Lipschitz Constants of Functions of Neural Networks

Sunjoong Kim

Contents

Abstract

Neural networks can be thought of as a function between euclidean spaces. The *Lipschitz constant* of the neural nework measures how sensitive the output is, with respect to the input. Therefore, we can use the constant to access the robustness of the machine learning architecture, as in the Wasserstein GAN. The Lipschitz constant of the simplest function, or the linear map, is the *operator norm* of the matrix which is the absolute value of the dominating eigenvalue. For a differentiable function in general, the Lipscthiz constant is related to the operator norm of the Jacobian matrix. However, evaluating the exact value of the constant is difficult or often impossible. Rather, we can think of a systematic way called the *AutoLip* to find an upper bound for the Lipschitz constant. We can apply this algorithm to multilayer perceptrons and convolutional neural networks, where the power method is introduced to approximate the operator norms of big matrices.

Chapter 1

Introduction

Deep neural networks have made many advances including computer vision, language modeling, machine translation and text and picture generating. Still, one of the difficulties in applying deep learning algorithms to reality is that the algorithm often lacks its robustness. As an example, Szegedy et al. have found that a small perturbation of an input for true image may cause the classifier to misclasssify the image as false image [2].

To overcome the instabiltiy of training and to enhance robustness of generating models such as GANs, M. Arjovsky et al. proposed Wasserstein distance between distributions and restrict their attention to 1-Lipschitz critic(discriminator) [3], [4]. As this example suggest, the Lipschitz constant can be a good metric to access the robustness of the algorithm.

The Lipschitz constant $\mathrm{Lip}(f)$ of a function f measures the sensitiveness or the rate of changes of the function. In Chapter 2, we define the Lipschitz constant of the functions between euclidean spaces and explore the computation of this optimal constant for various functions including linear maps, affine maps and compositions of functions. However, if the function becomes more complex, it is difficult or almost impossible to calculate the optimal constant. Instead of finding the exact value, we propose algorithms to estimate the constant in Chapter 3.

Chapter 2

Lipschitz Constants of Functions Between Euclidean Spaces

2.1 Lipschitz Constants

For vectors $\boldsymbol{x} = [x_1 \ \cdots \ x_n]^T$ and $\boldsymbol{y} = [y_1 \ \cdots \ y_m]^T$ in Euclidean spaces $\mathbb{R}^n$ and $\mathbb{R}^m$, respectively, $||\boldsymbol{x}||$ and $||\boldsymbol{y}||$ are the usual standard norms of $\boldsymbol{x}$ and $\boldsymbol{y}$ defined by

$$||x|| = \sqrt{{x_1}^2 + \cdots + {x_n}^2}, \quad ||y|| = \sqrt{{y_1}^2 + \cdots + {y_m}^2}. \tag{2.1.1}$$

Definition 2.1. A function $f : \mathbb{R}^n \to \mathbb{R}^m$ is called *Lipschitz continuous* if there is a nonnegative real number c such that

$$||f(\boldsymbol{x}) - f(\boldsymbol{y})|| \leq c||\boldsymbol{x} - \boldsymbol{y}|| \qquad (\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n) \tag{2.1.2}$$

The infimum of c which satisfies (2.1.2) is called the *Lipschitz constant* of f and is denoted by $\mathrm{Lip}(f)$.

This definition can also be extended to functions from a metric space to another metric space if we replace the norms by the distances. It is useful to know that there is an equivalent definition ;

$$\mathrm{Lip}(f) = \sup_{\boldsymbol{x} \neq \boldsymbol{y}} \frac{||f(\boldsymbol{x}) - f(\boldsymbol{y})||}{||\boldsymbol{x} - \boldsymbol{y}||} \tag{2.1.3}$$

To show the equivalence of the two definitions, let

$$\begin{aligned} L_1 &= \inf\{c \geq 0 : ||f(\boldsymbol{x}) - f(\boldsymbol{y})|| \leq c||\boldsymbol{x} - \boldsymbol{y}||, \quad \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n\} \\ L_2 &= \sup\left\{\frac{||f(\boldsymbol{x}) - f(\boldsymbol{y})||}{||\boldsymbol{x} - \boldsymbol{y}||} : \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n, \boldsymbol{x} \neq \boldsymbol{y}\right\}. \end{aligned} \tag{2.1.4}$$

Let $\mathcal{L}_1$ and $\mathcal{L}_2$ be two the sets in the definition of L_1 and L_2, respectively. Then, $\mathcal{L}_2 \subset \mathcal{L}_1$ since

$$\mathcal{L}_2 = \{c \geq 0 : ||f(\boldsymbol{x}) - f(\boldsymbol{y})|| = c||\boldsymbol{x} - \boldsymbol{y}||, \quad \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n, \boldsymbol{x} \neq \boldsymbol{y}\},$$

and it follows that $L_1 \leq L_2$. Pick $c \in \mathcal{L}_1$. Then, $||f(\boldsymbol{x}) - f(\boldsymbol{y})|| \leq c||\boldsymbol{x} - \boldsymbol{y}||$ for all $\boldsymbol{x}$ and $\boldsymbol{y}$ in $\mathbb{R}^n$. Assuming $\boldsymbol{x} \neq \boldsymbol{y}$, we have

$$\frac{||f(\boldsymbol{x}) - f(\boldsymbol{y})||}{||\boldsymbol{x} - \boldsymbol{y}||} \leq c.$$

Taking supremum for all $\boldsymbol{x}$ and $\boldsymbol{y}$ ($\boldsymbol{x} \neq \boldsymbol{y}$), we have $L_2 \leq c$. and taking infimum for all $c \leq \mathcal{L}_1$ yields $L_2 \leq L_1$. Thus the two definitions are indeed equivalent.

The optimal Lipschitz constant $\mathrm{Lip}(f)$ is not only the infimum, but also the minimum of L satisfying (2.1.2). To show this, it suffices to prove that $\mathrm{Lip}(f) \in \mathcal{L}_1$, or that $c = \mathrm{Lip}(f)$ satisfies the condition. Let $\boldsymbol{x}$, $\boldsymbol{y} \in \mathbb{R}^n$. If $\boldsymbol{x} = \boldsymbol{y}$, then (2.1.2) holds trivially. Assuming $\boldsymbol{x} \neq \boldsymbol{y}$, we have

$$\frac{||f(\boldsymbol{x}) - f(\boldsymbol{y})||}{||\boldsymbol{x} - \boldsymbol{y}||} \leq \mathrm{Lip}(f)$$

because of the second definition (2.1.3) of $\mathrm{Lip}(f)$. Multiplying both sides by

$||\boldsymbol{x}-\boldsymbol{y}||$, we can conclude that $c = \mathrm{Lip}(f)$ satisfies the condition (2.1.2).

Note also that $\mathrm{Lip}(f) = 0$ if and only if f is a constant function. Suppose that f is a constant function. Then, the condition (2.1.2) holds vacuously and $\mathcal{L}_1 = [0, \infty)$; $\mathrm{Lip}(f) = 0$. Suppose, on the contrary, that $\mathrm{Lip}(f) = 0$. Pick $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$. Since $0 \in \mathcal{L}_1$, we have $||f(\boldsymbol{x}) - f(\boldsymbol{y})|| = 0$. Thus, $f(\boldsymbol{x}) = f(\boldsymbol{y})$ and f is a constant function.

