Improved Bounds on Neural Complexity for Representing Piecewise Linear Functions

Kuan-Lin Chen[1], Harinath Garudadri[2], and Bhaskar D. Rao[1]

[1]Department of Electrical and Computer Engineering, [2]Qualcomm Institute
University of California, San Diego

NeurIPS 2022
November 3, 2022

Outline

The rectified linear unit and deep neural networks

- The rectified linear unit (ReLU) (Fukushima, 1980; Nair and Hinton, 2010) is the most popular nonlinearity and building block in deep neural networks (DNNs).
- ReLU DNNs are also probably the most understandable nonlinear deep models due to their ability to be "un-rectified" (Hwang and Heinecke, 2019).
- The ability to demystify ReLU DNNs via "un-rectifying ReLUs" dates back to a seminal work by Pascanu et al. in 2014.
- A ReLU DNN divides the input space into many *linear regions*.
- Bounds on the number of linear regions are studied by (Montúfar, 2017; Raghu et al., 2017; Arora et al., 2018; Serra et al., 2018; Hinz and van de Geer, 2019), just to name a few.

Continuous piecewise linear (CPWL) functions

- A neural network using rectified linear units represents a CPWL function.
- Arora et al. (2018) proved that the reverse is also true: Any CPWL function can be represented by a neural network using rectified linear units.

Question 1

How many hidden neurons are required for a ReLU network to represent a given CPWL function?

Question 2

Can we find a network representing any given CPWL function?

Bounds in prior work

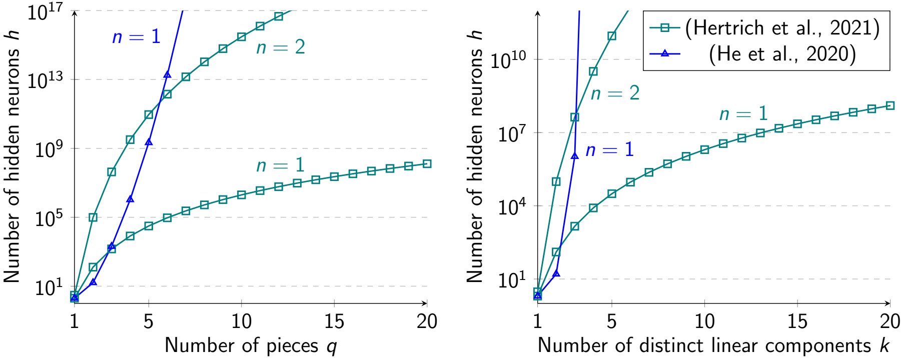

Figure: Any CPWL function $\mathbb{R}^n \rightarrow \mathbb{R}$ with q pieces or k distinct linear components can be exactly represented by a ReLU network with at most h hidden neurons. Existing bounds in the literature seem to imply the cost of representing a CPWL function in a ReLU network is extremely high.

Our work

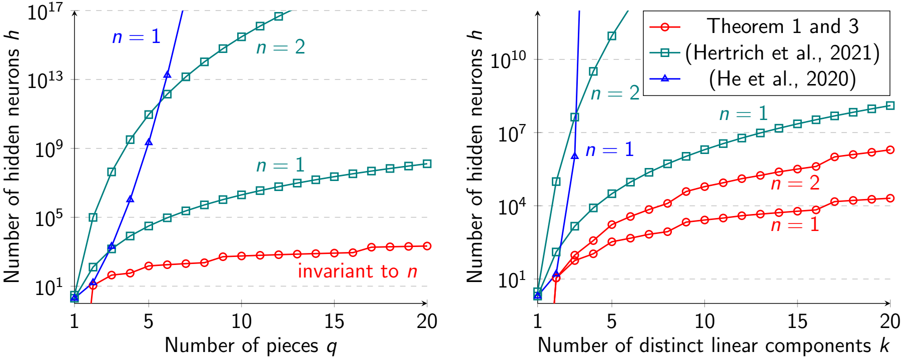

Figure: In Theorem 1 and 3, $h = 0$ when $q = 1$ or $k = 1$. The upper bounds given by Theorem 1 and 3 are substantially lower than existing bounds in the literature, implying that any CPWL function can be exactly realized by a ReLU network at a much lower cost.

