. We will offer a simplified proof of an improved version of their Theorem 1 in section 2. Section 3 explains Theorems 2 and 3 and critiques their underlying assumptions. Section 4 examines Theorem 4 and formalizes the related question posed in the paper. Section 5 questions the experimental setup used in the paper and recreates the experiments. Section 6 concludes.

2. Simplifying and Improving Theorem 1

Theorem 1 in Lu et al. is as follows: For any Lebesgue-integrable function $f : \mathbb{R}^d \to \mathbb{R}$ and any $\epsilon > 0$, there exists a fully-connected ReLU network $\mathcal{A}$ with width $w \leq d + 4$ such that the function $F_{\mathcal{A}}$ represented by this network satisfies

$$\int_{\mathbb{R}^d} |f(x) - F_{\mathcal{A}}(x)| dx < \epsilon.$$

One strange part of the theorem is the fixed width at $d + 4$. The 4 is non-optimal and I will show below that we can achieve the result with 2 instead.

The next thing to note about this theorem is that it differs in its setting from more traditional approximation results by using L^1 norm over all of $\mathbb{R}^d$. This is an unusual setting because it is unrealistic. Neural networks are used on real data that cannot realistically assume values over all of $\mathbb{R}^n$ with equal likelihood. It makes much more sense to follow the traditional paradigm of working over a domain with finite measure. This usually corresponds to R^d under some probability measure μ (e.g. $[0, 1]^d$ with Lebesgue measure).

Lu et al. go on to prove this result with a lengthy construction in the appendix. Their idea is to construct blocks of width $d+4$ and depth $4d+1$ that can closely approximate an indicator function of a d-dimensional cube. The cubes are constructed by smaller networks such that stacking them into one deeper network sums their outputs. Then, since summing over many such indicators can approximate any L^1 function, the result follows. The construction gets quite messy and is unnecessary to do from scratch as there is previous work on similar subjects.

The main issues with this theorem are (1) the $d + 4$ is not optimal, (2) the setting is unorthodox, and (3) the proof is long and inelegant. Below I will solve these issues by providing a simpler proof of the result in all of the usual settings with $d + 2$ as the bound on the width rather than $d + 4$.

The following result is due to [1] and uses the results of [2]. This requires one lemma, and then a slightly different (but more realistic) statement of the theorem. Let $[\cdot]_+$ represent the positive part or "ReLU" function.

Lemma 2.1. [1] *Let $f : \mathbb{R}^d \to \mathbb{R}$ be a function computed by a ReLU network with one hidden layer of width m. So, we have that $f(\vec{x}) = \sum_{j=1}^{m} c_j \left[\vec{a}_j \cdot \vec{x} + b_j\right]_+$, where for each j we have $\vec{a}_j \in \mathbb{R}^d$ and $b_j, c_j \in \mathbb{R}$. Then, there exists a ReLU network with $m+1$ hidden layers of width $d+2$ that computes f.*

Proof. Let $T > 0$ such that $T + \sum_{j=1}^{k} c_j \left[\vec{a}_j \cdot \vec{x} + b_j\right]_+ > 0$ for all $k = 1, \ldots, m$ and $T + x_i > 0$ for all $i = 1, \ldots, d$. This value will allow us to avoid having inputs mapped to 0 by the repeated application of ReLU functions.

Now, note that for each j we can define $d_j \in \mathbb{R}$ such that $\vec{a}_j \cdot (\vec{x} + T\vec{1}) + d_j = \vec{a}_j \cdot \vec{x} + b_j$, just by distributing the dot product over the subtraction and letting $d_j = b_j - \vec{a}_j \cdot T\vec{1}$.