2.2 Real-valued Functions of a Real Variable

For univariate real function $f : \mathbb{R} \to \mathbb{R}$, the above condition (2.1.2) is equivalent to saying that the average rate of change is upper bounded by c;

$$\left| \frac{f(x) - f(y)}{x - y} \right| \leq c. \tag{2.2.5}$$

If f is diffrentiable everywhere in the domain, the mean value theorem guarantees that this is equivalent to

$$|f'(x)| \leq c \tag{2.2.6}$$

for every $x \in \mathbb{R}$. Moreover, $\mathrm{Lip}(f)$ is the least nonnegative number c satisfying the condition (2.2.6) for all x and we can express $\mathrm{Lip} f$ in its exact form ;

$$\mathrm{Lip}(f) = \sup_{x \in \mathbb{R}} |f'(x)| \tag{2.2.7}$$

In this sense, we can easily evaluate the optimal Lipschitz constant of the sigmoid function

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{2.2.8}$$

and the hyperbolic tangent function

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}. \tag{2.2.9}$$

Since $0 < \sigma(x) < 1$ and $\sigma'(x) = \sigma(x)\,(1 - \sigma(x))$, we have $\sigma'(x) < \frac{1}{4}$ so that

$$\mathrm{Lip}(\sigma) = \frac{1}{4}. \tag{2.2.10}$$

Since $\tanh(x) = 2\sigma(2x) - 1$, we have

$$\tanh'(x) = 4\sigma'(2x) < 1$$

and

$$\mathrm{Lip}(\tanh) = 1. \tag{2.2.11}$$

The ReLU function

$$\mathrm{ReLU}(x) = \max\{0, x\} \tag{2.2.12}$$

is not differentiable everywhere, but we can use (2.2.5) to conclude

$$\mathrm{Lip}(\mathrm{ReLU}) = 1. \tag{2.2.13}$$

Here is a list of the optimal Lipschitz constants of univariate activation functions, frequently used in machine learning and deep learning. (Table 2.1)

In the definition of $\mathrm{Lip}(f)$ in (2.1.3), we can't replace the supremum by the maximum. Consider a function ; $f(x) = \ln(1 + e^x)$. This function is called the softplus activation function and is an antiderivative of the sigmoid function. It is a monotonically increasing function whose derivative is also increasing, with an assymptote $y = x$. Thus, $\mathrm{Lip}(f) = 1$, but no x and y satisfy the equality $\frac{f(x)-f(y)}{x-y} = 1$.

Table 2.1: The Lipschitz constants of univariate activation functions.

Activation Functions	Formula	Lip(f)
Sigmoid	$\sigma(x) = \frac{1}{1+e^{-x}}$	$\frac{1}{4}$
Hyperbolic tangent	$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$	1
Rectified Linear Unit	$\mathrm{ReLU}(x) = \max\{0, x\}$	1
Leaky ReLU	$\mathrm{LReLU}(x) = \max\{\alpha x, x\}$	1
Exponential Linear Unit	$\mathrm{ELU}(x) = \begin{cases} x & (x \geq 0) \\ \alpha(e^x - 1) & (x < 0) \end{cases}$	1
Softplus	$f(x) = \frac{1}{\beta}\log(1 + e^{\beta x})$	1
Gaussian	$g(x) = e^{-x^2}$	$\sqrt{\frac{2}{e}}$

2.3 Linear Functions

2.3.1 Operator Norms $||W||$

Consider a linear function $f(\boldsymbol{x}) = W\boldsymbol{x}$ for some $m \times n$ matrix W. Because of the linearity, the condition (2.1.2) reduces to

$$||W\boldsymbol{x}|| \leq c||\boldsymbol{x}||. \qquad (\boldsymbol{x} \in \mathbb{R}^n) \tag{2.3.14}$$

The smallest c which satisfies (2.3.14) is called *the operator norm of* W and denoted by $||W||$. Thus, the optimal Lipschitz constant of f equals the operator norm of W ; $||W|| = \mathrm{Lip}(f)$. The definitions of the forms (2.1.4) are as follows;

$$\begin{aligned} ||W|| &= \inf\{c \geq 0 \ : \ ||W\boldsymbol{x}|| \leq c||\boldsymbol{x}||\} \\ &= \sup\left\{\frac{||W\boldsymbol{x}||}{||\boldsymbol{x}||} \ : \ \boldsymbol{x} \in \mathbb{R}^n, \ \boldsymbol{x} \neq \boldsymbol{0}\right\} \end{aligned} \tag{2.3.15}$$

If we modify the second definition of (2.3.15) to

$$||W|| = \sup\left\{||W\boldsymbol{x}|| : ||\boldsymbol{x}|| = 1\right\}, \tag{2.3.16}$$

we can easily verify that the supremum is actually the maximum in this case. Because the set $\{\boldsymbol{x} : ||\boldsymbol{x}|| = 1\}$ is a compact subset of $\mathbb{R}^m$ and the map $\boldsymbol{x} \mapsto ||W\boldsymbol{x}||$ is continuous, the set in (2.3.16) has its maximum.

The operator norm is indeed a norm of a vector space ;

Proposition 2.2. *The operator $||\cdot|| : W \mapsto ||W||$ satisfies the following three properties. Thus, the set $\mathcal{M}_{m,n}(\mathbb{R})$ of all m by n real matrices is a normed vector space with respect to this norm ;*

(a) $||W|| \geq 0$ *for all* $W \in \mathcal{M}_{m,n}(\mathbb{R})$ *;* $||W|| = 0$ *if and only if* $W = 0$*.*

(b) $||kW|| = |k| \cdot ||W||$ *for all* $W \in \mathcal{M}_{m,n}(\mathbb{R})$ *and* $k \in \mathbb{R}$*.*

(c) $||W + V|| \leq ||W|| + ||V||$ *for all* $W, V \in \mathcal{M}_{m,n}(\mathbb{R})$*.*

For (a), that $||W|| \geq 0$ is obvious. If $||W|| = 0$, then $||W\boldsymbol{x}|| \leq 0||\boldsymbol{x}|| = 0$ for all $\boldsymbol{x} \in \mathbb{R}^n$. It follows that $W\boldsymbol{x} = 0$ for all $\boldsymbol{x} \in \mathbb{R}^n$, by the definition of vector norm, for which $W = 0$ Suppose, on the other hand, that $W = 0$. Then $W\boldsymbol{x} = 0$ for all $x \in \mathbb{R}^n$. Thus, $c = 0$ is the minimal value satisfying (2.3.14). To prove (b), we have

$$\begin{aligned}
||kW|| &= \inf\left\{c : ||kWx|| \leq c||x||\right\} \\
&= \inf\left\{c : |k| \cdot ||Wx|| \leq c||x||\right\} \\
&= \inf\left\{c : ||Wx|| \leq \frac{c}{|k|}||x||\right\} \\
&= \inf\left\{|k|b : ||Wx|| \leq b||x||\right\} \\
&= |k| \inf\left\{b : ||Wx|| \leq b||x||\right\} \\
&= |k| \cdot ||W||.
\end{aligned}$$

Now, consider (c). For any $\boldsymbol{x} \in \mathbb{R}^n$, we have

$$\begin{aligned}||(W+V)\boldsymbol{x}|| &= ||W\boldsymbol{x}+V\boldsymbol{x}|| \leq ||W\boldsymbol{x}|| + ||V\boldsymbol{x}|| \\ &\leq ||W||\,||\boldsymbol{x}|| + ||V||\,||\boldsymbol{x}|| = (||W||+||V||)||\boldsymbol{x}||\end{aligned}$$

By the minimality of $||W+V||$, we have $||W+V|| \leq ||W|| + ||V||$.

