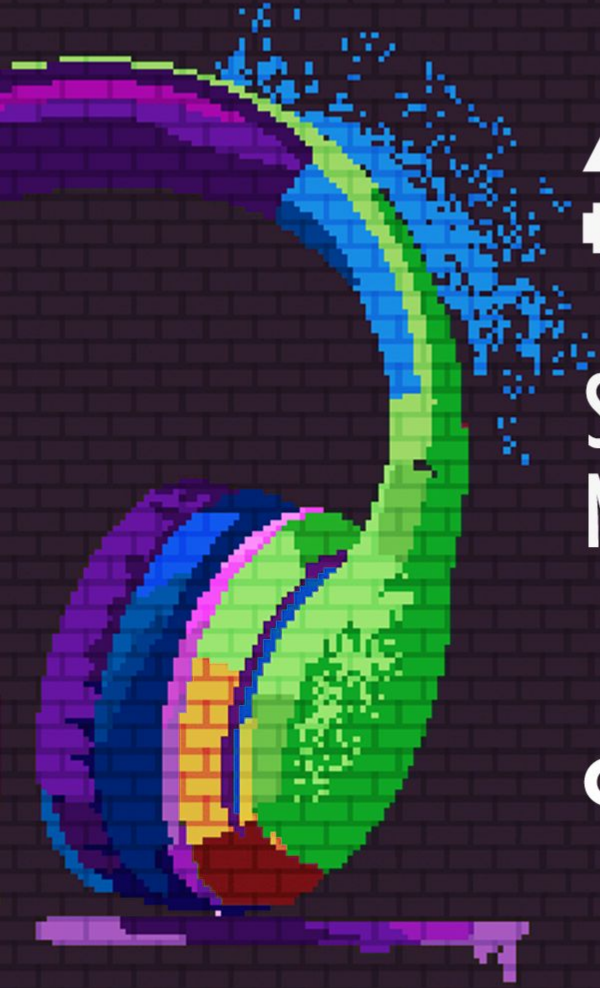


ADCx GATHER

SHRINK YOUR VA MODEL NEURAL NETWORKS!

CHRISTOPHER CLARKE

I. INTRODUCTION

Not-so-heuristic methodology to determine neural network size

What is the issue we face?

- Often, we rely on intuition to determine the size of the neural network
- Failing that, we rely on other factors such as maximum compute resource available or $\cdots$
- Heuristic methods like grid search, optimization methods like `optuna`

In these cases, there is often no time to ascertain a bound on *how much resource SHOULD be required*

time doesn't wait for your math - misquoting the famous R.Bencina article

Reiterating: We should not waste any of the hard work done in the past.

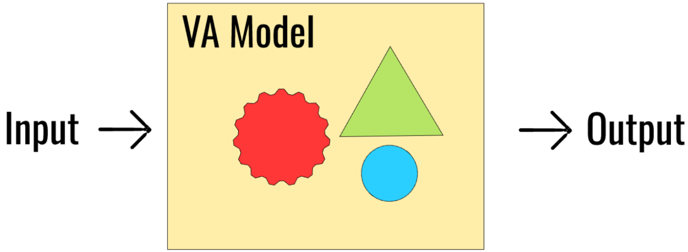
Instead, the aim is to inform our decisions with prior research or apriori knowledge

II. INSIDE/OUTSIDE OF SCOPE

What this doesn't / isn't solve(ing) [out of scope]

- Not a just-add-water one-size-fits-all solution, not an immediate solution
- Doesn't address multi-model MIMO problems
- Non-regression type problems
- PAC-learning framework

Specifically, this talk will also not address **pruning**, **distillation**, and **quantization**. **Instead** we will focus on an abstract case of a single model problem, and provide a framework for determining a methodology for your given case —subjected to the complexity of your **CPU-based** VA model



Reducing time taken with heuristic techniques

Time taken to perform grid search

- Learning a *tanh* transfer function (arbitrary)

- Linear (Dense) RELU network

- Search across 50 width and 50 layer count

- Total models to train 2500

1e4

No batching, ~~1e3~~ samples, 1e3 epochs

- Time Taken: 1179.01 seconds $\approx$ 0.014 days

No batching, 1e6 samples, 1e3 epochs

- Time Taken: 105705.32 seconds $\approx$ 1.223 days

What if we could limit our search range?

