Statistical Learning Theory
Final Project

Noah Amsel

May 4, 2019

This report is about a recent paper by Neyshabur et al. [2] on the role of overparametrization in neural networks' ability to generalize from training data to testing data. Traditionally in statistics overparameterization is considered a flaw, as models with too many parameters are likely to overfit the training data and thus generalize poorly. Surprisingly, neural network models often seem to buck this trend, even without help from regularization or training tricks like dropout. Even when a neural network can fit the training data perfectly, increasing the complexity of the model may actually help generalization.

Previous work has attempted to account for this phenomenon using the tools of Statistical Learning Theory. Classical complexity bounds increase with the number of parameters. For example, Zhang et al. show that neural nets can learn to perfectly classify random images with random labels [3]. (They provide both empirical demonstration and a formal proof.) This would seem to doom the usefulness of the VC dimension for bounding the complexity of neural nets. Thus, the goal is to find a new bound on the complexity of these networks that decreases with the number of parameters. Other authors have proved several bounds based on norms of the weight matrices of the network that they hoped would better explain the generalization behavior. In the current paper, the authors show emprically that those bounds still increase with the number of parameters. The key insight of this paper is, as the authors write, "to characterize complexity at a unit level." This means they consider the individual contribution of each hidden unit in the hidden layer and construct a bound based on that. This bound actually does decrease with the size of the model, and thus shows promise in explaining the generalization properties they observed.

This report is organized as follows. First we discuss the authors experiments and the motivation behind the paper. Then, we prove the key result of the paper—a bound on the Rademacher complexity of a class of neural networks with given bounds on the weights. Next we use covering arguments to prove a bound on the generalization error that holds for all networks in this class with high probability, without bounds on the weights, and analyze it empirically. Lastly, we explore some further experimental directions. I've tried to highlight the key steps and give intuitions wherever possible while still presenting the selected proofs in full.

Contents

1 Experiments and Motivation

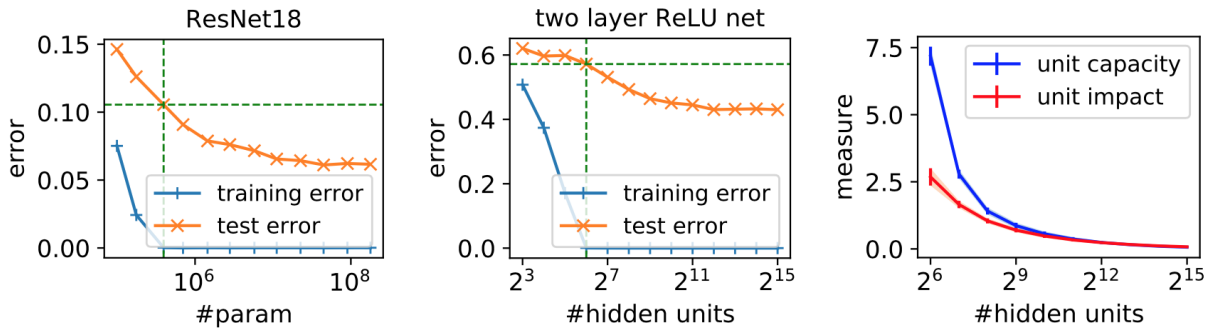

Figure 1: The authors' experiments with generalization. Left panel shows training and testing performance of the sophisticated ResNet18 architecture on the CIFAR-10 image classification dataset. Middle panel shows the performance of a two layer ReLU network with varying numbers of hidden units on CIFAR-10. In both experiments, test accuracy improves long after the model fits the training error perfectly. Right panel: the unit capacties and unit impacts (described below) decay rapidly as the number of hidden units grows.

The authors conduct experiments to explore the unintuitive generalization properties of neural networks. To distill the issue and make it tractable for mathematical analysis, they study a simple architecture—2-layer fully connected ReLU networks. The results for CIFAR-10 are shown in Figure 1. The left and center panels show that the simple feed-forward architecture exhibits the same generalization behavior as the state-of-the-art image recognition network. In both cases, training error goes to zero when the models reach a certain size, but the test error continues to improve even though the models seem to be overparameterized.

The third panel contains the key insight of the paper. Consider a single unit in the hidden layer. Each entry of the input contributes to its value according to a weight vector u. Likewise, it contributes to each dimension of the output according to a vector v. If we imagine that each hidden unit encodes some feature of the input, we can think of the second layer as a simple logistic classifier. Adding more hidden units to the model increases the number of features available to this classifier. If we add enough hidden units, at least some of them will be randomly initialized to capture meaningful features. This means that with many hidden units, we shouldn't need to train the first layer very much to get good results. Likewise, if we increase the number of hidden units, the importance of each individual unit should decrease.

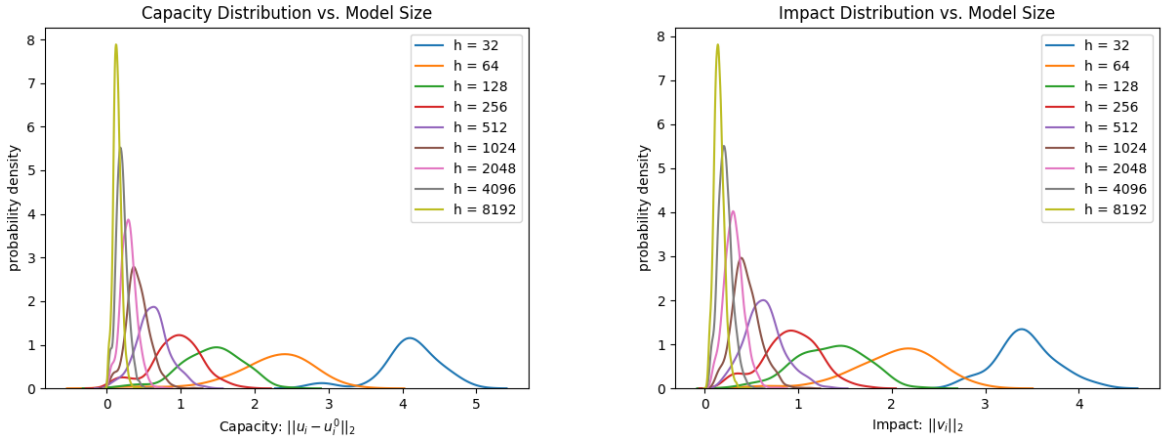

Figure 2: Distribution of per-unit complexity measures for different sized models. Each sample in a given histogram corresponds to a different hidden unit. The left panel shows the how much the incoming weights to each hidden unit have changed during training, and the right panel shows the norm of out the outgoing weights from each hidden unit (see text). h is the number of hidden units. As h increases, the complexity measures decrease rapidly. Results are from my experiments with MNIST.

We can run some experiments to see how these per-unit weights are affected by over-parameterization. (The following figures come from my own replication results based on MNIST.) Figure 2 shows the distribution of the unit complexities for different sized models. For a given hidden unit i, let u_i be its associated incoming weights in the first layer, and let v_i be its outgoing weights in the second layer. Let u_i^0 be the initial value of its incoming weights before training takes place. The histograms in the left panel show the distribution of $\|u_i - u_i^0\|$ over all the hidden units i. Those in the right panel show the distribution of $\|v_i\|$. As you can see, these per-unit complexity measures decrease rapidly as the size of the model grows (h ranges from 32 to 8192). It appears that, as posited, the model doesn't need to train the first layer as much when h is very large. Likewise, the impact of any one unit's activation on the final output (measured by $\|v_i\|$) decreases too.

Now return to the third panel of Figure 1. It shows the average value of $\|u - u^0\|$ and $\|v\|$ plotted against the number of hidden units h. As the authors note, "these unit level

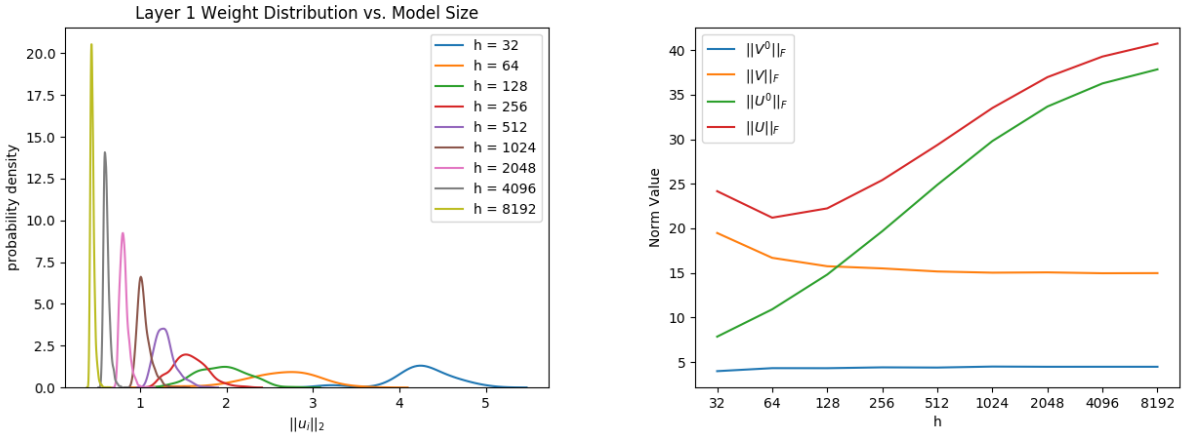

Figure 3: Further visualizations of per-unit and matrix level measures. The left panel shows a similar plot to those in Figure 2, but for the measure $\|u_i\|$. The figure looks promising, as this per-unit measure decays as h grows. But unlike $\|v_i\|$, it doesn't decay quickly enough to keep up when we aggregate over the whole matrix, as shown in the right panel (red line). The right panel shows that, unlike V, the norm of U is largely driven by the large norm of its initialization U^0 (green line), not by training. This observation suggests why taking the difference $(U - U^0)$ is needed for this layer. Results are from my experiments on MNIST.

