Activation Region Complexity of Deep ReLu Networks

Justin Lewis

August 2019

1 Introduction

These notes are meant to give a brief exposition to the complexity of *activation regions* for deep ReLu networks. An *activation region* is simply a subset of the networks input space (possibly empty) which achieves a particular binary configuration of the neurons. This will be formalized below. The main points of the notes:

- We consider the worst case *activation region* complexity and provide a simple construction which nearly achieves a naive upper bound
- We point to other work which considers the average *activation region* complexity when considering networks at random initialization.
- We list questions / open problems.

2 Basic Notation

- Integer subscript for network parameters is used to denote layer number. Ex: W_i, b_i, w_i
- Integer superscript is used to denote a further subdivision. Ex: $w_i^{[j]}$ is jth column of matrix W_i
- $[i]$ and $[i, j]$ notation is used to select an element from an object.
- Generally, indexing is assumed to start from 1.

3 Deep ReLu Networks

To begin with, we introduce the structure of a standard deep neural network (DNN) and provide some notation. Simply, a DNN is some function $\mathcal{N}(x)$ which is formed by repeated composition of a network layer function. The

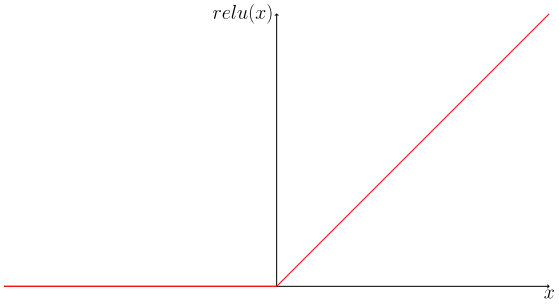

Figure 1: The ReLu activation function

layer function compromises of a linear transformation followed by a nonlinear transformation, $\sigma_i(x) : \mathbb{R}^{d_i} \mapsto \mathbb{R}^{d_i}$. Thus:

$$\mathcal{N}(x) = W_n^T \sigma_{n-1}(\cdots \sigma_2(W_2^T \sigma_1(W_1^T x + b_1) + b_2) \cdots) + b_n \tag{1}$$

$$x \in \mathbb{R}^{d_0 \times 1} \tag{2}$$

$$W_i \in \mathbb{R}^{d_{i-1} \times d_i} \tag{3}$$

$$b_i \in \mathbb{R}^{d_i \times 1} \tag{4}$$

As is apparent, each layer i of the network takes a vector of dimension d_{i-1} and maps it to d_i. For further reference, let us denote $\mathcal{N}_i(x)$ as the network function up to layer i [e.g: $\mathcal{N}_1(x) = \sigma_1(W_1^T x + b_1)$]. Typically, the nonlinear function $\sigma(x)$ applies some activation function $a(x) : \mathbb{R} \mapsto \mathbb{R}$ to its input x, elementwise. Thus:

$$\sigma_i(x)[j] = a(x[j]) \quad \forall\, i, j \tag{5}$$

This elementwise application of an activation function is where the analogy of a neuron comes from. Each element of the output of a layer $\{\mathcal{N}_i(x)[j] \ \forall\, i, j\}$ is considered a neuron; it accumulates signals from previous neurons and "fires" if the accumulation is strong enough. Let us denote the jth neuron of the ith layer as $n[i, j] = \mathcal{N}_i(x)[j]$ There are many common choices for the activation function $a(x)$ such as the sigmoid, tanh, and ReLu functions. In these notes we will focus on networks using the ReLu activation:

$$a(x) = relu(x) = max(x, 0) \tag{6}$$

4 Network Activation Regions

Here we consider a measure of network complexity via the number of unique *activation regions*. To define such a region, first define a list of binary vectors

$C_{\mathcal{N}}(x)$. The list is ordered by network layer with each element being a binary vector of that layer's neuron states. More specifically:

$$C_{\mathcal{N}}(x)[i,j] = \begin{cases} 1, & \text{if } n[i,j] > 0 \\ 0, & \text{if } n[i,j] \leq 0 \end{cases} \tag{7}$$