Quadratic bounds

Theorem 1

Any CPWL function $p \colon \mathbb{R}^n \to \mathbb{R}$ *with* q *pieces can be represented by a ReLU network whose number of layers* l*, maximum width* w*, and number of hidden neurons* h *satisfy*

$$l \leq 2 \lceil \log_2 q \rceil + 1, \tag{1}$$

$$w \leq \mathbb{I}[q > 1] \left\lceil \frac{3q}{2} \right\rceil q, \tag{2}$$

and

$$h \leq \left(3 \cdot 2^{\lceil \log_2 q \rceil} + 2 \lceil \log_2 q \rceil - 3\right) q + 3 \cdot 2^{\lceil \log_2 q \rceil} - 2 \lceil \log_2 q \rceil - 3. \tag{3}$$

Furthermore, Algorithm 1 finds such a network in $\operatorname{poly}(n, q, L)$ *time where* L *is the number of bits required to represent every entry of the rational matrix* $\mathbf{A}_i$ *in the polyhedron representation* $\{\mathbf{x} \in \mathbb{R}^n | \mathbf{A}_i \mathbf{x} \leq \mathbf{b}_i\}$ *of the piece* $\mathcal{X}_i$ *for every* $i \in [q]$.

A polynomial time algorithm satisfying the bounds

Algorithm Find a ReLU network that computes a given CPWL function

Require: A CPWL function p with pieces $\{\mathcal{X}_i\}_{i\in[q]}$ of $\mathbb{R}^n$.
Ensure: A ReLU network g computing $g(\mathbf{x}) = p(\mathbf{x}), \forall \mathbf{x} \in \mathbb{R}^n$.

$f_1, f_2, \cdots, f_k \leftarrow$ Find all distinct linear components of p
for $i = 1, 2, \cdots, q$ **do**
  $\mathcal{A}_i \leftarrow \emptyset$
  for $j = 1, 2 \cdots, k$ **do**
    if $f_j(\mathbf{x}) \geq p(\mathbf{x}), \forall \mathbf{x} \in \mathcal{X}_i$ **then**
      $\mathcal{A}_i \leftarrow \mathcal{A}_i \bigcup \{j\}$
    end if
  end for
  $v_i \leftarrow$ A ReLU network representing the min-affine function of $\{f_m\}_{m\in\mathcal{A}_i}$
end for
$v \leftarrow$ Combine ReLU networks $v_1, v_2, \cdots, v_q$ in parallel
$u \leftarrow$ A ReLU network computing the maximum of q elements
$g \leftarrow$ A ReLU network computing the composition $u \circ v$

Bilinear bounds

Theorem 2

Any CPWL function $p\colon \mathbb{R}^n \to \mathbb{R}$ *with* k *linear components and* q *pieces can be represented by a ReLU network whose number of layers* l*, maximum width* w*, and number of hidden neurons* h *satisfy*

$$l \leq \lceil \log_2 q \rceil + \lceil \log_2 k \rceil + 1, \tag{4}$$

$$w \leq \mathbb{I}\left[k > 1\right] \left\lceil \frac{3k}{2} \right\rceil q, \tag{5}$$

and

$$h \leq \left(3 \cdot 2^{\lceil \log_2 k \rceil} + 2 \lceil \log_2 k \rceil - 3\right) q + 3 \cdot 2^{\lceil \log_2 q \rceil} - 2 \lceil \log_2 k \rceil - 3. \tag{6}$$

Furthermore, Algorithm 1 finds such a network in $\operatorname{poly}(n, k, q, L)$ *time where* L *is the number of bits required to represent every entry of the rational matrix* $\mathbf{A}_i$ *in the polyhedron representation* $\{\mathbf{x} \in \mathbb{R}^n | \mathbf{A}_i \mathbf{x} \leq \mathbf{b}_i\}$ *of the piece* $\mathcal{X}_i$ *for every* $i \in [q]$.