measures shrink at a rate faster than $1/\sqrt{h}$ for each hidden unit." This is important because when we aggregate the unit-level measures together, we're going to be squaring and adding them. If the measures decay more slowly than $1/\sqrt{h}$, their increasing number will dominate the per-unit decay, and the aggregate measure will grow. (Figure 3 demonstrates how this happens for $\|u\|_2$.) Choosing the right matrix and the right norm is trickier than it may seem, as Figure 6 will demonstrate. The right panel of Figure 4 plots the matrix level complexity measures that we will focus on: $\|V\|_F$ and $\|U - U^0\|_F$, where U and V are the full weight matrices of the first and second layer respectively. As you can see, these measures decay as model size grows even though they aggregate over all hidden units. This result suggests that these simple measures can help us understand over-parameterization in neural networks. The bulk of this paper will be showing how to use them to rigorously bound the model's generalization error.

2 Mathematical Formulaion

We first define our terms for easy reference.

- d is the input dimension.
- h is the number of hidden units (the size of the hidden layer).
- c is the output dimension (the number of classes).

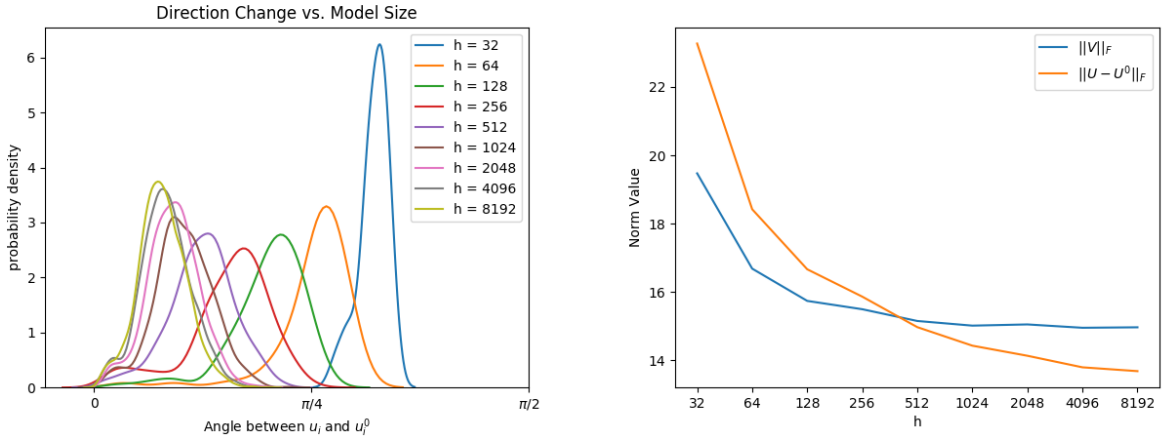

Figure 4: The right panel shows distributions of angles between u_i and u for various model sizes. For small models, training changes the directions of these weights by more than 90°, but for large models their directions barely change at all. The right panel plots the main matrix level complexity measures we will work with. Note that they display the same rapid decay seen in per-unit measures (see Figure 1, right panel).

- $U \in \mathbb{R}^{h \times d}$ is the weight matrix of the first layer. The ith row of U, u_i, is the vector of incoming weights to the ith hidden unit. The initial setting of this matrix before training takes place is called U^0, and its rows are u_i^0.
- We define $\beta_i = \|u_i - u_i^0\|_2$ to be the **unit capacity** of the ith hidden unit, and $\beta = (\beta_1, \ldots \beta_h)$.
- $V \in \mathbb{R}^{c \times h}$ is the weight matrix of the second layer. The ith *column* of V, v_i, is the vector of outgoing weights from the ith hidden unit. The initial setting of this matrix before training takes place is called V^0, and its columns are v_i^0.
- We define $\alpha_i = \|v_i\|_2$ to be the **unit impact** of the ith hidden unit, and $\alpha = (\alpha_1, \ldots \alpha_h)$.
- $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^m$ is the dataset and $X = \begin{bmatrix} x_1 \cdots x_m \end{bmatrix}$ is the input matrix
- The **2-layer ReLU network** is defined as follows:

$$f_{V,U}(x) = V[Ux]_+$$

  where $[\cdot]_+$ denotes the element-wise ReLU activation. Note that for simplicity, the network has no bias terms. To perform classification, the class with the largest output score is chosen.

- Given an input $x \in R^d$ and a label $y \in [c]$, the **margin** μ is the difference between the score of the true class and the largest score besides that:

$$\mu(f(x), y) = f(x)[y] - \max_{i \neq y} f(x)[i]$$

- We use the **ramp loss function**, defined as follows:

$$l_\gamma(f(x), y) = \begin{cases} 0 & \mu(f(x), y) > \gamma \\ \mu(f(x), y)/\lambda & \mu(f(x), y) \in [0, \gamma] \\ 1 & \mu(f(x), y) < 0 \end{cases}$$

  That is, the loss is 0 if the margin is at least γ, it's 1 if the output is incorrect (that is, $\max_{i \neq y} f(x)[i] > f(x)[y]$) and it increases linearly between these. Notice that setting $\gamma = 0$ yields the classification loss. Also note that the ramp loss is $\frac{\sqrt{2}}{\gamma}$-Lipschitz. Imagine that for a given vector of input scores $f(x)$, the margin is exactly γ, so $l_\gamma(f(x), y) = 0$. The fastest way to change the loss is to decrease the score of the correct class while simultaneously increasing the score of the highest incorrect class. If we increase/decrease each by $\frac{\gamma}{2}$, the margin will become 0 and $l_\gamma(f'(x), y)$ will be 1 (where $f'(x)$ is the vector of altered scores). Then:

$$\frac{|l_\gamma(f'(x)) - l_\gamma(f(x))|}{\|f'(x) - f(x)\|} = \frac{1}{\sqrt{2}\frac{\gamma}{2}} = \frac{\sqrt{2}}{\gamma}$$

- $L_\gamma(f)$ is the risk of f and $\widehat{L}_\gamma(f)$ is the empirical risk.
- For reference, the Rademacher complexity of a class of scalar-valued functions $\mathcal{F}$ with respect to a dataset $\mathcal{D}$ is

$$\mathcal{R}_\mathcal{D}(\mathcal{F}) = \mathbb{E}_{\xi \in \{\pm 1\}^m}\Big[\sup_{f \in \mathcal{F}} \frac{1}{m} \sum_{i=1}^{m} \xi_i f(x_i)\Big]$$

We want to bound the risk with respect to the classification loss. To do this we can use a standard generalization bound in terms of the Rademacher complexity. If $\mathcal{D}$ is a dataset of size m, then

$$L_0(f) \leq \widehat{L}_\gamma(f) + 2\mathcal{R}_\mathcal{D}(l_\gamma \circ \mathcal{F}) + 3\sqrt{\frac{\ln(2/\delta)}{2m}} \tag{1}$$

with probability $1 - \delta$. So our goal is now to bound $\mathcal{R}_\mathcal{D}(l_\gamma \circ \mathcal{F})$.

3 Bounding Rademacher Complexity

The main result of the paper is a bound on Rademacher complexity in terms of the unit capacities α_j and the unit impacts β_j. To make the bound slightly more general, we will actually let α_j and β_j be the upper bounds on the capacity and impact of the jth unit. To make that formal, we can define a restriction of our function class

$$W = \{(V, U) : \|v_i\|_2 \leq \alpha_i, \left\|u_i - u_i^0\right\|_2 \leq \beta_i \quad \forall i\}$$
$$\mathcal{F}_W = \{f_{V,U}(x) : (V, U) \in W\}$$

and take the Rademacher complexity over it. (Later in the paper we'll see how to remove these restrictions.)