An analogue of (c) also holds for multiplication; if $W \in \mathcal{M}_{m,n}(\mathbb{R})$ and $V \in \mathcal{M}_{n,k}(\mathbb{R})$, then

$$||WV|| \leq ||W||\,||V||. \tag{2.3.17}$$

This is because of the inequality

$$||(WV)\boldsymbol{x}|| = ||W(V\boldsymbol{x})|| \leq ||W||\,||V\boldsymbol{x}|| \leq ||W||\,||V||\,||\boldsymbol{x}||$$

and the minimality of $||WV||$.

2.3.2 Evaluation of $||W||$ for square matrices

First, consider the case when W is a square matrix so that $f : \mathbb{R}^n \to \mathbb{R}^n$. Let $\lambda_1, \cdots, \lambda_n$ be (possibly repeated) eigenvalues of W and let $\tilde{\lambda} = \max\{|\lambda_i| : i = 1, \cdots, n\}$. ($\tilde{\lambda}$ is sometimes called the *dominating eigenvalue*, especially in the context of the power method : Subsection 3.4.2) Then, the following inequality holds in general ;

$$\tilde{\lambda} \leq ||W||. \tag{2.3.18}$$

For a proof, pick an eigenvalue λ_i and the corresponding eigenvector $\boldsymbol{x}_i$ which is nonzero. Since $W\boldsymbol{x}_i = \lambda_i \boldsymbol{x}_i$, Then, we have $||W\boldsymbol{x}_i|| = |\lambda_i|\,||\boldsymbol{x}_i||$ and

$$|\lambda_i| = \frac{||W\boldsymbol{x}_i||}{||\boldsymbol{x}_i||} \leq ||W||.$$

Taking the maximum over i, we have the desired inequality. The above inequality (2.3.18) becomes equality when W is symmetric.

Suppose that W is real symmetric. Then, W is orthogonally diagonalizable in the sense that, there exists an orthogonal matrix

$$V = [\boldsymbol{v}_1 \quad \cdots \quad \boldsymbol{v}_n],$$

such that

$$W = V\Lambda V^T,$$

where $\Lambda = \mathrm{diag}\{\lambda_1, \cdots, \lambda_n\}$ and λ_i is an eigenvalue of W with the corresponding eigenvector $\boldsymbol{v}_i$. Note that $\{\boldsymbol{v}_1, \cdots, \boldsymbol{v}_n\}$ forms an orthonormal basis for $\mathbb{R}^n$. For any nonzero $\boldsymbol{x} \in \mathbb{R}^n$, there exist real numbers c_1, $\cdots$, c_n such that $\boldsymbol{x} = c_1\boldsymbol{v}_1 + \cdots + c_n\boldsymbol{v}_n$. Since

$$\begin{aligned} W\boldsymbol{x} &= c_1 W\boldsymbol{v}_1 + \cdots + c_n W\boldsymbol{v}_n \\ &= c_1\lambda_1\boldsymbol{v}_1 + \cdots + c_n\lambda_n\boldsymbol{v}_n, \end{aligned}$$

we have

$$\begin{aligned} ||W\boldsymbol{x}||^2 &= ||c_1 W\boldsymbol{v}_1 + \cdots + c_n W\boldsymbol{v}_n||^2 \\ &= |c_1\lambda_1|^2||\boldsymbol{v}_1||^2 + \cdots + |c_n\lambda_n|^2||\boldsymbol{v}_n||^2 \\ &= {c_1}^2{\lambda_1}^2 + \cdots + {c_n}^2{\lambda_n}^2. \end{aligned}$$

Thus,

$$\frac{||W\boldsymbol{x}||^2}{||\boldsymbol{x}||^2} = \frac{{c_1}^2{\lambda_1}^2 + \cdots + {c_n}^2{\lambda_n}^2}{{c_1}^2 + \cdots + {c_n}^2} \leq \tilde{\lambda}^2,$$

for which $||W\boldsymbol{x}||/||\boldsymbol{x}|| \leq \tilde{\lambda}$. Therefore, we have the reverse inequality $||W|| \leq \tilde{\lambda}$.

As examples consider the following matrices W_1, W_2, W_3, W_4, W_5 defined by

$$W_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad W_2 = \begin{bmatrix} 2 & 0 \\ 0 & 3 \end{bmatrix}, \quad W_3 = \begin{bmatrix} 4 & 2 \\ 2 & 7 \end{bmatrix},$$

$$W_4 = \begin{bmatrix} 1 & 3 \\ 2 & 0 \end{bmatrix}, \quad W_5 = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}.$$

W_1 is the identity matrix and thus $||W|| = 1$. Indeed, the characteristic polyinomial of W_1 is $(\lambda - 1)^2 = 0$, $\lambda_1 = \lambda_2 = 1$ and thus the maximum absolute value of them is 1. The matrix W_2 stretches a given vector $\boldsymbol{x}$ twice in the x-axis direction and three times in the y-axis direction. Thus, $||W_2||$ should be 3. Indeed, $|W_2 - \lambda I| = (\lambda - 2)(\lambda - 3)$ and $\tilde{\lambda} = 3$. W_3 is the last example that is symmetric ; $|W_3 - \lambda I| = (\lambda - 3)(\lambda - 8)$ and $\tilde{\lambda} = 8$.

W_4 is not symmetric and we can't evaluate $||W_4||$ for now. Still, we have, $\lambda_1 = -2$, $\lambda_2 = 3$ and $||W_4|| \geq 3$ by (2.3.18). The reverse inequality is not valid since the eigenvectors $x_1 = [1 \ \ -1]^T$ and $x_2 = [3 \ \ 2]^T$ are not perpendicular. W_5 is not symmetric either, but we can postulate that $||W_5|| = 1$ since

$$W_5\boldsymbol{x} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} x_2 \\ 0 \end{bmatrix}$$

and

$$\frac{||W_5\boldsymbol{x}||}{||\boldsymbol{x}||} = \frac{|x_2|}{\sqrt{{x_1}^2 + {x_2}^2}} \leq 1.$$

Since W_5 has the only eigenvector 0, the inequality (2.3.18) still holds.