i.e Find a boundary or point that tells us the correct location

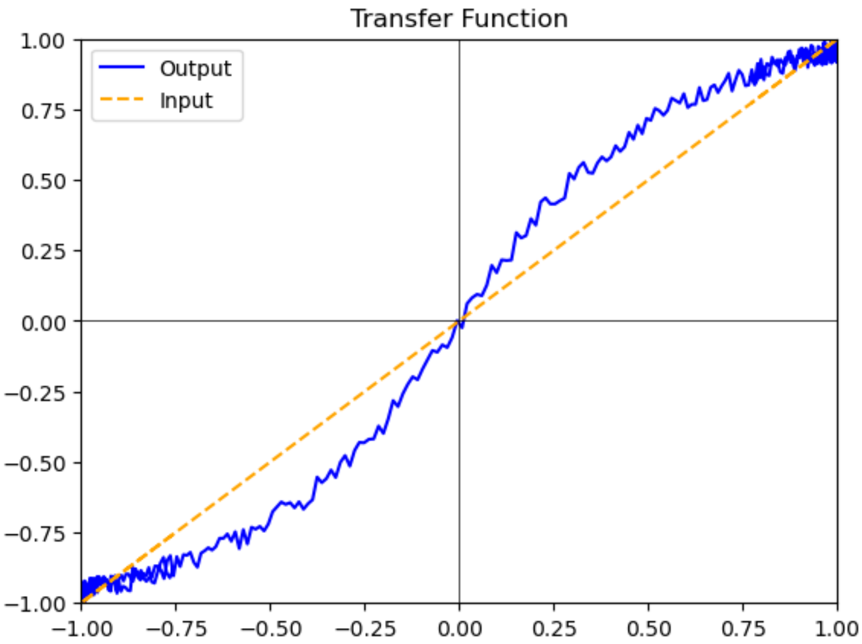
From Xu *et al.* [1] and Li and Littman [2]

Under the Probably-Approximately-Correct learning framework

Definition IV.1 (8.2)**.** (Algorithmic Alignment) Let g be a (...) function and $\mathcal{N}$ a neural network with n modules $\mathcal{N}_i$. The module functions $f_1, \ldots, f_n$ generate g for $\mathcal{N}$ if, by replacing $\mathcal{N}_i$ with f_i, the network $\mathcal{N}$ simulates g. Then $\mathcal{N}(M, \epsilon, \delta)-$algorithmically aligns with g if

- (1) $f_1, \ldots, f_n$ generate g.
- (2) there are learning algorithms $\mathcal{A}_i$ for the $\mathcal{N}_i$'s s.t

$$n \cdot \max_i C_{\mathcal{A}_i}(f_i, \epsilon, \delta) \leq M$$



Example Transfer Function

$$y[i] = \sum_{k=1}^{4} \frac{1}{2^{k-1}} \cdot \tanh(k \cdot x[i]) + \mathrm{rand}()$$

- Difficult to intuit size of network

VI. PROBLEM STATEMENT

Given a function $f(x)$ defined on the interval $[a, b]$ and a specified approximation error $\varepsilon > 0$, we aim to define a measure of the function's complexity and relate it to the minimal neural network architecture required to approximate $f(x)$ within ε.

Let's apply some theory from the branch/field of real analysis as examples

VII. “FEATURES“ OF FUNCTION COMPLEXITY

Modulus of continuity

$$\omega_f(\delta) \text{ of function } f : [a, b] \to \mathbb{R} = \sup_{\substack{x,y \in [a,b] \\ |x-y| \leq \delta}} |f(x) - f(y)|$$

sup() is the least upper bound of a set i.e the smallest number that is $\geq$ every number in the set.

Lipschitz Continuity

f is *Lipschitz continuous* on $[a, b]$ *iff* $\exists L \geq 0$

$$|f(x) - f(y)| \leq L|x - y|, \quad \forall x, y \in [a, b]$$

Max Bounded Second Derivative

iff $\exists \frac{d^2 f(x)}{dx}$ bounded in absolute value, we define:

$$M = \max_{x \in [a,b]} |f''(x)|$$

VIII. METRIC DEFINITION

Metric. Let $f : [a, b] \to \mathbb{R}$ be a continuous function and error $\epsilon > 0$. The $\mathcal{C}(f, \epsilon)$ is defined as the $\lfloor N \in \mathbb{Z}$ s.t $\exists N$ functions $s_i : [x_{i-1}, x_i] \to \mathbb{R}$ bounded within $[a, b]$ where $\{|x_{i-1}, x_i|\}_{i=1}^{N}$ partitions $f(x)$ satisfying:

- $a = x_0 < x_1 .... < x_N = b$
- $\forall x \in \{|x_{i-1}, x_i|\}_{i=1}^{N} \Rightarrow |f(x) - s_i(x)| \leq \epsilon$

This defines a metric for us to score a given function based on our "function feature extraction"

This notion was first introduced by Kolmogorov as well as Afraimovich and Glebsky [3], but here, it is defined slightly differently in order to relate to our notion of error vs. network architecture.