Theorem 1. Given training set $\mathcal{D} = \{x_i\}_{i=1}^m$ and $\gamma > 0$,

$$\mathcal{R}_{\mathcal{D}}(l_\gamma \circ \mathcal{F}_W) \leq \frac{2\sqrt{2c}+2}{\gamma m} \sum_{j=1}^{h} \alpha_j (\beta_j \|X\|_F + \left\|u_j^0 X\right\|_2) \tag{2}$$

$$\leq \frac{2\sqrt{2c}+2}{\gamma m} \|\alpha\|_2 \left(\|\beta\|_2 \|X\|_F + \left\|U^0 X\right\|_F \right) \tag{3}$$

Here the second line immediately follows from the first by writting the summation as an inner product $\langle \alpha, \beta \|X\|_F + \text{ vector derived from } U^0 X \rangle$ and applying Cauchy-Schwartz. Based on our heuristic analysis above, we should expect this bound to decrease as h increases. The bound is expressed as the sum across the hidden units of a sort of per-unit complexity score. Unlike previous approaches, which break up complexity by layer, this bound cuts the other way around: it cuts by unit and combines information from different layers. As we saw above, α_j and β_j both decrease a bit faster than $O(1/\sqrt{h})$. If we simply sum up α_j across all h units, we'll get a bound which would increase with h. By multiplying them however, we are able to get terms which decrease like $O(1/h)$, and thus stay ahead of the growing number of terms.
The proof proceeds in three steps:

Lemma 2. For $c \in \mathbb{Z}_{>0}$, $r > 0$, reference matrix $V^0 \in \mathbb{R}^{c \times d}$ and dataset $x_{i\,i=1}^m$,

$$\mathop{\mathbb{E}}_{\substack{\xi_i \in \{\pm 1\}^c \\ i \in [m]}} \left[\sup_{\|V - V^0\|_F \leq r} \sum_{i=1}^{m} \langle \xi_i, V x_i \rangle \right] \leq r\sqrt{c} \|X\|_F$$

The left hand side of this inequality is "Rademacher-like", where the function in question is the linear operator V. Unlike the regular Rademacher average, this function is not scalar-valued, so we must use a whole matrix of Rademacher random variables ($\xi_1, \ldots \xi_c$ are each vectors) for the expression to work. The key to the lemma is that this flexibility of this operator is hobbled by the requirement that it be close to V^0. It doesn't matter how big or small V is, just that it falls within a narrow "range". It also makes sense that the size of the input matrix $\|X\|_F$ should be part of the bound. Imagine doubling X (thus doubling its Frobenius norm as well). To restore the terms on the left hand size to what they were, the supremum would need to pick $V/2$ instead of V. But because of the restriction on V, this might not be possible. Thus, scaling the matrix X could drastically change the "Rademacher-like" flexibility of the function class V. The proof is not very deep – it just involves breaking up the terms and messing around with norms.

Proof. We can rewrite $\langle \xi_i, V x_i \rangle = \langle V, \xi_i x_i^\top \rangle$ and then use linearity to transform the sum

$$\sum \langle V, \xi_i x_i^\top \rangle = \langle V - V_0, \sum \xi_i x_i^\top \rangle + \langle V_0, \sum \xi_i x_i^\top \rangle$$

Since the sup of a sum is less than the sum of the sup, we can bound the left-hand side of

the lemma by

$$\mathop{\mathbb{E}}_{\substack{\xi_i\in\{\pm1\}^c\\ i\in[m]}}\left[\sup_{\|V-V^0\|_F\le r}\sum_{i=1}^m\langle\xi_i,Vx_i\rangle\right]\le\mathop{\mathbb{E}}_{\substack{\xi_i\in\{\pm1\}^c\\ i\in[m]}}\left[\sup_{\|V-V^0\|_F\le r}\langle V-V_0,\sum_{i=1}^m\xi_i x_i^\top\rangle\right]$$
$$+\mathop{\mathbb{E}}_{\substack{\xi_i\in\{\pm1\}^c\\ i\in[m]}}\left[\sup_{\|V-V^0\|_F\le r}\langle V_0,\sum_{i=1}^m\xi_i x_i^\top\rangle\right]$$

But now, notice that the second term disappears. Since it does not depend on V, we can cross out the sup. Then, since xi_i's are symmetrically distributed, the expectation is 0. This is the crucial step, as it shows that this "Rademacher-like" complexity term does not depend on the scale of V_0 at all. From here, note that by using Cauchy-Schwartz we can eliminate the sup in the first term too, since

$$\langle V-V_0,\sum_{i=1}^m\xi_i x_i^\top\rangle\le\|V-V_0\|_F\left\|\sum_{i=1}^m\xi_i x_i^\top\right\|_F\le r\sqrt{\left\|\sum_{i=1}^m\xi_i x_i^\top\right\|_F^2}$$

Plugging this back into the expectation and using Jensen's Inequality yields

$$\mathop{\mathbb{E}}_{\substack{\xi_i\in\{\pm1\}^c\\ i\in[m]}}\left[\sup_{\|V-V^0\|_F\le r}\langle V-V_0,\sum_{i=1}^m\xi_i x_i^\top\rangle\right]\le r\sqrt{\mathop{\mathbb{E}}_{\substack{\xi_i\in\{\pm1\}^c\\ i\in[m]}}\left\|\sum_{i=1}^m\xi_i x_i^\top\right\|_F^2}$$

Now, each term in each entry in the matrix $\left\|\sum_{i=1}^m\xi_i x_i^\top\right\|_F^2$ is an entry of X multiplied by a Rademacher random variable. The expectation of the square of one of these elements is simply the corresponding entry of X. Each entry of X appears c times, so we're really just taking the sqare root of $c\|X\|_F^2$. Thus

$$=r\sqrt{c}\|X\|_F$$

□

We now come to the heart of the proof. This is the step where we break the Rademacher complexity up as a sum over the hidden units.

Lemma 3. Given training set $\mathcal{D}=\{x_i\}_{i=1}^m$ and $\gamma>0$, let the Rademacher complexity of the restricted function class $\mathcal{F}_W$, composed with the ramp loss, is bounded as follows:

$$\mathcal{R}_{\mathcal{D}}(l_\gamma\circ\mathcal{F}_W)\le\frac{2}{\gamma m}\sum_{j=1}^h\mathop{\mathbb{E}}_{\substack{\xi_i\in\{\pm1\}^c\\ i\in[m]}}\left[\sup_{\|v_j\|\le a_j}\sum_{i=1}^m(\left|\langle u_j^0,x_i\rangle\right|+\beta_j\|x_i\|_2)\langle\xi_i,v_j\rangle\right]$$
$$+\frac{2}{\gamma m}\sum_{j=1}^h\mathop{\mathbb{E}}_{\xi\in\{\pm1\}^m}\left[\sup_{\left\|u_j-u_j^0\right\|_2\le\beta_j}\sum_{i=1}^m\xi_i\alpha_j\langle u_j,x_i\rangle\right]$$

As you can see, the bound is a sum where each term depends on only the weights associated with hidden unit j. The terms in each sum resemble Rademacher complexities, but as above, they measure the complexity of a set of vectors rather than of scalars. The first summation measures the complexity of the second layer ($\{v_j : \|v_j\|_2 \leq a_j\}$, from the definition of $\mathcal{F}_W$). The expression $\langle u_j^0, x_i \rangle$ is the activation of the hidden unit before training, and $\beta_j \|x_i\|_2$ measures how much this activation can change during training (recall $\beta_j = \|u_j - u_j^0\|_2$). Together, they bound the magnitude of the activation, which is needed to scale the complexity score. The second summation measures the complexity of the second layer ($\{u_j : \|u_j - u_j^0\|_2 \leq \beta_j\}$), and the scalar α_j measures how much the activations calculated in the first layer (that is, $\langle u_j, x_i \rangle$) are capable of influencing the output of the network (this is why the authors refer to $\alpha_j = \|v_j\|$ as the unit's 'impact').

Proof. For simplicity let $\rho = \left|\langle u_j^0, x_i \rangle\right|$. The definition of the Rademacher complexity is given in Equation 4. (Note that in the first expression, the expectation is taken over m vectors ξ_i, but only the first coordinate of each is used, so it accords with the definition of Rademacher complexity.) Our immediate goal is to bound it by Equation 5.

$$m\mathcal{R}_{\mathcal{D}}(l_\gamma \circ \mathcal{F}_W) = \mathop{\mathbb{E}}_{\substack{\xi_i \in \{\pm 1\}^c \\ i \in [m]}} \left[\sup_{(V,U) \in W} \sum_{i=1}^{m} \xi_{i1} l_\gamma(V[Ux_i]_+, y_i) \right] \tag{4}$$

$$\leq \mathop{\mathbb{E}}_{\substack{\xi_i \in \{\pm 1\}^c \\ i \in [m]}} \left[\sup_{(V,U) \in W} \frac{2}{\gamma} \sum_{i=1}^{m} \sum_{j=1}^{h} \left((\rho_{ij} + \beta_j \|x_i\|_2) \langle \xi_i, v_j \rangle + \xi_{i1} \alpha_j \langle u_j, x_i \rangle \right) \right] \tag{5}$$

At first this seems a simple matter of bounding each of the summands in Equation 4 by the one in Equation 5, making the desired transformation in one fell swoop. But because of the supremum over V and U, which affects all terms, we will have to transform the sum one term (one datapoint) at a time using induction. Thus, using Equation 4 as a base case, we wish to show

$$m\mathcal{R}_{\mathcal{D}}(l_\gamma \circ \mathcal{F}_W) \leq \mathop{\mathbb{E}}_{\substack{\xi_i \in \{\pm 1\}^c \\ i \in [m]}} \left[\sup_{(V,U) \in W} \sum_{i=t}^{m} \xi_{i1} l_\gamma(V[Ux_i]_+, y_i) + \frac{2}{\gamma} \sum_{i=1}^{t-1} \sum_{j=1}^{h} \left((\rho_{ij} + \beta_j \|x_i\|_2) \langle \xi_i, v_j \rangle + \xi_{i1} \alpha_j \langle u_j, x_i \rangle \right) \right]$$

by induction on t. At every t, we transfer another term from the first sum to the second. We'll define some notation to declutter the expression and highlight the term we are transfering. Let

$$\phi_t(V,U) = \sum_{i=t+1}^{m} \xi_{i1} l_\gamma(V[Ux_i]_+, y_i) + \frac{2}{\gamma} \sum_{i=1}^{t-1} \sum_{j=1}^{h} \left((\rho_{ij} + \beta_j \|x_i\|_2) \langle \xi_i, v_j \rangle + \xi_{i1} \alpha_j \langle u_j, x_i \rangle \right)$$

which is exactly the expression in the sup, except we're starting i at $t+1$ instead of t. Thus we wish to show

$$m\mathcal{R}_{\mathcal{D}}(l_\gamma \circ \mathcal{F}_W) \leq \mathop{\mathbb{E}}_{\substack{\xi_i \in \{\pm 1\}^c \\ i \in [m]}} \left[\sup_{(V,U) \in W} \xi_{t1} l_\gamma(V[Ux_t]_+, y_t) + \phi_t(V,U) \right]$$