Here is a short description of what we've done in this subsection.

Lemma 2.3. *Let $W \in \mathcal{M}_n(\mathbb{R})$ be a real symmetric matrix. Then, the operator norm $||W||$ is the maximal absolute eigenvalue of W.*

2.3.3 Evaluation of $||W||$ for rectangular matrices

Almost all matrices that we encounter in machine learning problem are neither symmetric matrices, nor are they square matrices. But, we can always calculate the Lipschitz constant or the operator norm of any rectangular matrices.

Theorem 2.4. *[5] Let $W \in \mathcal{M}_{m,n}(\mathbb{R})$ be an m by n real matrix. Then*

$$||W|| = \sqrt{||W^T W||}. \tag{2.3.19}$$

Note that the norm on the left hand side is the operator norm on $\mathcal{M}_{m,n}(\mathbb{R})$ and the norm on the right hands side is the operator norm on $\mathcal{M}_n(\mathbb{R})$. Note also that the matrix $W^T W$ is symmetric in that $(W^T W)^T = W^T (W^T)^T = W^T W$ and we are always able to calculate the norm of $||W^T W||$ by making use of the lemma 2.3.

Before the proof of (2.3.19), we illustrate elementary properties of the operator norm of matrices. The euclidean norm $||\boldsymbol{x}||$ of a vector $\boldsymbol{x} \in \mathbb{R}^n$ in (2.1.1) can also be defined by means of the inner product on $\mathbb{R}^n$;

$$||\boldsymbol{x}|| = \sqrt{\langle \boldsymbol{x}, \boldsymbol{x} \rangle}. \tag{2.3.20}$$

The inner product $\langle \boldsymbol{x}, \boldsymbol{y} \rangle = \boldsymbol{x}^T \boldsymbol{y}$ has the following properties. (we omit the proofs of them.)

$$\begin{aligned} \langle \boldsymbol{x}, W\boldsymbol{y} \rangle &= \langle W^T \boldsymbol{x}, \boldsymbol{y} \rangle, \\ \langle k\boldsymbol{x}, \boldsymbol{y} \rangle &= k \langle \boldsymbol{x}, \boldsymbol{y} \rangle, \\ |\langle \boldsymbol{x}, \boldsymbol{y} \rangle| &\leq ||\boldsymbol{x}|| \, ||\boldsymbol{y}||. \end{aligned}$$

Now we are ready for the proof of (2.3.19). If $W = 0$, then the equation

(2.3.19) holds trivially. Suppose that $W \neq 0$. Let $\boldsymbol{x} \in \mathbb{R}^n$. Then

$$\begin{aligned} ||W\boldsymbol{x}||^2 &= \langle W\boldsymbol{x}, W\boldsymbol{x} \rangle \\ &= \langle W^T W\boldsymbol{x}, \boldsymbol{x} \rangle \\ &\leq ||W^T W\boldsymbol{x}|| \, ||\boldsymbol{x}|| \\ &\leq ||W^T W|| \, ||\boldsymbol{x}||^2, \end{aligned}$$

and thus

$$||W\boldsymbol{x}|| \leq \sqrt{||W^T W||} \, ||\boldsymbol{x}||.$$

for all $\boldsymbol{x} \in \mathbb{R}^n$. By the minimality of $||W||$, we have

$$||W|| \leq \sqrt{||W^T W||}$$

Squaring both sides and making use of (2.3.17) yield

$$||W||^2 = ||W^T W|| \leq ||W^T|| \, ||W||. \tag{2.3.21}$$

Since $||W|| \neq 0$, we have

$$||W|| \leq ||W^T||.$$

Substituting W by its transpose, we get the reverse inequality. Therefore,

$$||W|| = ||W^T||. \tag{2.3.22}$$

By (2.3.22), the equation (2.3.21) reduces to

$$||W||^2 = ||W^T W||,$$

and this proves (2.3.19).

For example, we can evaluate the operator norm of W_4 and W_5 in the previous subsection. We have

$$W_4{}^T W_4 = \begin{bmatrix} 5 & 3 \\ 3 & 9 \end{bmatrix}, \quad W_5{}^T W_5 = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}.$$

$W_4{}^T W_4$ has its eigenvalues $7 \pm \sqrt{13}$, thus $||W_4|| = \sqrt{7+\sqrt{13}}$. And $W_5{}^T W_5$ has eigenvalues 0 and 1 ; $||W_5|| = \sqrt{1} = 1$ as expected. As a rectangular matrix, we may think of

$$W_6 = \begin{bmatrix} 1 & 0 \\ 2 & 1 \\ 0 & 1 \end{bmatrix}.$$

Since

$$W_6{}^T W_6 = \begin{bmatrix} 5 & 2 \\ 2 & 2 \end{bmatrix},$$

$\lambda = 1, 6$ and $||W_6|| = \sqrt{6}$.

2.4 Affine Functions

An affine function

$$f(\boldsymbol{x}) = W\boldsymbol{x} + \boldsymbol{b} \tag{2.4.23}$$

has the same optimal Liptschitz constant as that of its linear part ;

$$\mathrm{Lip}(f) = ||W|| \tag{2.4.24}$$

This is because the condition (2.1.2)

$$||f(\boldsymbol{x}) - f(\boldsymbol{y})|| \leq c||\boldsymbol{x} - \boldsymbol{y}||$$

applied to (2.4.23) becomes

$$||W(\boldsymbol{x}-\boldsymbol{y})|| \leq c||\boldsymbol{x}-\boldsymbol{y}||.$$

2.5 Elementwise Application of Nonlinear Functions

Let $f : \mathbb{R} \to \mathbb{R}$ be any function and let $F : \mathbb{R}^m \to \mathbb{R}^m$ be applying f to each element of the input $\boldsymbol{x} = [x_1 \ \cdots \ x_m]^T$:

$$F(\boldsymbol{x}) = F\left(\begin{bmatrix} x_1 \\ \vdots \\ x_m \end{bmatrix}\right) = \begin{bmatrix} f(x_1) \\ \vdots \\ f(x_m) \end{bmatrix}. \tag{2.5.25}$$

If f is a Liptschitz continuous function with its optimal constant L. Then,

$$\begin{aligned} ||F(\boldsymbol{x}) - F(\boldsymbol{y})||^2 &= \left|\left|\begin{bmatrix} f(x_1)-f(y_1) \\ \vdots \\ f(x_m)-f(y_m) \end{bmatrix}\right|\right|^2 = \sum_{i=1}^{m} |f(x_i)-f(y_i)|^2 \\ &\leq \sum_{i=1}^{m} L^2|x_i-y_i|^2 = L^2||\boldsymbol{x}-\boldsymbol{y}||^2, \end{aligned}$$

for which

$$||F(\boldsymbol{x}) - F(\boldsymbol{y})|| \leq L||\boldsymbol{x}-\boldsymbol{y}||.$$