Let A_j be the function calculated by the j-th layer. So we have that $A_1 : \mathbb{R}^d \to \mathbb{R}^{d+2}$, $A_j : \mathbb{R}^{d+2} \to \mathbb{R}^{d+2}$ for $j = 1, \ldots, m+1$, and the output layer is $A_{m+2} : \mathbb{R}^{d+2} \to \mathbb{R}$. Take $\vec{x} \in \mathbb{R}^d$ and $y, z \in \mathbb{R}$, then:

$$\begin{aligned}
A_1(\vec{x}) &= \left[(\vec{x} + T\vec{1}, \vec{a}_1 \cdot \vec{x} + b_1, T)\right]_+ \\
A_2(\vec{x}, y, z) &= \left[(\vec{x}, \vec{a}_2 \cdot \vec{x} + d_2, z + c_1 y)\right]_+ \\
&\cdots \\
A_j(\vec{x}, y, z) &= \left[(\vec{x}, \vec{a}_j \cdot \vec{x} + d_j, z + c_{j-1} y)\right]_+ \\
&\cdots \\
A_{m+1}(\vec{x}, y, z) &= \left[(\vec{x}, 0, z + c_m y)\right]_+ \\
A_{m+2}(\vec{x}, y, z) &= z - T
\end{aligned}$$

Building a network by composing these layers gives us the desired result.

Note, we are using the first d components of each layer to store the input vector $\vec{x}$ (with some correction by T in case of negative components). Then we use the $d+1$-th component to calculate the ReLU of the j-th affine transformation and the $d+2$-th to calculate the m dimensional linear combination of ReLUs one term at a time (again with correction by T). In such a manner, we reconstruct the entirety of the original wide and shallow network that calculates the function f. □

With Lemma 2.1, we can now translate any result about the approximation capability of ReLU networks with one hidden layer into a result about a ReLU network with fixed width $d+2$. An example of one possible such theorem is given below.

Theorem 2.2. *Let μ be a finite measure on $\mathbb{R}^d$ and $1 \leq p < \infty$, then the set $\mathcal{R}$ of ReLU networks of width $w \leq d+2$ is dense in $L^p(\mu)$.*

This follows directly from Lemma 2.1 and Theorem 1 in [2].[1] So, we have improved the theorem from Lu et al. and with a simpler proof.

3. Strong Assumptions in Theorems 2 and 3

The next idea in the paper of Lu et al. is to show that when the width is restricted too much, the network loses its universal approximation capabilities. They formalize this idea into two theorems.

[1]Theorem 1 in [2] gives universal approximation for networks with one hidden layer and bounded, nonconstant acitvation functions. Since $[x]_+ - [x-1]_+$ is a bounded, nonconstant function representable by a linear combination of ReLU's applied to affine functions, the universal approximation result for bounded, nonconstant activation functions also extends to ReLU activation networks.

Theorem 2 essentially states that a function f calculated by network with width w such that $w \leq d$, the input dimension, has $\int_{\mathbb{R}^d} |f(x)|dx = +\infty$ or 0 with respect to Lebesgue measure. The proof relies on induction over the depth of the network. It is shown that at each layer the region on which f is nonzero is either empty or has infinite measure. This inductive step relies on a few very technical lemmas that I will not go into here. This Theorem, like Theorem 1, operates over all of $\mathbb{R}^d$, which makes it less applicable to the traditional neural network setting.

Theorem 3 attempts to remedy this by working over $[-1, 1]^d$. However, this theorem only holds for a limited class of functions. The statement is as follows: let $f : [-1, 1]^d \to \mathbb{R}$ be a continuous function that is not constant along any direction. Then, there exists $\epsilon > 0$ such that any network $\mathcal{A}$ with width $w \leq d - 1$ representing the function $F_{\mathcal{A}}$ gives us

$$\int_{[-1,1]^d} |f(x) - F_{\mathcal{A}}(x)|dx \geq \epsilon.$$

The proof is very simple since the assumptions are so strong. Since $w \leq d - 1$, there must be a unit vector x_0 along which $F_{\mathcal{A}}$ is constant (since we drop dimension from the input to the first hidden layer). Thus, $F_{\mathcal{A}}$ cannot be f since f is not constant along any dimension. Then, the result follows from a few statements about continuous functions on compact sets.

It would be nice to have a stronger result about the maximum width network that cannot be a universal approximator. Above, we showed that with width $d+2$ we can approximate any L^p function under a finite measure. Is it possible to prove that this is the best we can do? Or can a network of width $d+1$ approximate any such function? After some reflection, these questions seem difficult to answer. So, it makes sense that Lu et al. make some pretty strong assumptions to get provable results in the direction of the above questions. These could be worthwhile directions to pursue for stronger results and a better understanding of the relationship of depth and width.