IX. CHOSEN FUNCTION COMPLEXITY EXAMPLE RELATING TO PARAMETER COUNT

Under the Max Bound f'' using Taylor's theorem from Burden and Faires [4]

$$E_{(f(x)-s(\{|x_{i-1},x_i|\}_{i=1}^{N})} = \left|\frac{f''(\xi)}{2}(x-x_0)^2\right| \leq \frac{M}{2}h^2$$

$$\text{desired error } \epsilon \leq \frac{M}{2}h^2 \Rightarrow h \leq \sqrt{\frac{2\epsilon}{M}}$$

Where E is the error between the actual function and the linear approximation, $f''(\xi)$ is the f'' at some point chosen $(x-x_0)$. *iff* f'' is bounded by the max value M (above), then the error is bound $\leq \frac{M}{2}h^2$

X. LITERATURE SHOWING PARTITIONS VS. PARAM. COUNT

In the literature, linking number of partitions regions R to parameter count:

- From Pascanu *et al.* [5] and Montufar *et al.* [6]

$$R \leq \left(\prod_{i=1}^{k-1} \lfloor \frac{n_i}{n_0} \rfloor \right) \sum_{i=0}^{n_0} \binom{n_k}{i}$$

- (...) n affine pieces can be represented (...) at most $\lceil log_2(n+1) \rceil + 1$ depth —from Arora *et al.* [7]

- (...) the number of activation patterns $\mathcal{A}(F_{A_{n,k}}(\mathbb{R}^m; W)$ is (tight) upper bounded by $O(k^{mn})$ for ReLU activations from —from Raghu *et al.* [8]

XI. RESULTS FROM CHOSEN FEATURE

If we assume number of partition regions $R \propto$ number of parameters N

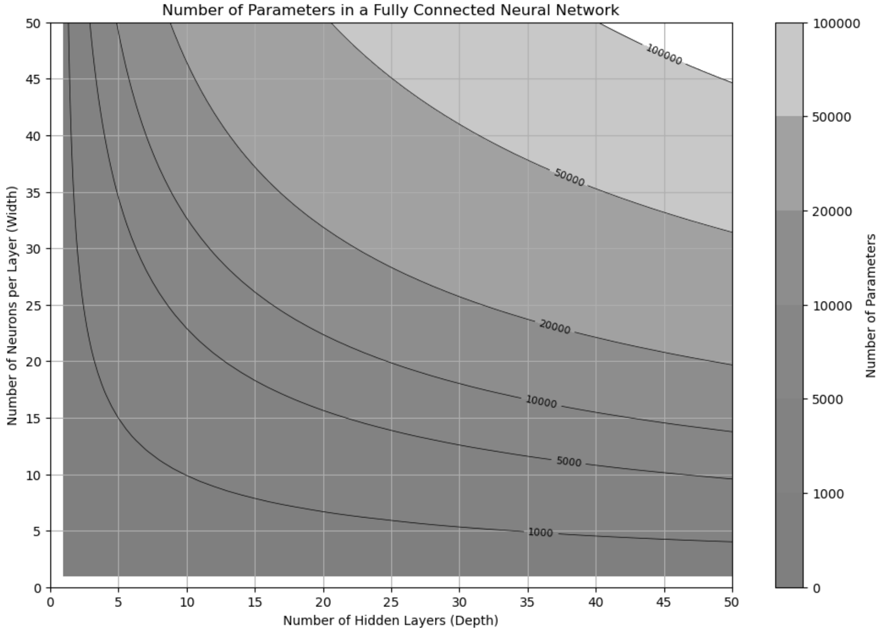
- include a (fudged) factor of 1.3 to account of overlap and non-infinite i.i.d datasets

Results

- Lipschitz Constant L: 1.9999999988425443
- Maximum Second Derivative M: 3.0792014471114486
- Modulus of Continuity $\omega_f(\delta)$: $8.333506943243284e - 05$
- Desired Error ϵ: $1e - 07$
- Computed Interval Length h: 0.000254856636253786
- Total X Range: $[-1, 1]$
- Number of Intervals N: 7848
- Estimated Number of Parameters ≈ 10202.4