On the number of linear components k

Theorem 3

Any CPWL function $p\colon \mathbb{R}^n \to \mathbb{R}$ with k linear components can be represented by a ReLU network whose number of layers l, maximum width w, and number of hidden neurons h satisfy

$$l \leq \left\lceil \log_2 \phi(n,k) \right\rceil + \lceil \log_2 k \rceil + 1, \tag{7}$$

$$w \leq \mathbb{I}\left[k > 1\right] \left\lceil \frac{3k}{2} \right\rceil \phi(n,k), \tag{8}$$

and

$$h \leq \left(3 \cdot 2^{\lceil \log_2 k \rceil} + 2\lceil \log_2 k \rceil - 3\right) \phi(n,k) + 3 \cdot 2^{\lceil \log_2 \phi(n,k) \rceil} - 2\lceil \log_2 k \rceil - 3 \tag{9}$$

where

$$\phi(n,k) = \min\left(\sum_{i=0}^{n} \binom{\frac{k^2-k}{2}}{i}, k! \right). \tag{10}$$

Effect of the input dimension n

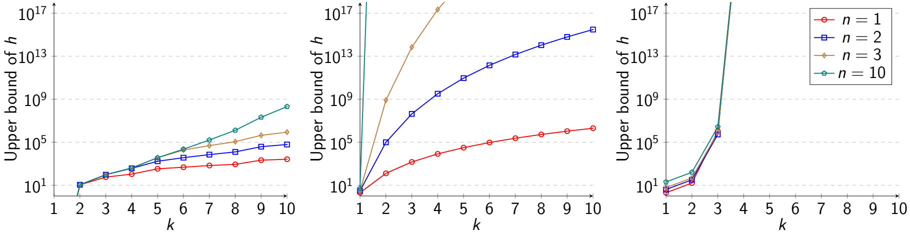

Figure: **Left**: The upper bound of h in Theorem 3 grows much slower when n grows sufficiently slower than k, leading to a much better upper bound compared to the worst-case asymptotic bound $\mathcal{O}(k \cdot k!)$ in Theorem 3. **Middle**: (Hertrich et al., 2021). **Right**: (He et al., 2020).

Open source implementation and run time of Algorithm 1

- Code is available at https://github.com/kjason/CPWL2ReLUNetwork.

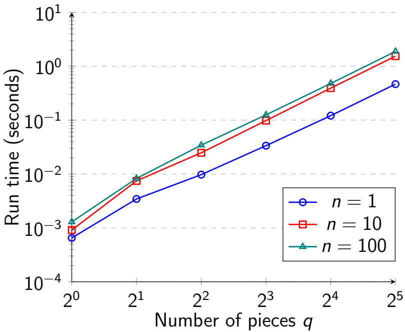

Figure: The run time of Algorithm 1 is an average of 50 trials. Every trial runs Algorithm 1 with a random CPWL function whose input dimension is n and number of pieces is q. The code provided in the above link is run on a computer (Microsoft Surface Laptop Studio) with the Intel Core i7-11370H.

References

Arora, R., Basu, A., Mianjy, P., and Mukherjee, A. (2018). Understanding deep neural networks with rectified linear units. In *International Conference on Learning Representations*.

Fukushima, K. (1980). Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36(4):193–202.

He, J., Li, L., Xu, J., and Zheng, C. (2020). ReLU deep neural networks and linear finite elements. *Journal of Computational Mathematics*, 38(3):502–527.

Hertrich, C., Basu, A., Di Summa, M., and Skutella, M. (2021). Towards lower bounds on the depth of ReLU neural networks. In *Advances in Neural Information Processing Systems*, pages 3336–3348.

Hinz, P. and van de Geer, S. (2019). A framework for the construction of upper bounds on the number of affine linear regions of relu feed-forward neural networks. *IEEE Transactions on Information Theory*, 65(11):7304–7324.

Hwang, W.-L. and Heinecke, A. (2019). Un-rectifying non-linear networks for signal representation. *IEEE Transactions on Signal Processing*, 68:196–210.

Montúfar, G. (2017). Notes on the number of linear regions of deep neural networks. In *International Conference on Sampling Theory and Applications*.

Nair, V. and Hinton, G. E. (2010). Rectified linear units improve restricted boltzmann machines. In *International Conference on Machine Learning*, pages 807–814.

Pascanu, R., Montúfar, G., and Bengio, Y. (2014). On the number of response regions of deep feed forward networks with piece-wise linear activations. *International Conference on Learning Representations*.

Raghu, M., Poole, B., Kleinberg, J., Ganguli, S., and Sohl-Dickstein, J. (2017). On the expressive power of deep neural networks. In *International Conference on Machine Learning*, pages 2847–2854. PMLR.

Serra, T., Tjandraatmadja, C., and Ramalingam, S. (2018). Bounding and counting linear regions of deep neural networks. In *International Conference on Machine Learning*, pages 4558–4566. PMLR.