4. Formalizing an Open Question Related to Theorem 4

The final theorem in the Lu et al. paper has to do with what the authors call "width efficiency". This idea basically means if W is a wide network with width w and fixed depth, does there exists a deep network D with fixed width and depth h dependent on w that calculates the same function as W. The theorem is as follows: Let $k \geq d+4$. There exists a ReLU network W with width $w_w = 2k^2$ and depth $h_w = 2$ such that for any $b > 0$ there exists $\epsilon > 0$ such that for any ReLU network D with width $w_d \leq k^{3/2}$ and depth $h_d \leq k + 1$ and parameters bounded in $[-b, b]$ we have

$$\int_{\mathbb{R}^n} (F_W(x) - F_D(x))^2 dx \geq \epsilon.$$

The proof is to first show the $d = 1$ case and then generalize it easily by ignoring extra inputs for higher d. The authors construct a sequence of $2k^4$ points that can be approximated by the wide network W. Then, they use some pretty serious machinery from analysis to prove that no deep network of the shape of network D can approximate W on those $2k^4$ points.

The authors say that the takeaway from the theorem is that there is a polynomial lower bound on "width efficiency". This is left ambiguous in the paper, as the authors never define how to quantify the "size" of a network. This may mean that the number of parameters in D is polynomial in the number of parameters in W. Or maybe instead it means that $w_w = 2k^2$ is polynomial in $h_d = k + 2$, but then the choice of $w_d = k^{3/2}$ seems fairly arbitrary.

According to Lu et al., recent work by Telgarsky in [4] shows an exponential lower bound on "depth efficiency". This means that there exists a deep network D with depth h such that any wide network W that can compute D to arbitrary precision has width exponential in h. This leads Lu et al. to pose an open question. Since we now have a polynomial lower bound on depth, is this the best that we can do? More specifically, is there a polynomial upper bound on

depth efficiency so that every wide network can be approximated by a network with polynomial relative depth? Or could there be an exponential lower bound on width as well?

These are definitely interesting questions, because this notion of efficiency could be a way to show that depth gives a network more approximation power than width. But more precise notions of both size and efficiency must be defined so that the question is itself more clear. So, we offer a few definitions to attempt to clarify the question.

Definition 4.1. *Let $\mathcal{F}(w, h)$ represent the set of all functions that can be computed by a network with width w and depth h.*

Now we can more succinctly state what Theorem 4 is saying. Now, it states that

$$\mathcal{F}(2k^2, 2) \not\subseteq Cl\left(\mathcal{F}(k^{3/2}, k+2)\right).$$

Where we note that the notation is obscuring what norm we are using over the space of functions. Moreover, the result of [4] states that

$$\mathcal{F}(2, 2k^3+3) \not\subseteq Cl\left(\mathcal{F}(2^k, k)\right).$$

We can even see Lemma 2.1 in this context as stating that

$$\mathcal{F}(m, 1) \subseteq \mathcal{F}(d+2, m+1).$$

The question of interest now becomes which sets $\mathcal{F}(w, h)$ are included in other such sets (or their closures). It is clear that $\mathcal{F}(w, h) \subseteq \mathcal{F}(w', h')$ if $w' \geq w$ and $h' \geq h$. With this in mind, an interesting question becomes if given some $f \in F(w, h)$, which other classes of functions $\mathcal{F}(w', h')$ must contain f. The figure below gives a rough sketch about what is currently known about this.