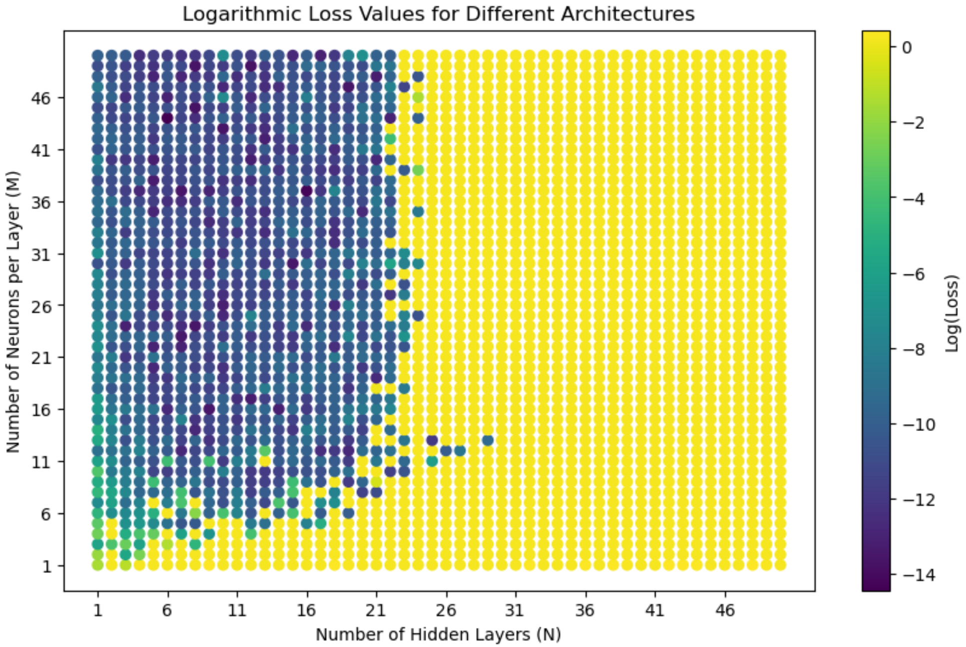
XII. PARAMETER COUNT PER RELU NETWORK SIZE

Plot of the parameter count over grid search architecture values of 50 width and 50 depth



XIII. EXPERIMENTAL RESULTS

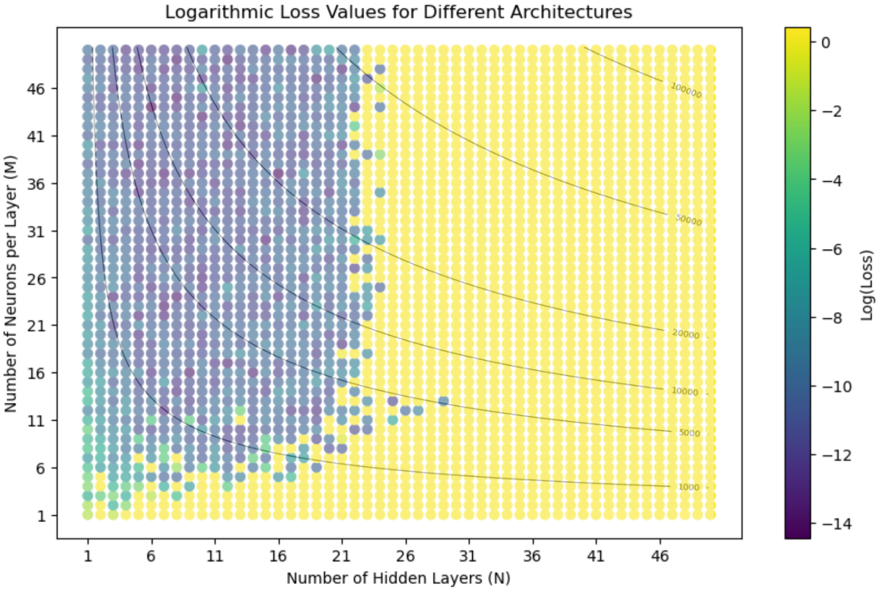
Plot of the grid search of $y = tanh(2x)$