Assume this holds for $t' \le t$, and let's prove it for $t' = t+1$. The first step resembles the symmetrization technique we've used in class. To get rid of the pesky ξ_{t1}, we will double up each term:

$$\mathop{\mathbb{E}}_{\substack{\xi_i \in \{\pm 1\}^c \\ i \in [m]}} \left[\sup_{(V,U) \in W} \xi_{t1} l_\gamma(V[Ux_t]_+, y_t) + \phi_t(V,U) \right] \tag{6}$$

$$= \frac{1}{2} \mathop{\mathbb{E}}_{\substack{\xi_i \in \{\pm 1\}^c \\ i \in [m]}} \left[\sup_{(V,U),(V',U') \in W} l_\gamma(V[Ux_t]_+, y_t) - l_\gamma(V'[U'x_t]_+, y_t) + \phi_t(V,U) + \phi_t(V',U') \right] \tag{7}$$

$$\le \frac{1}{2} \mathop{\mathbb{E}}_{\substack{\xi_i \in \{\pm 1\}^c \\ i \in [m]}} \left[\sup_{(V,U),(V',U') \in W} \frac{\sqrt{2}}{\gamma} \left\| V[Ux_t]_+ - V'[U'x_t]_+ \right\|_2 + \phi_t(V,U) + \phi_t(V',U') \right] \tag{8}$$

The intuition here is as follows (ignore ϕ_t for simplicity). If $\xi_{t1} = 1$, then the supremum will pick V, U to maximize the $l_\gamma(V[Ux_t]_+, y_t)$, and if $\xi_{t1} = -1$ if will seek to minimize it—that is, to set it to 0 (since l_γ is non-negative). Thus, the expectation over ξ_i is for the supremum to be half of the max value of $l_\gamma(V[Ux_t]_+, y_t)$. In the second expression, the first two terms are deterministic, and the supremum will try to maximize $l_\gamma(V[Ux_t]_+, y_t)$ while setting $l_\gamma(V'[U'x_t]_+, y_t)$ to 0. Dividing by $\frac{1}{2}$ once again yields half the max value. Finally, the last line follows because the ramp loss is $\frac{\sqrt{2}}{\gamma}$-Lipschitz.
Next we break up the norm using the triangle inequality twice:

$$\begin{aligned}
\left\| V[Ux_t]_+ - V'[U'x_t]_+ \right\|_2 &\le \left\| V[Ux_t]_+ - V'[Ux_t]_+ + V'[Ux_t]_+ - V'[U'x_t]_+ \right\|_2 \\
&\le \left\| V[Ux_t]_+ - V'[Ux_t]_+ \right\|_2 + \left\| V'[Ux_t]_+ - V'[U'x_t]_+ \right\|_2 \\
&\le \sum_{j=1}^{h} \left\| [\langle u_j, x_t \rangle]_+ v_j - [\langle u_j, x_t \rangle]_+ v'_j \right\|_2 + \left\| [\langle u_j, x_t \rangle]_+ v'_j - [\langle u'_j, x_t \rangle]_+ v'_j \right\|_2 \\
&\le \sum_{j=1}^{h} [\langle u_j, x_t \rangle]_+ \left\| v_j - v'_j \right\|_2 + \left| [\langle u_j, x_t \rangle]_+ - \langle u'_j, x_t \rangle]_+ \right| \left\| v'_j \right\|_2 \\
&\le \sum_{j=1}^{h} \left| \langle u_j, x_t \rangle \right| \left\| v_j - v'_j \right\|_2 + \left| \langle u_j, x_t \rangle - \langle u'_j, x_t \rangle \right| \alpha_j
\end{aligned}$$

In this last line, we've used the non-negativity and 1-Lipschitzness of ReLU to replace $[\langle u_j, x_t \rangle]_+$ with $|\langle u_j, x_t \rangle|$ and $\left| [\langle u_j, x_t \rangle]_+ - \langle u'_j, x_t \rangle]_+ \right|$ with $\left| \langle u_j, x_t \rangle - \langle u'_j, x_t \rangle \right|$. This seems to be the only place where the activation function comes into the proof, suggesting that any activation function with these properties would work. We have also replaced $\left\| v'_j \right\|$ with its bound, α_j. To introduce β_j, we will split up $\langle u_j, x_t \rangle$ using triangle inequality:

$$\left| \langle u_j, x_t \rangle \right| = \left| \langle u_j - u_j^0, x_t \rangle + \langle u_j^0, x_t \rangle \right| = \left\| u_j - u_j^0 \right\| \left\| x_t \right\|_2 + \left\| u_j^0, x_t \right\|_2 \le \beta \left\| x_t \right\|_2 + \rho_{tj}$$

Substituting back into Equation 8 above,

$$m\mathcal{R}_{\mathcal{D}}(l_\gamma \circ \mathcal{F}_W) \leq \frac{1}{2} \mathop{\mathbb{E}}_{\substack{\xi_i \in \{\pm 1\}^c \\ i \in [m]}} \Bigg[\sup_{(V,U),(V',U') \in W} \frac{\sqrt{2}}{\gamma} \sum_{j=1}^{h} (\beta \left\|x_t\right\|_2 + \rho_{tj}) \left\|v_j - v'_j\right\|_2 + \left|\langle u_j, x_t\rangle - \langle u'_j, x_t\rangle\right| \alpha_j + \phi_t(V,U) + \phi_t(V',U') \Bigg]$$

Now we can "undo" the symmetrization and return to just one copy of U and V. To deal with $\left\|v_j - v'_j\right\|_2$, we'll need a result of Maurer [1]. If the entries of ξ are Rademacher distributed random variables, then for any vector $z \in \mathbb{R}^h$,

$$\left\|z\right\|_2 \leq \sqrt{2} \mathop{\mathbb{E}}_{\xi \in \{\pm 1\}^d} \left|\langle \xi, z\rangle\right|$$

That is, applying the triangle inequality

$$\left\|v_j - v'_j\right\|_2 \leq \sqrt{2} \mathop{\mathbb{E}}_{\xi \in \{\pm 1\}^d} \left|\langle \xi, v_j\rangle\right| + \sqrt{2} \mathop{\mathbb{E}}_{\xi \in \{\pm 1\}^d} \left|\langle \xi, v'_j\rangle\right|$$

We now have two identical and independent copies of the same expression, one with u_j, v_j, U, and V, the other with u'_j, v'_j, U', and V' — except for the term $\left|\langle u_j, x_t\rangle - \langle u'_j, x_t\rangle\right|$. Pulling in the factor of $\frac{1}{2}$, we can treat $\sup_{(V,U),(V',U') \in W}$ as a $\sup_{(V,U) \in W}$ over just one copy, adding a Rademacher random variable to correct the term noted previously (by the reverse logic we discussed when performing the symmetrization). Hence:

$$m\mathcal{R}_{\mathcal{D}}(l_\gamma \circ \mathcal{F}_W) \leq \mathop{\mathbb{E}}_{\substack{\xi_i \in \{\pm 1\}^c \\ i \in [m]}} \Bigg[\sup_{(V,U) \in W} \frac{2}{\gamma} \sum_{j=1}^{h} (\beta \left\|x_t\right\|_2 + \rho_{tj}) \langle \xi_t, v_j\rangle + \xi_{t1} \alpha_j \langle u_j, x_t\rangle + \phi_t(V,U) \Bigg]$$

We have succeeded in transforming the t-th term from being in the form of the sum in Equation 4, to that of the sum in Equation 5. Our inducation step is finished, and by setting $t = m$, we can conclude:

$$m\mathcal{R}_{\mathcal{D}}(l_\gamma \circ \mathcal{F}_W) \leq \mathop{\mathbb{E}}_{\substack{\xi_i \in \{\pm 1\}^c \\ i \in [m]}} \Bigg[\sup_{(V,U) \in W} \frac{2}{\gamma} \sum_{i=1}^{m} \sum_{j=1}^{h} \Big((\rho_{ij} + \beta_j \left\|x_i\right\|_2) \langle \xi_i, v_j\rangle + \xi_{i1} \alpha_j \langle u_j, x_i\rangle \Big) \Bigg] \quad (9)$$