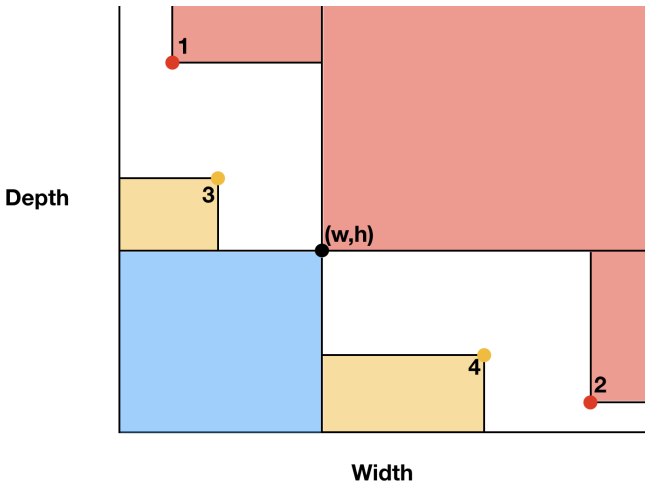

FIGURE 1. Choose any function $f \in \mathcal{F}(w, h)$. The red regions contain points (w', h') such that $f \in \mathcal{F}(w', h')$. The blue and orange sections contain points (w', h') such that there exist f with $f \notin \mathcal{F}(w', h')$. The distinction is that this property holds trivially in the blue section, but non-trivially in the orange sections (these are the proofs of Lu et al. and Telgarsky). *Point 1* and the accompanying red region are the result of the fixed width universal approximation from Theorem 1 of Lu et al (where the point is at width $d+2$). *Point 2* and the accompanying red region are the result of Hornik's fixed depth universal approximation result in [2]. *Point 3* and the accompanying orange region are the result of Theorem 4 in Lu et al. with the point at depth polynomial in w. And *Point 4* and the accompanying orange region are the result of Telgarsky [4] with the point at width exponential in h.

Note that when width increases and depth decreases or visa versa, the question of inclusion becomes difficult. These are the questions that Lu et al. are getting at with their notion of efficiency. To define efficiency, we first need a definition of the "size" of a network.

Definition 4.2. *To determine the size of a network, we will count the parameters. Let $p(w,h)$ be the number of parameters in a fully connected network of width w and depth h. Note that $p(w,h) = w^2(h-1) + w(h+d+1)$, where d is the input dimension and the output dimension is taken to be 1.*

Now we can define a more formal notion of width and depth efficiency.

Definition 4.3. *Depth Efficiency. Given a width w and depth h, and taking $h' < h$, the depth efficiency $DE_{w,h}(h')$ is the minimal ratio $\frac{p(w',h')}{p(w,h)}$ such that $\mathcal{F}(w,h) \subseteq Cl\left(\mathcal{F}(w',h')\right)$.*

The definition asks to take a deep network with width w and depth h, then defines a function $DE_{w,h}$ on depths $h' < h$ that outputs how big a network of depth h' would have to be to represent our original network. In other words, it tells us how "efficient" the depth h is compared to h'. Under this definition, the result of [4] says that $DE_{2,2k^3+3}(k) \geq \frac{p(2^k,k)}{p(2,2k^3)} = \Omega(2^{2k}/k^3)$. Similarly, we can define width efficiency.

Definition 4.4. *Width Efficiency. Given a width w and depth h, and taking $w' < w$, the depth efficiency $WE_{w,h}(w')$ is the minimal ratio $\frac{p(w',h')}{p(w,h)}$ such that $\mathcal{F}(w,h) \subseteq Cl\left(\mathcal{F}(w',h')\right)$.*

The definition asks to take a wide network with width w and depth h, then defines a function $WE_{w,h}$ on widths $w' < w$ that outputs how big a network of width w' would have to be to represent our original network. Under this definition, the result of Theorem 4 in Lu et al. says that $WE_{2k^2,2}(k^{3/2}) \geq \frac{p(k^{3/2},k+2)}{p(2k^2,2)} = \Omega(1)$. In this new light, this result is not very surprising. It just says that a narrower network of about the same size as the original network cannot represent the same functions as the original network.

So, the formal question is whether we can compute the functions WE and DE. An important insight from these notations is that the results presented in [4] and Lu et al. are not very general in nature. To define the DE function, one must fix w, h, and then these papers also choose to fix h', all three of these choices make the result less general. It makes sense why the authors have to make these choices as there is no clear relation between $DE_{w,h}$ and DE_{w_0,h_0}. But it is important to note that there could be more general results possible. This question is clearly very difficult and stating it this way is a clean and general way to understand the problem being posed.