Given a particular configuration of the neurons c, an *activation region* $R(c, \mathcal{N})$ of the network $\mathcal{N}(x)$ is defined as:

$$R(c, \mathcal{N}) = \{x \,|\, C_{\mathcal{N}}(x) = c\} \tag{8}$$

In other words, this region R is the set of all x which can achieve the particular configuration of the neurons c. Now, we will prove some properties concerning the activation regions of ReLu networks. First, the following:

Claim 4.1. *$R(c, \mathcal{N})$ forms a polytope (possibly empty)*

Proof of Claim 4.1.1. *The constraints which define R for configuration c will be constructed layer by layer. For the first layer, the binary state of each neuron $n[1,j]\ \forall\, j$ defines a halfspace in $\mathbb{R}^d$. This follows as the first layer neuron states are governed by a single linear transformation ($w^{[j]}$ denotes jth column):*

$$n[1,j] \gtreqless 0 \Leftrightarrow \langle w_1^{[j]}, x \rangle + b_1 \gtreqless 0 \tag{9}$$

This is apparent. To continue for subsequent layers, we make two observations. First, in order to satisfy the configuration c of all neurons up to layer i, the configuration of the first $i-1$ layers must hold. Thus, fix the neuron states up to layer $i-1$ according to c. Second, as the nonlinearity of the network is now fixed up to layer $i-1$, we can rewrite $f_i(x)$ as a simple one layer network.

From above, we know that the state of each neuron of this rewritten network contributes another halfspace. Thus, we have a halfspace constraint for every network neuron and the intersection of these halfspaces forms a polytope.

To flesh out the details of the above argument, we show that fixing the neuron states of a single ReLu layer according to c allows us to linearize the layer. This is done by defining a diagonal, binary valued matrix D which zeros out the rows of W^T corresponding to OFF neurons:

$$\mathcal{N}_1(x) = relu(W_1^T x) = D_1(c) W_1^T x = \tilde{W}_1 x \tag{10}$$

$$D_i(c)[j,j] = c[i,j]\ \ \forall\, i,j \tag{11}$$

So, we can extend this to linearize the network up to layer i for D_i defined as above. Below holds because a composition of linear functions is still linear:

$$\mathcal{N}_i(x) = D_i(c) W_i^T \sigma_{i-1}(\cdots \sigma_1(D_1(c) W_1^T x + D_1(c) b_1) \cdots) + D_i(c) b_i \tag{12}$$

$$= \tilde{W} x + \tilde{b} \tag{13}$$

5 Activation Region Complexity Upper Bound

One might think naively that all configuration states are achievable. This would give an upper bound of 2^{nL} regions; however, this bound is very loose as it does not consider the geometry of the problem. A better upper bound on the maximum number of *activation regions* follows from properties of hyperplane arrangements.

Lemma 5.1. *The maximum number of regions of an arrangement of n hyperplanes in $\mathbb{R}^d$ is $\sum_{k=0}^{d} \binom{n}{k}$*

Lemma 5.2. *Considering the network layerwise, a neuron of layer i can only linearly subdivide each activation region defined up to $\mathcal{N}_{i-1}(x)$.*

Claim 5.1. *The maximum number of activation regions $N_{max}(d, n, L)$ of any ReLu network $\mathcal{N}$ with input dimension d and L layers of n neurons such that $n > 2d$, satisfies $N_{max}(d, n, L) = O(n^{dL})$*

Proof of Claim 5.1.1. *We consider a series of divisions of $\mathbb{R}^d$ which may or may not be realizable by any network $\mathcal{N}$ with the assumed structural parameters. Starting at layer 1, for each neuron we have that $\{x \mid n[1, j](x) = 0\}$ is a hyperplane as mentioned in section 4. Thus for all $n \geq 2d$:*

$$N_{max}(d, n, 1) = \sum_{k=0}^{n} \binom{n}{k} = O(\binom{n}{d}) = O((\frac{ne}{d})^d)$$

which is $O(n^d)$ when considering d as a constant factor. Now at each layer i, by Lemma 5.2, one could potentially form an arrangement of n hyperplanes within each activation region formed up to layer $i-1$. Thus, repeatedly injecting an arrangement of n hyperplanes into every previous activation region for L layers gives our upper bound of $O((n^d)^L) = O(n^{dL})$

6 Nearly Maximal Network Construction

With an upper bound on N_{max}, we now seek a lower bound to N_{max} by construction. It turns out that these two bounds are surprisingly close (they are equal in order if d is considered a constant factor).