Now we have a separate term for each datapoint and hidden unit. Our supremum imposes restrictions on U and V, but notice that these restrictions are on the vectors u_j and v_j, the weights for a given unit. That is, the restrictions on hidden unit j are independent of those on hidden unit $j+1$. Here's the payoff. If we switch the order of the sums and expand the restriction $(V,U) \in W$, we get

$$\sup_{\substack{\left\|u_k - u_k^0\right\|_2 \leq \beta_k \\ \left\|v_k\right\|_2 \leq \alpha_k, k \in [h]}} \sum_{j=1}^{h} \sum_{i=1}^{m} \cdots$$

This makes it clear that we can optimize over u_k, for instance, without affecting u_{k+1}, since u_k will only appear in the term where $j = k$. Thus, we can pull the first sum outside the supremum. Then, since each inner term has one piece that depends on u_j and one piece that depends on v_j, we can further split the supremum into two pieces:

$$m\mathcal{R}_{\mathcal{D}}(l_\gamma \circ \mathcal{F}_W) \leq \frac{2}{\gamma} \sum_{j=1}^{h} \mathop{\mathbb{E}}_{\substack{\xi_i \in \{\pm 1\}^c \\ i \in [m]}} \left[\sup_{(V,U) \in W} \sum_{i=1}^{m} (\rho_{ij} + \beta_j \|x_i\|_2) \langle \xi_i, v_j \rangle + \sum_{i=1}^{m} \xi_{i1} \alpha_j \langle u_j, x_i \rangle \right]$$

$$= \frac{2}{\gamma} \sum_{j=1}^{h} \mathop{\mathbb{E}}_{\substack{\xi_i \in \{\pm 1\}^c \\ i \in [m]}} \left[\sup_{\|v_j\|_2 \leq \alpha_j} \sum_{i=1}^{m} (\rho_{ij} + \beta_j \|x_i\|_2) \langle \xi_i, v_j \rangle \right] + \frac{2}{\gamma} \sum_{j=1}^{h} \mathop{\mathbb{E}}_{\substack{\xi_i \in \{\pm 1\}^c \\ i \in [m]}} \left[\sup_{\|u_j - u_j^0\|_2 \leq \beta_j} \sum_{i=1}^{m} \xi_{i1} \alpha_j \langle u_j, x_i \rangle \right]$$

This proves the lemma. □

Finally, we combine the two lemmas to prove Theorem 1.

Proof. (Theorem 1) Our goal is to bound the "Rademacher-like" terms from the last lemma with simpler expressions. In the first term, we can bound

$$\langle \xi_i, v_j \rangle \leq \|\xi_i\| \, \|v_j\| \leq \sqrt{c} \alpha_j$$

Likewise, $\sum_{i=1}^{m} \|x_i\|_2 \leq \|X\|_F$ and $\sum_{i=1}^{m} \rho_{ij} = \left\|u_j^0 X\right\|_2$. In the second term, note that

$$\xi_{i1} \alpha_j \langle u_j, x_i \rangle = \langle \xi_i, (\alpha_j u_j) x_i \rangle$$

This puts the second term in the form necessar to apply Lemma 2, where the "reference matrix" is simply the vector $\alpha_j u_j^0 \in \mathbb{R}^{1 \times d}$. The restriction on u_j can be written

$$\left\|u_j - u_j^0\right\|_2 \leq \beta_j \implies \left\|\alpha_j u_j - \alpha_j u_j^0\right\|_F \leq \alpha_j \beta_j$$

Thus, Lemma 2 yields the bound

$$\mathop{\mathbb{E}}_{\substack{\xi_i \in \{\pm 1\}^c \\ i \in [m]}} \left[\sup_{\|u_j - u_j^0\|_2 \leq \beta_j} \sum_{i=1}^{m} \xi_{i1} \alpha_j \langle u_j, x_i \rangle \right] \leq \alpha_j \beta_j \|X\|_F$$

Combining:

$$\begin{aligned}
\mathcal{R}_{\mathcal{D}}(l_\gamma \circ \mathcal{F}_W) &\leq \frac{1}{m} \frac{2\sqrt{2c}}{\gamma} \sum_{j=1}^{h} \alpha_j \left(\beta_j \|X\|_F + \left\|u_j^0 X\right\|_2 \right) + \frac{2}{\gamma} \sum_{j=1}^{h} \alpha_j \beta_j \|X\|_F \\
&\leq \frac{2\sqrt{2c} + 2}{\gamma m} \sum_{j=1}^{h} \alpha_j \left(\beta_j \|X\|_F + \left\|u_j^0 X\right\|_2 \right) \\
&\leq \frac{2\sqrt{2c} + 2}{\gamma \sqrt{m}} \|\alpha\|_2 \left(\|\beta\|_2 \|X\|_F + \left\|U^0 X\right\|_F \right)
\end{aligned}$$

This completes the proof. □

4 Bounding Generalization Error

We now have a bound on the Rademacher complexity for the class of networks $\mathcal{F}_W$ in terms of the unit-level properties we care about – the maximum unit capacities (β_i) and maximum unit impacts (α_i). This is good news, as our experiments show that these quantities decrease as the size of the model increases. Naively, we can plug our Rademacher bound into Equation 1 to get a bound on the generalization of networks in $\mathcal{F}_W$:

$$L_0(f) \leq \widehat{L}_\gamma(f) + \frac{4\sqrt{2c}+2}{\gamma\sqrt{m}} \|\alpha\|_2 \left(\|\beta\|_2 \|X\|_F + \left\|U^0 X\right\|_F \right) + 3\sqrt{\frac{\ln(2/\delta)}{2m}}$$

holds with probability $1-\delta$ over the choice of training data. But recall that we had to specify α and β ahead of time to define $\mathcal{F}_W$. If we make α and β very large, we will cover more networks, but the bound will become weaker and weaker. To obtain a single bound on all two-layer ReLU networks, we need to cover the entire space $\mathcal{F}$ with a many different choices of α and β. However, the more different settings of α and β we use, the less likely it will be that the bound holds for *all* of them. (Recall that our bound on generalization error is only probably true) Thus, we will prove a lemma to show that we can cover a large number of networks with only a few different choices of α and β. The authors prove the statement for an l_p ball of arbitrary radius, but we will use a unit ball and assume $p = 2$ for simplicity.

Lemma 4. Given a tolerance $\epsilon > 0$ a dimension D, let $S^D = \{x \in \mathbb{R}^D : \|x\|_2 \leq \beta\}$ denote the unit ball. Let

$$K = \lceil \frac{D}{(1+\epsilon)^2 - 1} \rceil$$
$$N = \binom{K+D-1}{D-1}$$

Then there exist N sets $T_1, \ldots T_N$, each of the form $T_i = \{x \in \mathbb{R}^D : |x_j| \leq \alpha_j^i \quad \forall j \in [D]\}$ such that $\|\alpha^i\| \leq (1+\epsilon)$ for all $i \in [N]$ and $S^D \subset \cup_{i=1}^N T_i$.

Consider one of the sets in the cover T_i. Notice that it is entirely determined by the choice of bounding vector, α^i. The set T_i consists of all those vectors which are bounded by α^i *coordinate-wise* (in absolute value). That is, the sets T_i look like rectangles (or hyperrectangles $D > 2$), with one corner at α^i and the opposite corner at $-\alpha^i$. We are interested in this kind of entry-wise covering because that is how we defined our function classes $\mathcal{F}_W$–the set T_i consists of all choices of $\|v_1\|, \ldots \|v_h\|$ which are included in the $\mathcal{F}_W$ corresponding to α^i (of course, we would also need to specify a β^i). The parameter ϵ tells us how frugal we should be in bounding the ball. If we let $1+\epsilon = \sqrt{D}$, we can bound the entire unit ball with a single vector, but as we will see, the bound increases with ϵ. If we force $\epsilon = 0$, we would need an infinite number of vectors to capture every point on the surface of the ball. Figure 5 demonstrates the theorem for $D = 2$.

Proof. The proof idea is simply to discretize the space and send each point on the surface of the ball to the closest point that bounds it in the grid. The spacing of the grid must be

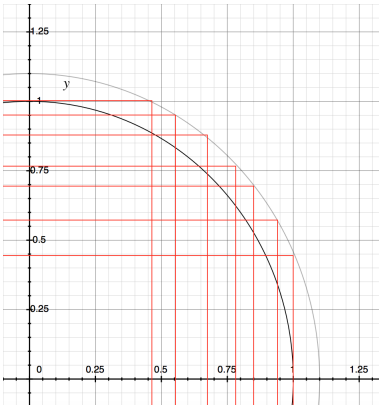

Figure 5: Demonstration of Lemma 4 for the unit ball in $\mathbb{R}^2$. The light circle represents the ball of radius 1.1. Each rectangle is a set T_i. Notice that each T_i lies entirely within the 1.1 radius and that they cover the unit ball completely. Here, $N = 7$.

tight enough that this rounding never sends a point out of the $(1 + e)$ ball. If we then add one set T_i for every point on the grid, we will have covered the ball.
Let $Q = \{a \in \mathbb{R}^D : \alpha_i^2 \in \{1/K, \dots K/K\}, \|\alpha\|_2^2 = (1+D/K)\}$. These are vectors just outside the unit ball whose coordinates take on one of K discrete values. For a given point $x \in S^D$, round its ith coordinate up to $\alpha_i' = \sqrt{\frac{\lceil x_i^2 K \rceil}{K}}$. Clearly $|x_i| \leq \alpha_i'$. Further,

$$\|\alpha'\|_2^2 = \sum_{i=1}^{D} \frac{\lceil x_i^2 K \rceil}{K} \leq \sum_{i=1}^{D} \frac{x_i^2 K + 1}{K} = \|x\|_2 + \frac{D}{K} \leq 1 + \frac{D}{K}$$