Thus, $\mathrm{Lip}(F)$ is upper bounded by $L = \mathrm{Lip}(f)$;

$$\mathrm{Lip}(F) \leq \mathrm{Lip}(f). \tag{2.5.26}$$

2.6 Composites of Functions

Let $f : \mathbb{R}^n \to \mathbb{R}^m$ and $g : \mathbb{R}^m \to \mathbb{R}^k$ be Lipstchitz continuous functions. Then, the composite function $g \circ f$ is also Lipstchitz continuous and the optimal constant is bounded by the product of those of f and g ;

$$\mathrm{Lip}(g \circ f) \leq \mathrm{Lip}(f)\mathrm{Lip}(g) \tag{2.6.27}$$

This is readily proved by the following reason. Let L and M be the optimal Lipschitz constants for f and g respectively. Then, for all $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$,

$$\begin{aligned} ||(g \circ f)(\boldsymbol{x}) - (g \circ f)(\boldsymbol{y})|| &= ||g(f(\boldsymbol{x})) - g(f(\boldsymbol{y}))|| \\ &\leq M||f(\boldsymbol{x}) - f(\boldsymbol{y})|| \\ &\leq LM||\boldsymbol{x} - \boldsymbol{y}||. \end{aligned}$$

By the minimality of $\mathrm{Lip}(g \circ f)$, we have (2.6.27).

Note that (2.3.17) is a specific case of (2.6.27). Here, the inequality can be strict. As an example, $||W_2||||W_4|| = 3\times\sqrt{7+\sqrt{13}} > \sqrt{1+\sqrt{37}} = ||W_2W_4||$. Note also that (2.6.27) can be generalized further, to the case when the composition involoves several functions. That is, if $f_i : \mathbb{R}^{n_{i-1}} \to \mathbb{R}^{n_i}$ are functions between euclidean spaces, with its optimal Lipschitz constant L_i, for $i = 1, 2, \cdots, p$, the composite

$$f_p \circ \cdots \circ f_1 : \mathbb{R}^{n_1} \to \mathbb{R}^{n_{p+1}}$$

is Lipschitz constinuous and

$$\mathrm{Lip}(f_p \circ \cdots \circ f_1) \leq \prod_{i=1}^{k} L_i. \tag{2.6.28}$$

Chapter 3

Lipschitz Constants of Neural Networks

Now we turn to the problem of finding the optimal Lischitz constant of various architectures frequently appearing in machine learning and deep learning.

For locally Lipschitz function, it is possible to express the Lipschitz constant of f in its exact form, as the Rademacher Theorem suggests. But, it is not always easy or feasible to find the exact value of Lipschitz constant whenever the function f is given. In fact, although the function is of simple form such as 2-layered MLP, solving the precise value of Lipschitz constant of the function is known to be NP-hard. Instead of struggling to find analytic solutions, we present a systematic algorithm for each architectuare of neural network[1].

3.1 Rademacher Theorem

Here is a theorem that can be applied to all Lipschitz functions between euclidean spaces. The Lipschitz condition that we impose is not the global one ; it only need to be locally Lipschitz. It has two conclusions : the differentiability in almost everywhere sense and the formula for the optimal constant.

Theorem 3.1. *[8] Let $f : \mathbb{R}^n \to \mathbb{R}^m$ be a locally Lipschitz function. Then, f is differentiable almost everywhere, and*

$$\mathrm{Lip}(f) = \sup_{\boldsymbol{x} \in \mathbb{R}^n} ||D_{\boldsymbol{x}}(f)|| \tag{3.1.1}$$

The first statement — differentiability — is usually refered to the main conclusion of this theorem. But the proof of it is so lengthy that we omit the proof here. As for one-dimensional case, many of the textbooks of analysis covered this issue. In the process, concepts like absolute continuity, Radon-Nykodym theorem, total variation are involved [7].

The second statement, or the optimal Lipschitz constant is the one that we can use for our purpose. Note first that, $D_{\boldsymbol{x}} f$ is the Jacobian matrix A of f satisfying

$$\lim_{\boldsymbol{y} \to \boldsymbol{x}} \frac{||f(\boldsymbol{y}) - f(\boldsymbol{x}) - A(\boldsymbol{y} - \boldsymbol{x})||}{||\boldsymbol{y} - \boldsymbol{x}||} = 0,$$

and that $||D_{\boldsymbol{x}}(f)||$ is nothing but the operator norm we've defined in the previous chapter.

Note that although the equation (3.1.1) expresses $\mathrm{Lip}(f)$ explicitly, it doesn't mean that we can actually find the value of $\mathrm{Lip}(f)$.

3.2 Upperbounding the Lipschitz constant : AutoLip

Here is an algorithm called *AutoLip* that evaluate an upperbound for the Lipschitz constant. The algorithm can be applied to any deterministic neural network (resp. stochastic neural networks or bayesian neural networks) with the computation graph, which is a composite of elementary functions.

The only difficulty in the Rademacher theorem is the expression supremum ; it is not easy or sometimes impossible to evaluate the exact value of the supremum or the maximum of a function. But, it is possible, in most case, to find the maximum in the relatively simple function. In other words, the supremum for the Jacobian matrix can be calcuated for most elementary functions.

For a function $f : \mathbb{R}^n \to \mathbb{R}^m$, we've used the notation for the jacobian matrix of f as

$$D_{\boldsymbol{x}}(f)$$

in the previous section. In this section, we occasionally use

$$\frac{\partial \boldsymbol{y}}{\partial \boldsymbol{x}}$$

to represent the same matrix, provided that $\boldsymbol{y} = f(\boldsymbol{x})$. In this context, $||\frac{\partial \boldsymbol{y}}{\partial \boldsymbol{x}}||$ represents the operator norm for the matrix. Furthermore, if y is a function of a univariate variable x, we usually denote the ordinary derivative of y with respect to x as $\frac{dy}{dx}$. But, in order to unify the notation, we stick to use the partial derivative notation $\frac{\partial y}{\partial x}$.

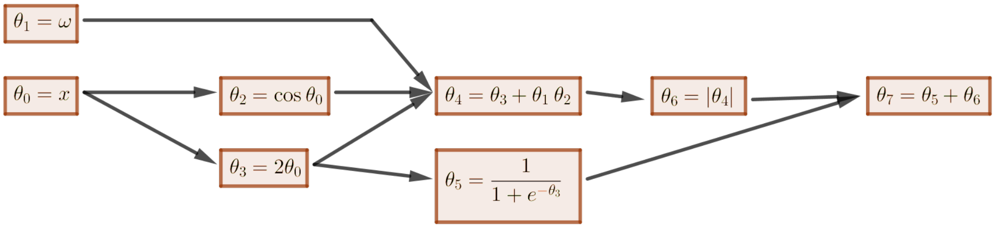
3.2.1 An exmaple for univariate case

Consider, for example, a univariate function f_ω with a parameter ω defined by

$$f_\omega(x) = \frac{1}{1 + e^{-x}} + |2x + \omega \cos x| \, .$$

In order to construct a systematic way to estimate the constant and to make use of automatic differentiation in implementing the backpropagation by Tensorflow or Pytorch, we think of a computation graph for f_ω. (Figure 3.1)

We can bound the Lipschitz constant of f_ω from above, by upperbounding the partial derivatives of θ_1, θ_2, $\cdots$, θ_7 consecutively where $\theta_7 = f_\omega(x)$. Since $\theta_1 = g_1(\theta_0) = \omega$, we have $\frac{\partial \theta_1}{\partial x} = 0$. The next parameter θ_2 is a function of one variable θ_0, but we regard θ_2 as a function of two variables ; $\theta_2 = g_2(\theta_0, \theta_1) =$



$\theta_1 = \omega$
$\theta_0 = x$
$\theta_2 = \cos\theta_0$
$\theta_3 = 2\theta_0$
$\theta_4 = \theta_3 + \theta_1\,\theta_2$
$\theta_5 = \frac{1}{1+e^{-\theta_3}}$
$\theta_6 = |\theta_4|$
$\theta_7 = \theta_5 + \theta_6$

Figure 3.1: A computation graph for the function $f_\omega(x) = \frac{1}{1+e^{-2x}} + |2x + \omega\cos x|$.