Claim 6.1. *There exists an explicitly defined network $\mathcal{N}$ with usual parameters (d, n, L) which can achieve a number of activation regions which is $\Theta((n/d)^{dL})$.*

To prove this overarching claim it will be useful to first consider networks in one dimension. We will then be able to utilize this 1-d construction for a construction in arbitrary dimension. So first we prove the lemma:

Lemma 6.1. *There exists an explicitly defined network $\mathcal{N}(x)$ with $x \in \mathbb{R}$ which can achieve a number of activation regions which is $\Theta(n^L)$.*

Proof of Lemma 6.1.1. *We choose the first layer weights such that each neuron* $n[1,j] = \sigma(x - j')\ \forall\, j$ *with* $j' = j - 1$*. This forms a series of* n *ReLu functions of slope one which turn on at* $x = 0, x = 1, ...$ *and so on. This forms* $\Theta(n)$ *regions.*

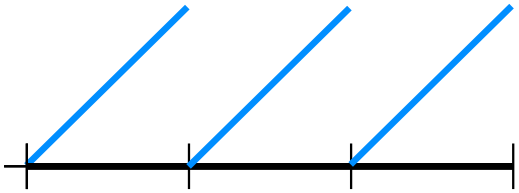

Figure 2: First layer neurons from $\mathbb{R}^1$ construction ($n = 3$)

So far, the choices of weights have been made roughly w.l.o.g, but in the second layer we choose a special structure of alternating positive and negative weights for each neuron s.t:

$$n[2,j] = \sigma(\,\langle\,[1,-2,+2,-2,...], \mathcal{N}_1(x)\rangle + b_j) \qquad \forall\, j$$
$$b_j = -\frac{j'}{n}$$

This forms a set of n *nested saw-tooth functions, one for each second layer neuron. So now, within almost every activation region of* $\mathcal{N}_1(x)$ *we have formed* n *unique states of the second layer neurons. Thus, after two layers we have* $\Theta(n^2)$ *regions.*

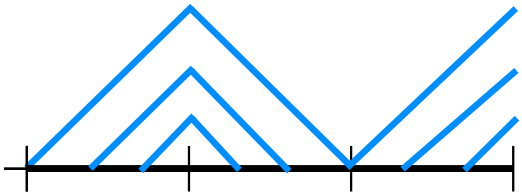

Figure 3: Second layer neurons from $\mathbb{R}^1$ construction ($n = 3$)

For every subsequent layer, we continue this same strategy of alternating weights and adjusted biases to form neuron functions with ever finer teeth. Specifically we choose the network parameters s.t:

$$n[i,j] = \sigma(\,\langle\,[1,-2,+2,-2,...], \mathcal{N}_{i-1}(x)\rangle + b_{i,j}) \qquad \forall\, i,j$$
$$b_{i,j} = -\frac{j'}{n^{i'}}$$

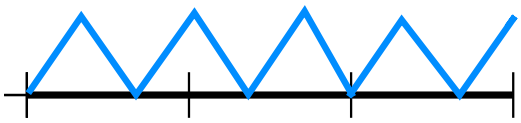

Figure 4: A single third layer neuron from $\mathbb{R}^1$ construction ($n = 3$)

For every layer we multiplicatively increase the number of regions by n, thus an L layer network can achieve $\Theta(n^L)$ activation regions.

Proof of Claim 6.1.1. *With Lemma 6.1 proven it is quite straightforward to form the construction in $\mathbb{R}^d$. The $\mathbb{R}^d$ construction follows by replicating the $\mathbb{R}^1$ construction in each dimension. Thus, for each layer we will divide the n neurons of each layer into d groups to form d copies of the $\mathbb{R}^1$ construction. To clarify, every row of W_1 will be a 1-sparse vector and every row of W_j will be a $(\frac{n}{d})$-sparse vector $\forall\, j > 1$. To count the number of activation regions in our general construction, we count the number of regions in a d-dimensional regular grid with $\Theta((n/d)^L)$ divisions per dimension. This gives us our desired claim of $\Theta((n/d)^{dL})$.*