To get the norm to be exactly $1 + \frac{D}{K}$, we can pick an arbitrary coordinate α_i' an increase it a few notches. This shows that $\alpha' \in Q$. Finally note that

$$1 + \frac{D}{K} \leq 1 + D\lfloor \frac{D}{(1+\epsilon)^2 - 1} \rfloor \leq (1+\epsilon)^2$$

so α' (and any point in Q) is inside the ball of radius $(1 + \epsilon)$. So the vectors in satisfy our conditions. How many of them are there? That is, how many different ways are there to choose

$$\begin{aligned}
&\alpha_1^2 \in \{1/K, \dots K/K\} \implies K\alpha_1^2 \in [K] \\
&\alpha_2^2 \in \{1/K, \dots K/K\} \implies K\alpha_2^2 \in [K] \\
&\vdots \\
&\alpha_D^2 \in \{1/K, \dots K/K\} \implies K\alpha_D^2 \in [K]
\end{aligned}$$

such that

$$\|\alpha\|_2^2 = (1 + D/K) \implies K\alpha_1^2 + \dots + K\alpha_D^2 = K + D$$

Equivalently, how many ways are there to distribute $K+D$ balls among D urns so that each contains between 1 and K? Since each urn must contain at least 1 ball, let's immediately allocate 1 to each of them. There are now K balls remaining to allocate among the D urns however we like. There are at most $\binom{K+D-1}{D-1} = N$ ways to do so, and the proof is complete. □

Now that we know how to cover balls with entrywise dominance, we can get bounds on more useful function classes than $\mathcal{F}_W$. The bound in Theorem 1 relied on having separate, specified bounds α_i and β_i for each hidden unit. The next lemma will allow us to replace these with just two bounds, one each for the matrix $U - U^0$, and one for V. Again, the authors prove a more general statement with l_p norms, but I state the result for $p = 2$.

Lemma 5. For any $h, d, c, \gamma, \mu > 0$, $\delta \in (0,1)$, $U^0, U \in \mathbb{R}^{h\times d}$, $V \in \mathbb{R}^{c\times h}$, if $\|V\|_F \leq C_1$ and $\|U - U^0\|_F \leq C_2$, the generalization error is bounded by

$$L_0(f) \leq \widehat{L}_\gamma(f) + \frac{4\sqrt{2c}+2}{\gamma\sqrt{m}}(\mu+1)C_1\big(C_2\|X\|_F + \|U^0X\|_F\big) + 3\sqrt{\frac{2\ln N_h + \ln(2/\delta)}{2m}}$$

with probability $1-\delta$ over choice of the training dataset $\mathcal{D} = x_{i=1}^m$. Here $N = \binom{\lceil h/\mu\rceil + h - 1}{h-1}$.

This lemma is the same as the naive bound given in Equation 4 except in two places. First is the additional factor $(\mu+1)$. In this lemma $\mu = (\epsilon+1)^2 - 1$ parameterizes how frugal we are about over-covering. With a high μ, we use fewer vectors (so N_h goes down) but make larger choices of α and β, so we must weaken our bound by penalizing it with a factor $(\mu+1)$. A small value of μ means we can find a covering set with a tighter fit, but we pay for it with a higher N_h. We will select a good μ later. The other difference from the naive bound is the $2\ln N_h$ in the last term. This exists to ensure the probability is still $1-\delta$ after taking the Union Bound. These are small prices to pay for getting to cover a larger function class.

Proof. This one is simple. Since $\alpha = (\|v_1\|_2, \ldots, \|v_h\|_2)$, then we can rewrite $\|V\|_F = \|\alpha\|_2$. Thus, the requirement $\|V\|_F \leq C_1$ is the same as requiring that α lie within a ball of radius C_1. Likewise, $\|U - U^0\| \leq C_2$ means that β lies within a ball of radius C_2. We now apply Lemma 4 to this ball to find coverings $\{\alpha^1, \ldots, \alpha^N\}$ and $\{\beta^1, \ldots \beta^N\}$. If a network satisfies the conditions $\|V\|_F \leq C_1$ and $\|V\|_F \leq C_1$, then by Lemma 4 there must be some i and j such that the impact vector of the network, α, is dominated entrywise by α^i, and the capacity vector β is dominated by β^j. We can then apply the naive bound from Equation 4 for each combination of α^i and β^j to bound the generalization error for all networks that satisfy the conditions. Given a training dataset, what is the probability that the bound actually holds for all networks? Equation 4 was applied N_h^2 times (once for every combination of α^i and β^j). By union bound, the probability that they are all true is at least $1 - N_h^2\delta$.
To normalize this probability, we can reset the δ used in Equation 4 to $\delta' = \frac{\delta}{N_h^2}$. Then, the probability that they all hold is once again $1-\delta$. But we must also change the final term in the bound to match the new δ:

$$3\sqrt{\frac{\ln(2/\delta')}{2m}} = 3\sqrt{\frac{\ln(2N_h^2/\delta)}{2m}} = 3\sqrt{\frac{2\ln N_h + \ln(2/\delta)}{2m}}$$

□

The final step is to remove the restrictions on $\|V\|_F \leq C_1$ and $\|U - U^0\|_F \leq C_2$ altogether. This will yield a high probability bound on the entire function class $\mathcal{F}$. We do so by using another covering argument, this time on C_1 and C_2 themselves. This time, the cost we pay for increasing our coverage is simply a 1 added into some of the terms.

Lemma 6. For any $h, d, c, \gamma, \mu > 0$, $\delta \in (0,1)$, $U^0, U \in \mathbb{R}^{h \times d}$, $V \in \mathbb{R}^{c \times h}$, the generalization error is bounded by

$$L_0(f) \leq \widehat{L}_\gamma(f) + \frac{4\sqrt{2c}+2}{\gamma\sqrt{m}}(\mu+1)(\|V\|_F + 1)\Big(\big\|U - U^0\big\|_F \|X\|_F + \big\|U^0 X\big\|_F + 1\Big) + 3\sqrt{\frac{\ln N_h + \ln(\gamma\sqrt{m}/\delta)}{m}}$$

with probability $1 - \delta$ over choice of the training dataset $\mathcal{D} = x_{i\,i=1}^{m}$.

Proof. We cover the possible values of C_1 and C_2 by applying Lemma 5 for

$$C_1 \in \{1, 2, \ldots \lceil \frac{\gamma\sqrt{m}}{4} \rceil\}$$
$$C_2 \in \{1/\|X\|_F, 2/\|X\|_F, \lceil \frac{\gamma\sqrt{m}}{4} \rceil / \|X\|_F\}$$

For example, if $\|V\|_F = 9.5$, then we can apply Lemma 5 with $C_1 = 10$. We then replace C_1 in that bound with $\|V\|_F + 1 = 10.5 \geq C_1$, which still holds. This will work for any $\|V\|_F \leq \frac{\gamma\sqrt{m}}{4}$. If $\|V\|_F > \frac{\gamma\sqrt{m}}{4}$, then the middle term in the bound we are proving is greater than 1 (because of the scalar $\frac{4\sqrt{2c}+2}{\gamma\sqrt{m}}$). But the risk $L_0(f)$ is surely less than 1, so the bound holds trivially. This argument required applying Lemma 5 $\left(\frac{\gamma\sqrt{m}}{4}\right)^2$ times. Thus we set $\delta' = \delta(\frac{4}{\gamma\sqrt{m}})^2$ and get

$$3\sqrt{\frac{2\ln N_h + \ln(2/\delta')}{2m}} = 3\sqrt{\frac{\ln N_h + \ln(\gamma\sqrt{m}/\delta)}{m}}$$

□

We can now provide a general purpose bound for all two layer ReLU networks simply by choosing a value for μ:

Theorem 7. For any $h \geq 2, \gamma > 0, \delta \in (0,1)$ and $U^0 \in \mathbb{R}^{h \times d}$, the generalization error of a function $f(x) = V[Ux]_+ \in \mathcal{F}$ is bounded by

$$L_0(f) \leq \widehat{L}_\gamma(f) + \widetilde{O}\left(\frac{\sqrt{c}\,\|V\|_F\,(\|U - U^0\|_F + \|U^0\|_2)\,\|X\|_F}{\gamma m} + \sqrt{\frac{h}{m}}\right)$$

Proof. In the previous lemmas, we had an open parameter μ to control the tradeoff between the tightness of the covering and the size of the covering. Now, set $\mu = \frac{3\sqrt{2}}{4} - 1$. This leads to the following bound on N_h:

$$\ln N_h = \ln\binom{\lceil h/\mu \rceil + h - 1}{h-1} \leq \ln\left(\left[e\frac{\lceil h/\mu \rceil + h - 1}{h-1}\right]^{h-1}\right) \leq (h-1)\ln\left(e\frac{\lceil h/\mu \rceil + h - 1}{h-1}\right)$$
$$\leq (h-1)\ln\left(e + e\frac{\lceil h/\mu \rceil}{h-1}\right) \leq h\ln(\epsilon + 2\epsilon/\mu) \leq 5h$$