$\cos\theta_0$. It is legal to apply the chain rule to get

$$\begin{aligned}\frac{\partial\theta_2}{\partial x} &= \frac{\partial\theta_2}{\partial\theta_0}\frac{\partial\theta_0}{\partial x} + \frac{\partial\theta_2}{\partial\theta_1}\frac{\partial\theta_1}{\partial x}\\ &= (-\sin\theta_0)\times 1 + 0\times 0\\ &= -\sin x.\end{aligned}$$

Then, the Lipschitz constant $\mathrm{Lip}(\theta_2)$ of θ_2 is

$$\left|\left|\frac{\partial\theta_2}{\partial x}\right|\right| = \sup_x|-\sin x| = 1.$$

Again, since $\theta_3 = g_3(\theta_0,\theta_1,\theta_2) = 2\theta_0$, we have

$$\begin{aligned}\frac{\partial\theta_3}{\partial x} &= \frac{\partial\theta_3}{\partial\theta_0}\frac{\partial\theta_0}{\partial x} + \frac{\partial\theta_3}{\partial\theta_1}\frac{\partial\theta_1}{\partial x} + \frac{\partial\theta_3}{\partial\theta_2}\frac{\partial\theta_2}{\partial x}\\ &= 2\times 1 + 0\times 0 + 0\times(-\sin x) = 2,\end{aligned}$$

and thus $\left|\left|\frac{\partial\theta_2}{\partial x}\right|\right| = 2$. Note that the value of partial derivatives evaluated above, such as $\frac{\partial\theta_1}{\partial x}$ and $\frac{\partial\theta_2}{\partial x}$, can be applied to calcuate $\frac{\partial\theta_3}{\partial x}$. To represent that $\left|\left|\frac{\partial\theta_3}{\partial x}\right|\right|$ is upper bounded by 2, we write $L_3 = 2$. Likewise, let $L_1 = 0$ and $L_0 = 1$.

For the next parameter $\theta_4 = g_4(\theta_0, \theta_1, \theta_2, \theta_3) = \theta_3 + \theta_1\theta_2$, we have

$$\begin{aligned}\frac{\partial\theta_4}{\partial x} &= \frac{\partial\theta_4}{\partial\theta_0}\frac{\partial\theta_0}{\partial x} + \frac{\partial\theta_4}{\partial\theta_1}\frac{\partial\theta_1}{\partial x} + \frac{\partial\theta_4}{\partial\theta_2}\frac{\partial\theta_2}{\partial x} + \frac{\partial\theta_4}{\partial\theta_3}\frac{\partial\theta_3}{\partial x} \\ &= 0 \times 1 + \theta_2 \times 0 + \theta_1 \times (-\sin x) + 1 \times 2\end{aligned}$$

and

$$\begin{aligned}\left|\left|\frac{\partial\theta_4}{\partial x}\right|\right| &\leq |\omega| \cdot \left|\left|\frac{\partial\theta_2}{\partial x}\right|\right| + 1 \times \left|\left|\frac{\partial\theta_3}{\partial x}\right|\right| \\ &\leq |\omega| \cdot L_2 + L_3 = |\omega| + 2.\end{aligned}$$

Thus, $L_4 = |\omega| + 2$.

In similar fashions, $\theta_5 = g_5(\theta_0, \theta_1, \cdots, \theta_4) = \frac{1}{1+e^{-\theta_3}}$, $\theta_6 = g_6(\theta_0, \theta_1, \cdots, \theta_5) = |\theta_4|$ and $\theta_7 = g_7(\theta_0, \theta_1, \cdots, \theta_6) = \theta_5 + \theta_6$ have

$$\begin{aligned}\frac{\partial\theta_5}{\partial x} &= \frac{\partial\theta_5}{\partial\theta_3}\frac{\partial\theta_3}{\partial x} & \left|\left|\frac{\partial\theta_5}{\partial x}\right|\right| &\leq \frac{1}{4} \times 2 = \frac{1}{2} \\ \frac{\partial\theta_6}{\partial x} &= \frac{\partial\theta_6}{\partial\theta_4}\frac{\partial\theta_4}{\partial x} & \left|\left|\frac{\partial\theta_6}{\partial x}\right|\right| &\leq \left|\left|\frac{\partial\theta_6}{\partial\theta_4}\right|\right| \left|\left|\frac{\partial\theta_6}{\partial\theta_4}\right|\right| = |\omega| + 2 \\ \frac{\partial\theta_7}{\partial x} &= \frac{\partial\theta_7}{\partial\theta_5}\frac{\partial\theta_5}{\partial x} + \frac{\partial\theta_7}{\partial\theta_6}\frac{\partial\theta_6}{\partial x} & \left|\left|\frac{\partial\theta_7}{\partial x}\right|\right| &\leq \left|\left|\frac{\partial\theta_5}{\partial x}\right|\right| + \left|\left|\frac{\partial\theta_6}{\partial x}\right|\right| \leq |\omega| + \frac{5}{2}.\end{aligned}$$

Thus, the Lipschitz constant for the function f_ω is upper bounded by $\omega + \frac{5}{2}$.

3.2.2 AutoLip

Consider $f : \mathbb{R}^n \to \mathbb{R}^m$, a composite function of elementary functions g_0, g_1, g_2, $\cdots$, g_K, so that

$$f = g_K \circ \cdots \circ g_2 \circ g_1 \circ g_0.$$

For every $\boldsymbol{x} \in \mathbb{R}^n$, define the intermediate variables $\theta_0 = g_0(\boldsymbol{x})$, $\theta_1 = g_1(g_0(\boldsymbol{x}))$, $\cdots$, $\theta_K = g_K(g_{K-1}(\cdots g_1(g_0(\boldsymbol{x}))))= f(\boldsymbol{x})$. For a notational purpose, let g_0 be the identity function and think of g_k as a function of θ_0, θ_1, $\cdots$, θ_{k-1} for $1 \leq k \leq K$ and write

$$\begin{aligned} \theta_1 &= g_1(\theta_0) \\ \theta_2 &= g_2(\theta_0, \theta_1) \\ &\vdots \\ \theta_K &= g_K(\theta_0, \theta_1, \cdots, \theta_{K-1}). \end{aligned}$$

An upper bound L_k for the lipschitz constant $\sup_{\boldsymbol{x}} \left|\left|\frac{\partial \theta_k}{\partial \boldsymbol{x}}\right|\right|$ of θ_k, can be evaulated inductively, where L_K is an upper bound for $\mathrm{Lip}(f)$.