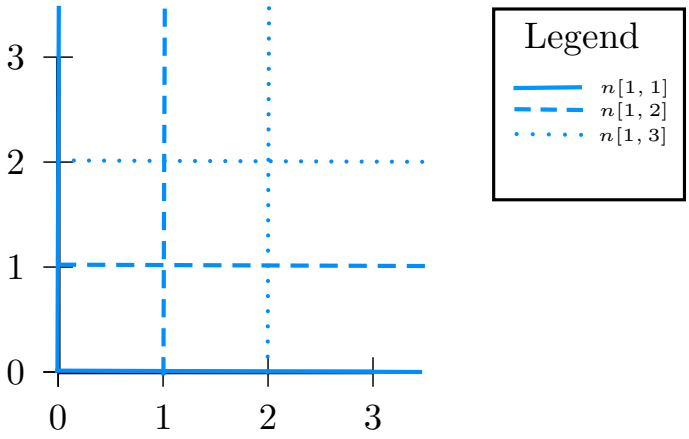

Figure 5: First layer neuron boundaries from $\mathbb{R}^2$ construction ($n = 6$). I.e. where each nueron is exactly zero.

Corollary 6.0.1. *From Claim 6.1 we can then say that $N_{max}(d, n, L) = \Omega((n/d)^{dL}$*

Remark 6.1. *Notice that our lower bound of $\Omega((\frac{n}{d})^{dL})$ is quite close to the naive upper bound of $O((\frac{ne}{d})^{dL})$. Ultimately, there is a gap in the bounds as a multiplicative factor of e^{dL}. It would be interesting to prove whether or not this gap can be improved, although it is a minor theoretical note.*

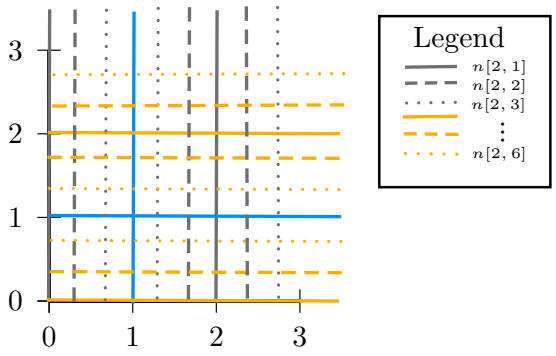

Figure 6: Second layer neuron boundaries from $\mathbb{R}^2$ construction ($n = 6$). I.e. where each nueron is exactly zero.

Remark 6.2. *In an attempt to close the gap described above, I analyzed a construction which does not proceed by replicating in each dimension. Rather, it applied the concept of alternating sign of weights in multiple dimensions; however, it was not clear whether or not such a construction is better. To clarify, rather than using only $\frac{n}{d}$-sparse row vectors I hoped to leverage the full complexity of the network by considering 0-sparse vectors. In this way, layers beyond layer 1 would progress by forming a linear combination of all previous neurons rather than just $\frac{n}{d}$ of them.*

Remark 6.3. *These sawtooth functions are a natural candidate for a maximal construction as they are self-replicating and periodic. As it turns out there are slightly more clever ways to compose functions to achieve a marginally better lower bound to give $N_{max}(d, n, L) = \Omega((\frac{n+1}{d})^{dL})$ [see paper]*

7 Activation Regions form a Polyhedral Complex

The way in which the neurons of a deep ReLu network divide the input space $\mathbb{R}^d$ has interesting hierarchical properties and structure. Furthermore, the divisions form a polyhedral complex [see our paper]

8 Activation Regions Under Assumption of Randomness

There are two notable papers which prove that the expected number of activation regions of a deep ReLu network with random weights can be much lower

than the worst case bounds investigated here in these notes. Here are some links:

1. https://arxiv.org/abs/1901.09021
2. https://arxiv.org/abs/1906.00904

9 Open Problems

1. Can the e^{dL} multiplicative factor be closed in the gap described in Remark 6.1?
2. Can some test be devised to bound the number of regions of a trained network? This is interesting as we cannot guarantee randomness after a network is trained (although the papers in section 8 show the number of regions empirically does not deviate much after training for low dimensions).
3. Can some restriction on the ℓ_1 or ℓ_0 norms of the matrices of the network $\mathcal{N}$ be leveraged to bound the number of activation regions? This could be interesting in both the generalized or random weights cases.
4. Resolving the questions in Remark 6.2 by implementing the construction and counting the regions.