Plugging this into Lemma 5 and using the $\widetilde{O}$ to hide constants yields the result. (You also need the inequality $\|U^0 X\|_F \leq \|U^0\|_2 \|X\|_F$.) □

4.1 Experimental Performance of the Bound

The authors calculate the experimental performance of their bound. As shown in the right panel of 6, it does in fact decrease with higher h. This is in contrast to bounds obtained by several other authors. Figure 6 shows that, while other author's bounds may be valid, they fail to explain the overparameterization phenomenon because they start to increase as the size of the model balloons. With the bound from Theorem 7 however, we may finally have caught the scent. Also note that the term $\|V\|_F \|U - U^0\|_F$, which includes the main measures we identified in the Section 1, dominates the other term. Interestingly, my experiments show that replacing $\|U^0\|_2$ with $\|U^0\|_F$ causes the bound to increase rapidly– in this case dominating $\|V\|_F \|U - U^0\|_F$ and causing the whole bound to increase with h. Thus, the $l2$ norm is still an improtant part of the bound

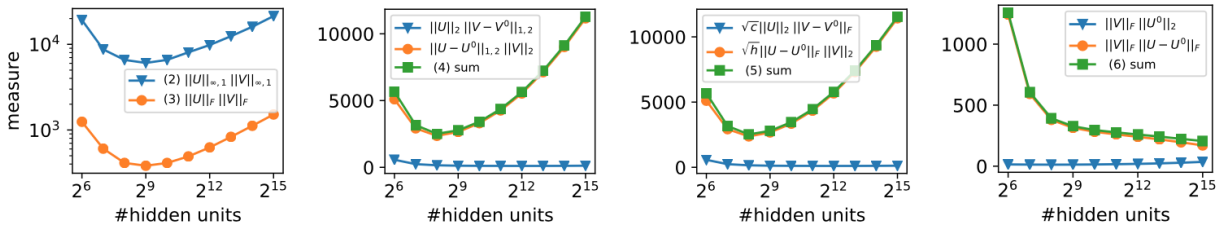

Figure 6: Performance of various bounds on generalization error. That of Neyshabur et al. (right panel) is the only one which actually decreases with high h.

4.2 Other Results

The authors present two further results about this bound that I do not have space to prove. Even though experimental results confirm that the bound decreases as h increases, they are still bothered by $\sqrt{h}$ term. For very large h, it would be nice if this dependence were not present in the bound. By redoing the covering arguments to use l_p balls instead of l_2, the authors are able to find a bound similar to Theorem 7 without it:

$$L_0(f) \leq \widehat{L}_\gamma(f) + \widetilde{O}\left(\frac{\sqrt{c} h^{\frac{1}{2}-\frac{1}{p}} \left\|V^\top\right\|_{p,2} (h^{\frac{1}{2}-\frac{1}{p}} \|U - U^0\|_{p,2} \|X\|_F + \|U^0 X\|_F)}{\gamma m} + \sqrt{\frac{e^{-p} h}{m}}\right)$$

Here, $\|\cdot\|_{p,2}$ is the l_p norm of the row l_2 norms. Setting $p = \ln h$ eliminates the dependence on h without badly weakening the other parts of the bound.
Second, the authors prove a lower bound on the Rademacher complexity of $\mathcal{F}_W$ that matches the dominant term in Theorem 7. Specifically,

$$\mathcal{R}_D(\mathcal{F}_w) = \Omega\left(\frac{\sum_{j=1}^h \alpha_j \beta_j \|X\|_F}{m}\right)$$

Here, the $\alpha_j \beta_j \|X\|_F$ corresponds to $\|V\|_F \|U - U^0\|_F \|X\|_F$ from Theorem 7. This shows that the bound is tight at least with respect to the most important terms.

5 Further Experiments

In this section I briefly describe the experiments and some futher avenues of exploration. The authors trained models on the CIFAR-10, SVHN, and MNIST datasets. They used SGD with momentum 0.9 and a fixed step size (0.01 for MNIST, 0.001 for CIFAR-10 and SVHN). They did not use weight decay, dropout, or batch normalization, and training was stopped when the cross-entropy reached 0.01. I replicated these parameters in my experiments on MNIST. Some more of my results are included in Figure 7. They closely match the original paper.

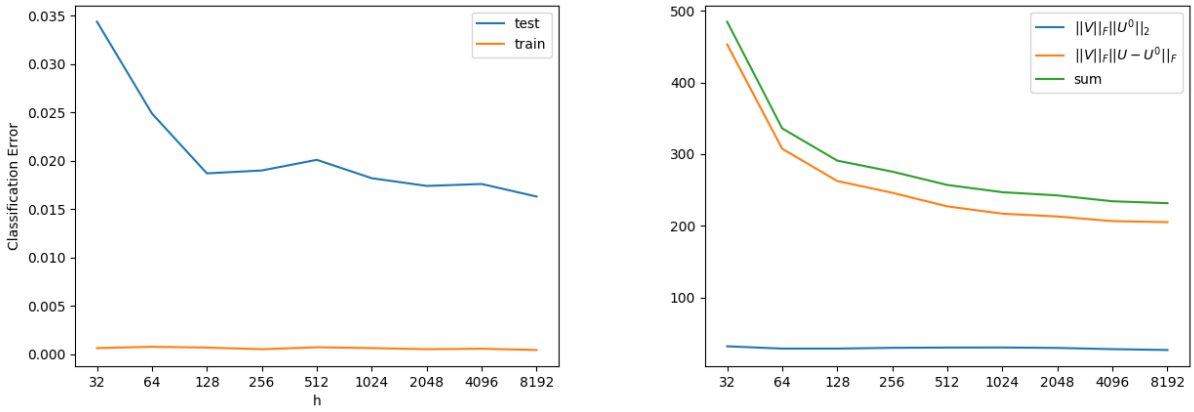

Figure 7: Main results on MNIST. The left panel plots the training and test error against h, showing continued improvement in test error even when the training set is learned perfectly. The right panel shows the performance of the bound we derived, showing that it does in fact decrease with h and that the dominant term is $\|V\|_F \|U - U^0\|_F$.

I also tried training the model with a different optimizer. When I used Adam with default parameters (learning rate of 0.001) to train on MNIST, I got quite different results. See Figure 8 for details. Whatever the explanation, this experiment suggests that care must be taken when generalizing these results. The model may not always learn weights with low capacity and impact scores. Perhaps the training rate has to be properly calibrated for this to work, or perhaps Adam is over-eager to keep changing weights compared to SGD, actually leading it away from the simplest (i.e. lowest complexity) solution (although it still generalizes just as well as SGD).

I also tried extending the experiment to 3-layer ReLU networks. I included two fully-connected hidden layers, each of size h and trained with SGD as before. When treating the middle layer as U and the third layer as V, the bound worked unchanged (see Figure 95).

However, when treating the first layer as U and the second as V, the bound grew with large h. The problem seems to be the middle layer. This middle transformation matrix connects two layers of size h, so it has h^2 entries. Thus, to get the norm of this matrix to decay, the individual units would have to decay faster than $1/\sqrt[4]{h}$. Evidently, they do not, but if you subtract $V - V^0$ then they do! (See Figure) One other way to generalize the bound would be to treat the first layer as U and the last layer as V, ignoring the troublesome middle layer entirely. This also seems to work. We conceptualized a two-layer network as a feature extractor plus a logistic classifier—it is more difficult to transfer this analogy to the three layer case. The first and second layers both seem to act sort of like feature extractors. But the third layer does not act quite like the top layer of the original network, either (see Figure 10). More attention is needed to extend the techniques of this paper to deeper networks. This is of great interest since after all, much of the over-parameterization in modern machine learning comes from having extremely deep networks.

6 Conclusion

The main contribution of Neyshabur et al. is to provide a new bound on the generalization error of 2-layer neural networks and to prove empirically that the bound tightens as the size of the model grows. The authors find that taking the difference between the trained and untrained weight matrix to be a fruitful measure of complexity. The key innovation in the proof is to split up the complexity by hidden unit, analyzing the contribution of each one separately rather treating a layer as a unit. Another important move was to find a covering lemma that allowed the argument to generalize with high probability. The empirical results show that stochastic gradient descent tends to learn certain kinds of weight matrices when the model gets big (though my experiments suggest this might not be universally true), and the theoretical results show that these same kindso f weight matrices correspond with low Rademacher complexity and good generalization properties. More research is needed to understand how and why optimization algorithms converge to these low complexity networks. But these results are an important step toward understanding the mystery behind neural networks' success.

References

[1] Andreas Maurer. A vector-contraction inequality for rademacher complexities. *Algorithmic Learning Theory*, page 317, 2016.

[2] Behnam Neyshabur, Zhiyuan Li, Srinadh Bhojanapalli, Yann LeCun, and Nathan Srebro. Towards understanding the role of over-parametrization in generalization of neural networks, 2018. (Preprint, `https://arxiv.org/abs/1805.12076`).

[3] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization, 2016. (Preprint, `https://arxiv.org/abs/1611.03530`).