Let $L_0 = 1$. Suppose that suppose that k is an integer such that $1 \leq k \leq K$ and we have an upper bound L_i of the Lipschitz constant

$$\sup_{\boldsymbol{x}} \left|\left|\frac{\partial \theta_i}{\partial \boldsymbol{x}}\right|\right| \leq L_i$$

for $1 \leq i \leq k-1$. Since

$$\begin{aligned} \frac{\partial \theta_k}{\partial \boldsymbol{x}} &= \frac{\partial \theta_k}{\partial \theta_0}\frac{\partial \theta_0}{\partial \boldsymbol{x}} + \frac{\partial \theta_k}{\partial \theta_1}\frac{\partial \theta_1}{\partial \boldsymbol{x}} + \cdots + \frac{\partial \theta_k}{\partial \theta_{k-1}}\frac{\partial \theta_{k-1}}{\partial \boldsymbol{x}} \\ &= \sum_{i=0}^{k-1} \frac{\partial \theta_k}{\partial \theta_i}\frac{\partial \theta_i}{\partial \boldsymbol{x}}, \end{aligned}$$

it follows that

$$\left|\left|\frac{\partial \theta_k}{\partial \boldsymbol{x}}\right|\right| \leq \sum_{i=0}^{k-1} \left|\left|\frac{\partial \theta_k}{\partial \theta_i}\right|\right| \left|\left|\frac{\partial \theta_i}{\partial \boldsymbol{x}}\right|\right|$$

and that

$$\sup_{\boldsymbol{x}} \left|\left|\frac{\partial \theta_k}{\partial \boldsymbol{x}}\right|\right| \leq \sum_{i=0}^{k-1} \sup \left|\left|\frac{\partial \theta_k}{\partial \theta_i}\right|\right| \cdot L_i.$$

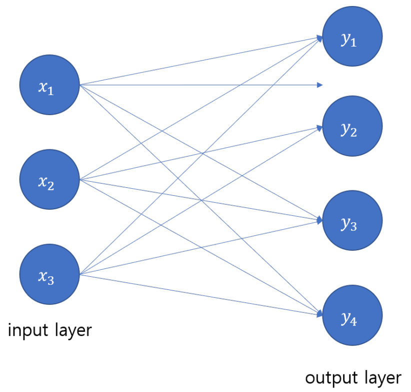


Figure 3.2: A single layer perceptron of input dimension 3 and output dimension 4 with weight matrix W and the bias b.

Let L_k be the right hand side of the above inequality and proceed for $k+1$. After K iterations, an upper bound L_K of the lipschitz constant for $\theta_K = f(\boldsymbol{x})$ is obtained. (Algorithm 1)

Algorithm 1: AutoLip, [1]

```
**Input:** the function $f = g_K \circ \cdots \circ g_0$,
where $\theta_k = (g_k \circ \cdots \circ g_0)(\boldsymbol{x})$ for $0 \leq k \leq K$
**Output:** An upper bound $L$ for Lip($f$)
$k \leftarrow 0$;
$L_0 \leftarrow 1$;
**while** $k < K$ **do**
    $L_k \leftarrow \sum_{i=0}^{k-1} \left( \sup \left|\left| \frac{\partial \theta_k}{\partial \theta_i} \right|\right| \right) L_i$;
    $k \leftarrow k+1$;
**end**
$L \leftarrow L_K$
```

3.3 Multi-Layer Perceptrons

Consider a single layer perceptron (Figure 3.2). Denote the single layer per-

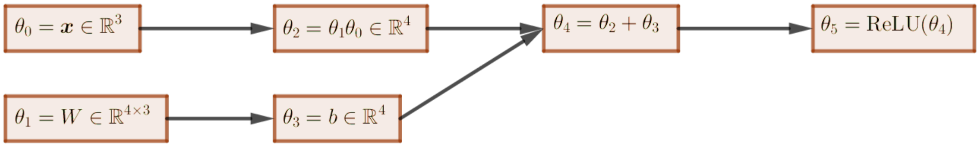

$\theta_0 = \boldsymbol{x} \in \mathbb{R}^3$
$\theta_1 = W \in \mathbb{R}^{4\times 3}$
$\theta_2 = \theta_1\theta_0 \in \mathbb{R}^4$
$\theta_3 = b \in \mathbb{R}^4$
$\theta_4 = \theta_2 + \theta_3$
$\theta_5 = \mathrm{ReLU}(\theta_4)$

Figure 3.3: A computation graph for the function the single layer perceptron

ceptron by $f_\omega : \mathbb{R}^3 \to \mathbb{R}^4$, which is defined as

$$f_\omega(\boldsymbol{x}) = \mathrm{ReLU}(W\boldsymbol{x} + b),$$

where ω stands for the set of parameters : $\omega = (W, b)$. With the computation graph illustrated in Figure 3.3, we have

$$\begin{aligned}
&k = 0;\;\; L_0 = 1 \\
&k = 1;\;\; L_1 = \sup\left|\left|\frac{\partial\theta_1}{\partial\theta_0}\right|\right| \times L_0 = 0 \times 1 = 0 \\
&k = 2;\;\; L_2 = \sup\left|\left|\frac{\partial\theta_2}{\partial\theta_0}\right|\right| \times L_0 + \sup\left|\left|\frac{\partial\theta_2}{\partial\theta_1}\right|\right| \times L_1 = ||W|| \\
&k = 3;\;\; L_3 = \sup\left|\left|\frac{\partial\theta_3}{\partial\theta_0}\right|\right| \times L_0 + \sup\left|\left|\frac{\partial\theta_3}{\partial\theta_1}\right|\right| \times L_1 + \sup\left|\left|\frac{\partial\theta_3}{\partial\theta_2}\right|\right| \times L_2 = 0 \\
&k = 4;\;\; L_4 = \sup\left|\left|\frac{\partial\theta_4}{\partial\theta_0}\right|\right| \times L_0 + \sup\left|\left|\frac{\partial\theta_4}{\partial\theta_1}\right|\right| \times L_1 \\
&\qquad\qquad + \sup\left|\left|\frac{\partial\theta_4}{\partial\theta_2}\right|\right| \times L_2 + \sup\left|\left|\frac{\partial\theta_4}{\partial\theta_3}\right|\right| \times L_3 \\
&\qquad\qquad = 0 \times 1 + 0 \times 0 + 1 \times ||W|| + 1 \times 0 = ||W|| \\
&k = 5;\;\; L_5 = \sup\left|\left|\frac{\partial\theta_5}{\partial\theta_0}\right|\right| \times L_0 + \sup\left|\left|\frac{\partial\theta_5}{\partial\theta_1}\right|\right| \times L_1 + \sup\left|\left|\frac{\partial\theta_5}{\partial\theta_2}\right|\right| \times L_2 \\
&\qquad\qquad + \sup\left|\left|\frac{\partial\theta_5}{\partial\theta_3}\right|\right| \times L_3 + \sup\left|\left|\frac{\partial\theta_5}{\partial\theta_4}\right|\right| \times L_4 \\
&\qquad\qquad = 0 \times 1 + 0 \times 0 + 0 \times ||W|| + 0 \times 0 + 1 \times ||W|| = ||W||.
\end{aligned}$$

The resulting upper bound for the Lipschitz constant is $||W||$. Note that we've used the fact that Lip(ReLU) = 1.