- 1. $N = 6,\ M = 44,\ \text{Loss} = 5.380 \times 10^{-7}$, Score = 66
- 2. $N = 16,\ M = 37,\ \text{Loss} = 6.351 \times 10^{-7}$, Score = 160
- 3. $N = 12,\ M = 49,\ \text{Loss} = 1.046 \times 10^{-6}$, Score = 156
- 4. $N = 8,\ M = 45,\ \text{Loss} = 1.096 \times 10^{-6}$, Score = 96
- 5. $N = 8,\ M = 49,\ \text{Loss} = 1.172 \times 10^{-6}$, Score = 104
- 6. $N = 10,\ M = 43,\ \text{Loss} = 1.216 \times 10^{-6}$, Score = 110
- 7. $N = 19,\ M = 47,\ \text{Loss} = 1.262 \times 10^{-6}$, Score = 228
- 8. $N = 8,\ M = 24,\ \text{Loss} = 1.290 \times 10^{-6}$, Score = 48
- 9. $N = 9,\ M = 16,\ \text{Loss} = 1.315 \times 10^{-6}$, Score = 36
- 10. $N = 7,\ M = 37,\ \text{Loss} = 1.340 \times 10^{-6}$, Score = 70

XIV. TOP PERFORMING MODEL LIES WITHIN DERIVED BOUNDS

Top performing model 6 layers, 44 width = 10033 parameters



XV. SCORE EVALUATION

Satisficing equation for “Score“

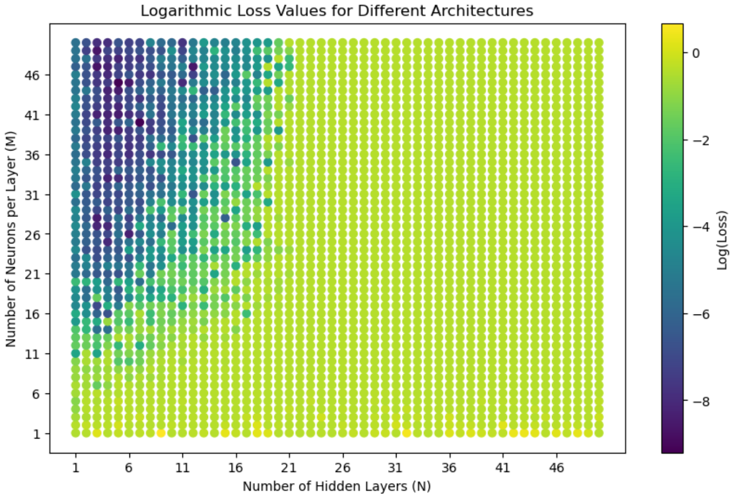
- Satisficing: constraint satisfaction, the process of finding a solution satisfying a set of constraints, without concern for finding an optimum

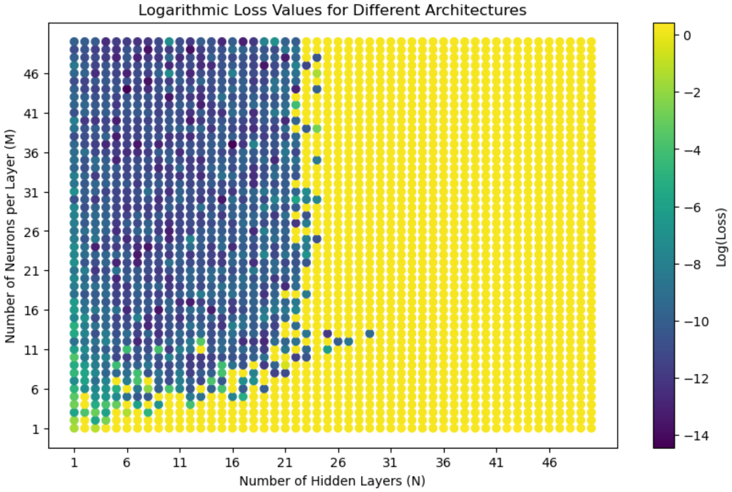
- For example: AVX SIMD (256-bit register) can hold 4 doubles

$$\arg\min_{N,M} \text{Loss}(N, M) \quad \text{subject to} \quad \text{Score} = V \times N$$

$$\text{where} \quad V = \begin{cases} \left\lfloor \frac{M}{4} \right\rfloor + 1 & \text{if } M \mod 4 \neq 0 \\ \frac{M}{4} & \text{if } M \mod 4 = 0 \end{cases}$$

XVI. DATA=CODE, NEURAL NETWORK=COMPILER, OUTPUT=FUNCTION





We can see here that the training results, given a particular dataset size changes, this is not new information. But combined with the scoring from the satisficing equations...