5. Questioning the Experimental Setup

To provide support for the theory of a polynomial upper bound on "width efficiency", Lu et al. conduct some experiments on a toy dataset. They generate a random wide and shallow network and create a dataset by sampling uniformly over the domain and plugging the points into the network. This gives a dataset where the authors can use stochastic gradient descent to train a deeper but narrower net (with ratios of parameters similar to those in Theorem 4 of the paper) to attempt to produce the same network.

It is questionable what kind of evidence these experiments will be able to show. Here are some potential shortcomings to the experimental setup in the paper:

(1) It is unclear what numerical level of error will mean that the deep network could compute the shallow network to arbitrary precision. This makes it very difficult to gain any knowledge from knowing the error numbers reported in the paper. There is no comparison between different depths and sizes, so the numbers they report have no context and no meaning.

(2) Randomly generated networks say little about the existence of pathological networks. Even getting 0 error on one example may just mean that the randomly generated shallow network defined an exceptionally easy to compute function. So using random networks in the first place is a crude way to answer these general questions.
(3) Training networks on data means that we are now reliant on SGD to find the optimal network. If the error is high, we cannot know if the results we get are indicative of the deep network not being able to approximate the shallow network or SGD not being able to find the best deep network parameters.
(4) The experiments suffer from a curse of dimensionality and thus only use toy examples, with input dimension $d = 1$ or 2 and depth $h \leq 7$. These small examples may not show some of the phenomena that come up in larger (more realistic) networks.

While it is questionable what value any experiments can play in this problem, it is still worth a try to get some traction on a difficult question. One experiment that is not attempted in the Lu et al. paper is to create one wide network and then attempt to approximate it by progressively deeper and narrower networks. This approach could tell us if adding depth (but not more parameters) improves the approximation. However, we must be wary that the results will also be effected by the difficulty of training progressively deeper networks and on how complicated our random networks are. The results of such experiments and others are reported below.

I used the Keras package in python to recreate the experimental setup described by Lu et al. to perform this experiment. So, we generate a random wide network by taking random normal parameters into the network of the desired dimensions. Then, we fit progressively deeper networks with shrinking width to keep the size constant (with size defined by number of parameters $p(w, h)$ as in Section 4). The training scheme of Lu et al. is then to create a data set of $(10,000)2^d$ randomly sampled points in the domain $[-1, 1]^d$ and produce outputs from the random wide network. This data set is equally split into train and test sets. The deep networks are then trained using mean squared error as the loss function with a mini batch AdaDelta optimizer with learning rate of 1.0 for 100 epochs.

Using $d = 1$ and a wide network with 1 hidden layer and width 20, we get results in Figure 2. Here we see that the training protocol failed for the deeper networks. In fact, for the deepest and narrowest networks, the network was actually just outputting the mean of the training data for every point in the test data.

These networks clearly did not work at depths 4 and greater. It is possible that the training did not work because the network is too deep. But it is also possible that they were too narrow to satisfy the restriction that they were smaller in size than the original network. So, we can try allowing the network to be wider, but still polynomial in the input size (proportional to the size of a network with the width squared). This gives the results in Figure 5. Here, even the deepest networks trained well, since they had enough parameters to capture the activity from the original network.

Since we had been using $d = 1$, we can actually explicitly visualize these networks. Figure 4 shows the functions calculated by the shallow network, and a larger deeper network fit to that shallow network. They are not exactly the same, but fairly close. The deeper network has more small fluxuations (likely because it has more expressive power, due to its size as well as its depth).