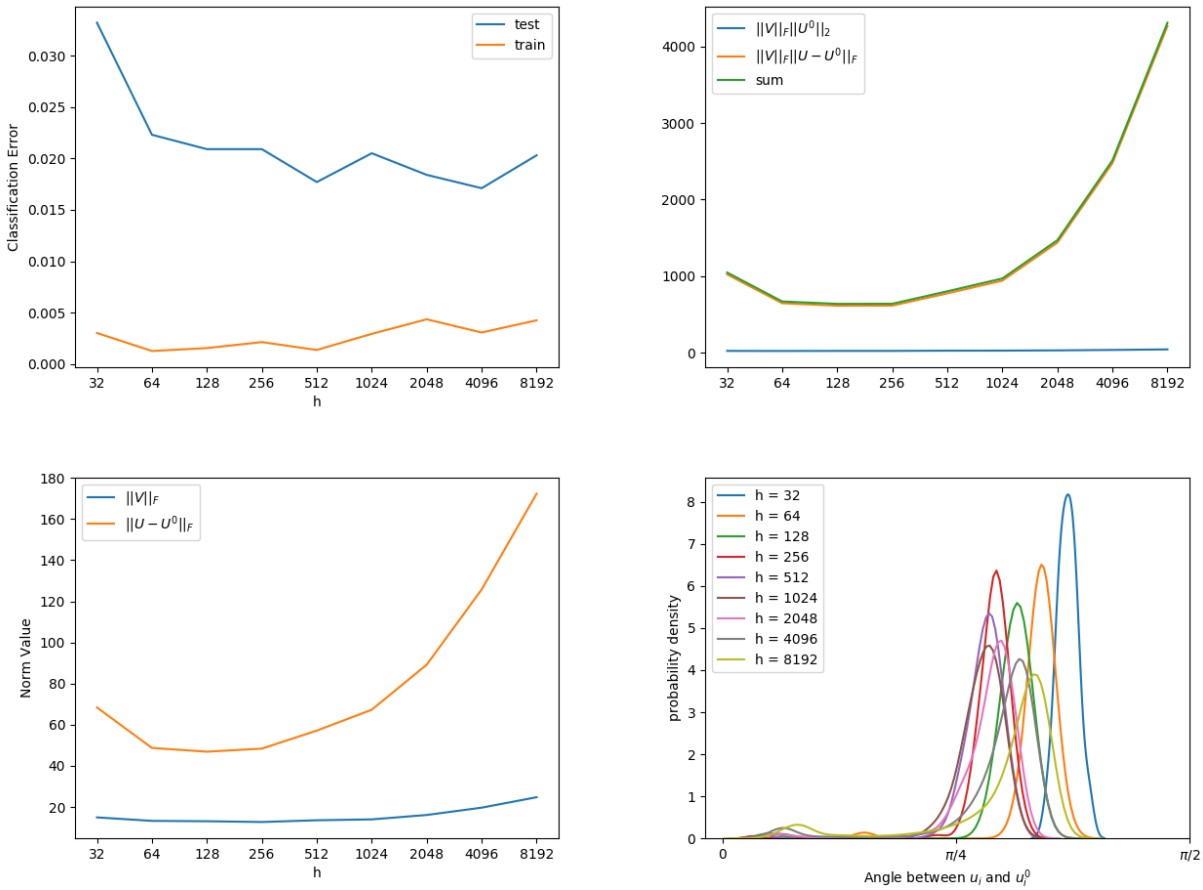

Figure 8: Results on MNIST when training with Adam with learning rate = 0.001. As shown in the top right panel, the bound now increases with h. While all parts of the bound are now increasing, the bottom left panel shows that the growth is largely driven by $\|U - U^0\|_F$. In the bottom right panel, which shows distributions of angles between u_i and u_i^0, the plots are jumbled together, further suggesting that something went wrong when training the first layer (compare with Figure 4). The error plot (top left) doesn't look quite right either, as testing error never quite gets to 0 and actually ticks up for large models.

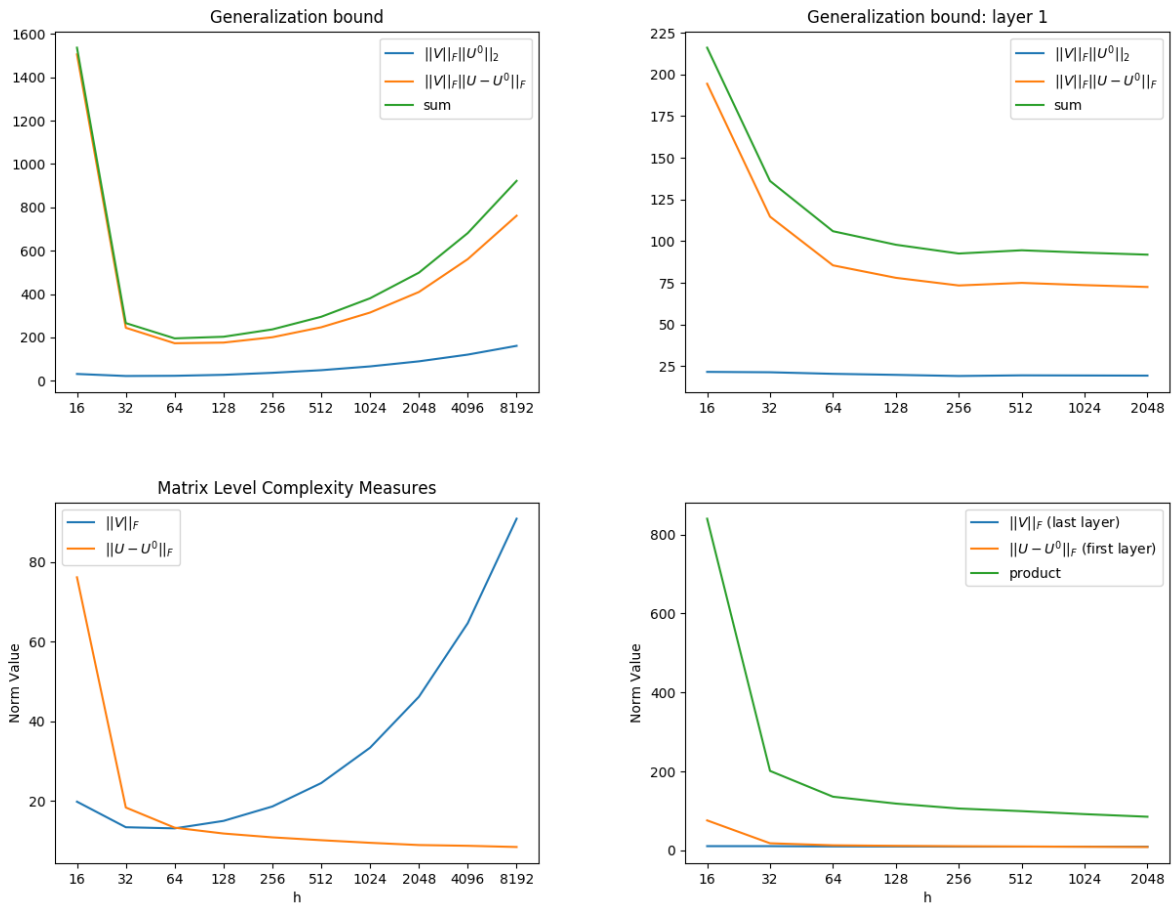

Figure 9: Matrix level norms for 3 layer ReLU. Top left panel treats the first layer as U and the second as V, top right treats the second layer as U and the third as V, bottom right treats the first lyaer as U and the third layer as V. Lower left shows that the norm of the middle layer increases with larger h. But the other bounds show that if we first subtract the initial value V^0, it decreases.

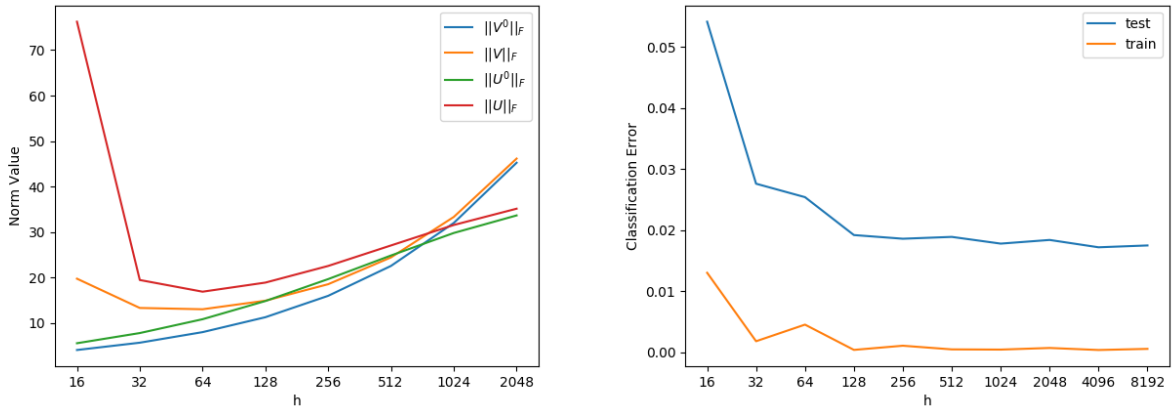

Figure 10: Left: Initial and final norms of the second layer (U) and third layer (V). The growth in the norm of the second layer appears driven by the growth in the norm of its initial value, much like U was in a 2-layer network. But the third layer does not act like V did in a 2-layer network, because its norm also grows significantly (compare to Figure 3). Right: error plot for 3-layer network. This looks a lot like the 2-layer version.