XVII. SCORE OF THE GRID SEARCHES

Dataset Size 1 vs Size 2, $\arg\min(\text{Score}, \text{Loss})$

- 1. $N = 7$, $M = 40$, Loss $= 1.007 \times 10^{-4}$, Score $= 70$
- 2. $N = 5$, $M = 45$, Loss $= 1.042 \times 10^{-4}$, Score $= 60$
- 3. $N = 3$, $M = 47$, Loss $= 1.702 \times 10^{-4}$, Score $= 36$
- 4. $N = 3$, $M = 49$, Loss $= 1.735 \times 10^{-4}$, Score $= 39$
- 5. $N = 6$, $M = 45$, Loss $= 1.896 \times 10^{-4}$, Score $= 72$
- 6. $N = 12$, $M = 47$, Loss $= 1.936 \times 10^{-4}$, Score $= 144$
- 7. $N = 5$, $M = 44$, Loss $= 2.085 \times 10^{-4}$, Score $= 55$
- 8. $N = 3$, $M = 28$, Loss $= 2.236 \times 10^{-4}$, Score $= 21$
- 9. $N = 5$, $M = 41$, Loss $= 2.244 \times 10^{-4}$, Score $= 55$
- 10. $N = 6$, $M = 26$, Loss $= 2.455 \times 10^{-4}$, Score $= 42$

- 1. $N = 6$, $M = 44$, Loss $= 5.380 \times 10^{-7}$, Score $= 66$
- 2. $N = 16$, $M = 37$, Loss $= 6.351 \times 10^{-7}$, Score $= 160$
- 3. $N = 12$, $M = 49$, Loss $= 1.046 \times 10^{-6}$, Score $= 156$
- 4. $N = 8$, $M = 45$, Loss $= 1.096 \times 10^{-6}$, Score $= 96$
- 5. $N = 8$, $M = 49$, Loss $= 1.172 \times 10^{-6}$, Score $= 104$
- 6. $N = 10$, $M = 43$, Loss $= 1.216 \times 10^{-6}$, Score $= 110$
- 7. $N = 19$, $M = 47$, Loss $= 1.262 \times 10^{-6}$, Score $= 228$
- 8. $N = 8$, $M = 24$, Loss $= 1.290 \times 10^{-6}$, Score $= 48$
- 9. $N = 9$, $M = 16$, Loss $= 1.315 \times 10^{-6}$, Score $= 36$
- 10. $N = 7$, $M = 37$, Loss $= 1.340 \times 10^{-6}$, Score $= 70$

XVIII. CONCLUSION

- Heuristic methods take time
- Given constraint we should allocate required resource for task
- Function Complexity as a Feature for determining network size
- Metric for this Feautre
- Real world results
- Scoring of the output architecture, designate depending on output platform

XIX. STUFF I REMOVED FROM THIS TALK

- Relationship to Sobolev Spaces and Rademacher Complexity
- Function Decomposition to reduce parameter space —Param$(f(x)) \leq$ Param$(g \circ^{-1} f(x))$
- Since CPU-based, use `RTNeural` compute times to provide scoring instead of ascribing from target SIMD register size
- Applications to non-memoryless or more complex architectures (GRU, DDSP reverb etc) —how to begin?
- Improving the bounds estimation using different circuit analysis methods

THANK YOU

github: algoravioli

II. REFERENCES

[1] K. Xu, J. Li, M. Zhang, S. S. Du, K. ichi Kawarabayashi, and S. Jegelka, in *International Conference on Learning Representations* (2020).

[2] M. Li and M. L. Littman, arXiv preprint arXiv:2008.03229 (2020).

[3] V. Afraimovich and L. Glebsky, Taiwanese Journal of Mathematics **9**, 397 (2005).

[4] R. Burden and J. Faires, *Numerical Analysis* (Cengage Learning, 2010).

[5] R. Pascanu, G. Montufar, and Y. Bengio, "On the number of response regions of deep feed forward networks with piece-wise linear activations," (2014), arXiv:1312.6098 [cs.LG].

[6] G. Montufar, R. Pascanu, K. Cho, and Y. Bengio, "On the number of linear regions of deep neural networks," (2014), arXiv:1402.1869 [stat.ML].

[7] R. Arora, A. Basu, P. Mianjy, and A. Mukherjee, "Understanding deep neural networks with rectified linear units," (2018), arXiv:1611.01491 [cs.LG].

[8] M. Raghu, B. Poole, J. Kleinberg, S. Ganguli, and J. Sohl-Dickstein, "On the expressive power of deep neural

networks," (2017), arXiv:1606.05336 [stat.ML].