These larger deep networks fit very well, and we did not see much effect from increasing depth. This may be because the random network we are trying to imitate is relatively easy to mimic for larger networks, regardless of their depths. To test this hypothesis, we can invert the experimental paradigm. Rather than create a random shallow network, we create a random deep network, then fit progressively shallower networks to it. Now we make the trained wide networks have size polynomial in the size of the random deep network (bounded by the size of the network with the original depth squared). The results are shown in Figure 5. Again, the errors are very low, and in fact lower than the previous experiment by about one order

depth	width	size	mean error
1	20	60	0.0007
2	6	60	0.0020
3	4	52	0.0021
4	3	45	1.473
5	3	57	0.9798
6	2	36	1.954
7	2	42	1.946
8	2	48	2.454
9	2	54	2.441
10	2	60	2.444

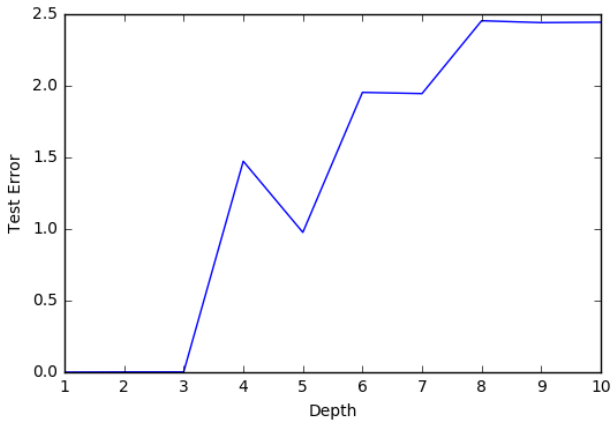

FIGURE 2. Input dimension $d = 1$ and a wide random network with 1 hidden layer and width 20. Network size is bounded by the size of the wide network. These results show the size and mean error (over 5 trials) of various deeper networks trained to mimic the wide network. The best performance comes from the shallowest network.

of magnitude. This suggests that the shallow and wide networks had an easier time fitting the narrow deep network. Mostly, I think that this shows the trouble with experimentation in this area. Even though we know that we can construct deep networks that require much much wider networks to be expressed, using random networks, this information is lost. The size of the networks seems to be more important than the structure when using this experimental setup of small random networks generating data.

My experiments run into the same computation problem as those of Lu et. al. It takes quite a while to train even these small networks for so many iterations, so I was not yet able to finish all of the experiments (or conduct as many trials) as I would have liked. In the future, I could hopefully extend these experiments to more input dimensions and different (larger) target networks. Given enough time and computational power, it may be possible to find some sort of expression for when the trained networks will not be able to approximate the random networks, but pen and paper proofs would be more conclusive.

At the end of the day, when it comes to defining inclusion of classes of neural network by depth and width, experiments are likely not so useful. These experiments confirmed this intuition by showing the issues with SGD and using random target networks. Pen and paper proofs are likely more valuable in this line of work.

6. Conclusion

The paper of Lu et al. provides an interesting perspective on how to quantify the importance of depth to the approximation power of a neural network. While some of the theorems are

depth	width	size	mean error
1	400	1200	0.00103
2	32	1152	0.00028
3	23	1173	0.00039
4	19	1197	0.00018
5	16	1136	0.00037
6	14	1092	0.00043
7	13	1131	0.00044
8	12	1182	0.00061
9	11	1089	0.00050
10	10	1020	0.00079

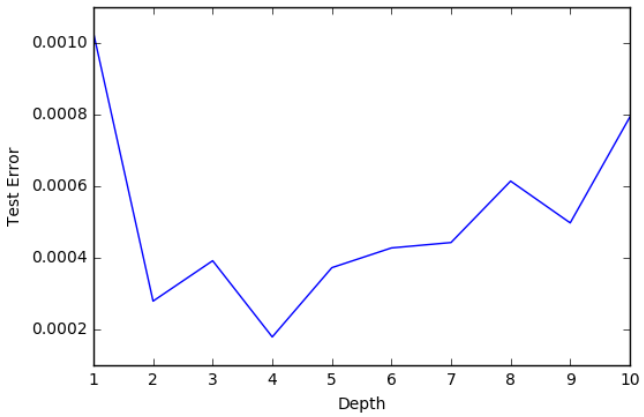

FIGURE 3. Input dimension $d = 1$ and a wide random network with 1 hidden layer and width 20. Network size is bounded by the size of the network with width $20^2 = 400$, ie. polynomial size in the original network. These results show the size and mean error (over 13 trials) of various deeper networks trained to mimic the wide network. The best performance comes from the shallowest network.

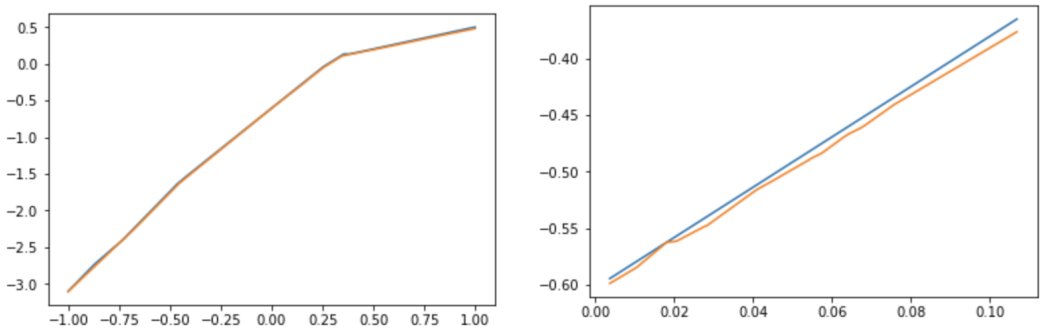

FIGURE 4. Since the input dimension is 1, we can plot the networks directly. On the left we see a randomly generated wide network (width 20, depth 1) in blue and a fitted deep network (width 16, depth 5) in orange. On the right is a zoomed in image of the same plot to better see where the networks differ.

slightly less than optimal, the paper poses interesting questions that if treated with the proper formalism could lead to some even better results on the importance of depth to neural networks as function approximators.

depth	width	size	mean error
7	10	690	0.000018
6	11	693	0.000039
5	12	660	0.000045
4	15	765	0.000019
3	18	738	0.000015
2	25	725	0.000011
1	255	765	0.000089

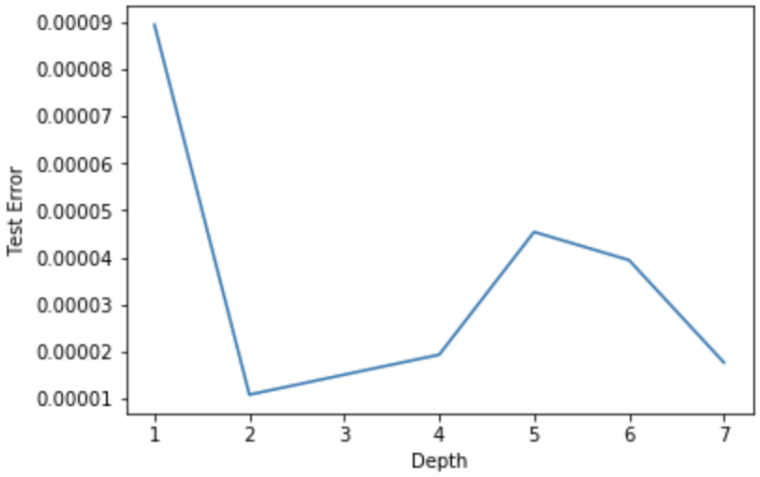

FIGURE 5. Input dimension $d = 1$ and a deep random network with 8 hidden layers and width 3. Network size is bounded by the size of the network with depth $8^2 = 64$, ie. polynomial size in the original network. These results show the size and mean error (over 5 trials) of various shallower networks trained to mimic the deep network. The best performance comes from the depth 2 network.

REFERENCES

[1] B. HANIN, *Universal function approximation by deep neural nets with bounded width and relu activations.* https://arxiv.org/abs/1708.02691, August 2017.

[2] K. HORNIK, *Approximation capabilities of multilayer feedforward networks*, Neural Networks, 4 (1991), pp. 251–257.

[3] Z. LU, H. PU, F. WANG, Z. HU, AND L. WANG, *The expressive power of neural networks: A view from the width.* https://arxiv.org/abs/1709.02540, accepted NIPS 2017, September 2017.

[4] M. TELGARSKY, *Benefits of depth in neural networks*, in JMLR: Workshop and Conference Proceedings, vol. 49, 2016, pp. 1–23.