Let f be a multi layer perceptron of N layers, with 1-Lipschitz activation functions (like ReLU, Leaky ReLU, tanh, or SoftPlus). Then, we can think of f as a composite of f_1, f_2, $\cdots$, f_N where each f_i is a single layer perceptron described in the above paragraph, for $1 \leq n \leq N$. Then,

$$\mathrm{Lip}(f) \leq \prod_{i=1}^{n} \mathrm{Lip}(f_n) \leq \prod_{i=1}^{n} ||W_n||,$$

where each W_n is a weight matrix in the nth layer.

3.4 Convolutional Neural Networks

A vanilla convolutional neural network is a composition of convolutional layers, max pooling layers and linear layers. VGG-16, for example, consists of 13 convolutional layers, 5 maxpooling layers, 3 fully connected layers and a softmax function in the output layer [16]. While linear layers are what we've discussed in the previous section, we need to evaluate the Lipschitz constant of convolutional layers and max pooling layers.

3.4.1 Convolutional layers

Consider a convolutional layer of 5×5 input array and 3×3 filters. To make it simple, suppose that it has only one channel and that padding option is set to `valid` so that the output array is of 3×3. Denote the input array, the filter and the output array by X, F and Y. Instead of viewing X and Y as matrices(e.g. $X = [x_{ij}]_{5\times5}$, $Y = [y_{ij}]_{3\times3}$), let X and Y be column vectors of

$$\underset{F}{\begin{bmatrix}
f_{11} & f_{12} & f_{13} & & & f_{21} & f_{22} & f_{23} & & & f_{31} & f_{32} & f_{33} & & & & & & & & & & & & \\
& f_{11} & f_{12} & f_{13} & & & f_{21} & f_{22} & f_{23} & & & f_{31} & f_{32} & f_{33} & & & & & & & & & & & \\
& & f_{11} & f_{12} & f_{13} & & & f_{21} & f_{22} & f_{23} & & & f_{31} & f_{32} & f_{33} & & & & & & & & & & \\
& & & & & f_{11} & f_{12} & f_{13} & & & f_{21} & f_{22} & f_{23} & & & f_{31} & f_{32} & f_{33} & & & & & & & \\
& & & & & & f_{11} & f_{12} & f_{13} & & & f_{21} & f_{22} & f_{23} & & & f_{31} & f_{32} & f_{33} & & & & & & \\
& & & & & & & f_{11} & f_{12} & f_{13} & & & f_{21} & f_{22} & f_{23} & & & f_{31} & f_{32} & f_{33} & & & & & \\
& & & & & & & & & & f_{11} & f_{12} & f_{13} & & & f_{21} & f_{22} & f_{23} & & & f_{31} & f_{32} & f_{33} & & \\
& & & & & & & & & & & f_{11} & f_{12} & f_{13} & & & f_{21} & f_{22} & f_{23} & & & f_{31} & f_{32} & f_{33} & \\
& & & & & & & & & & & & f_{11} & f_{12} & f_{13} & & & f_{21} & f_{22} & f_{23} & & & f_{31} & f_{32} & f_{33}
\end{bmatrix}} \times \underset{X}{\begin{bmatrix} x_{11} \\ x_{12} \\ x_{13} \\ x_{14} \\ x_{15} \\ x_{21} \\ x_{22} \\ x_{23} \\ x_{24} \\ x_{25} \\ x_{31} \\ x_{32} \\ x_{33} \\ x_{34} \\ x_{35} \\ x_{41} \\ x_{42} \\ x_{43} \\ x_{44} \\ x_{45} \\ x_{51} \\ x_{52} \\ x_{53} \\ x_{54} \\ x_{55} \end{bmatrix}} = \underset{Y}{\begin{bmatrix} y_{11} \\ y_{12} \\ y_{13} \\ y_{21} \\ y_{22} \\ y_{23} \\ y_{31} \\ y_{32} \\ y_{33} \end{bmatrix}}$$

Figure 3.4: A convolutional layer as a matrix multiplication

dimension 25 and 9, respectively, so that

$$X = \begin{bmatrix} x_{11} \\ x_{12} \\ \vdots \\ x_{55} \end{bmatrix}, \quad Y = \begin{bmatrix} y_{11} \\ y_{12} \\ \vdots \\ y_{33} \end{bmatrix}.$$

Then, we can think of the convolutional layer as left multiplication of a matrix F (Figure 3.4).

The Lipschitz constant for the convolution layer is, thus, the operator norm of the matrix F. But, it is complicated and time consuming to evaluate the value directly. Instead of applying the Theorem 2.4 and evaluate $||F|| = \sqrt{||F^T F||}$ rigorously, we can use numerical method called *power method.*

3.4.2 Power method

Supposet that $A \in \mathbb{R}^{n\times n}$ is a diagonalizable matrix, or that A has distinct eigenvalues λ_1, λ_2, $\cdots$, λ_n. This is usually the case, since the probability that the characteristic equation happen to have repeated root is 0.

Then, we have not only the distinct eigenvalues λ_1, $\cdots$, λ_n, but also the linearly independent eigenvectors v_1, $\cdots$, v_n. Without loss of generality, let $|\lambda_1| \geq |\lambda_2| \geq \cdots \geq |\lambda_n|$, so that λ_1 is the *dominating eigenvalue* ; $|\lambda_1| = \max_i |\lambda_i|$.

Let $\boldsymbol{x}$ be any nonzero vector in $\mathbb{R}^n$. There exist c_1, $\cdots$, c_n such that

$$\boldsymbol{x} = c_1\boldsymbol{v}_1 + \cdots + c_n\boldsymbol{v}_n.$$

Then,

$$\begin{aligned} A\boldsymbol{x} &= A(c_1\boldsymbol{v}_1 + \cdots + c_n\boldsymbol{v}_n) \\ &= c_1A\boldsymbol{v}_1 + \cdots + c_nA\boldsymbol{v}_n \\ &= c_1\lambda_1\boldsymbol{v}_1 + \cdots + c_n\lambda_n\boldsymbol{v}_n. \end{aligned}$$

Moreover,

$$\begin{aligned} A^2\boldsymbol{x} &= A(c_1\lambda_1\boldsymbol{v}_1 + \cdots + c_n\lambda_n\boldsymbol{v}_n) \\ &= c_1\lambda_1A\boldsymbol{v}_1 + \cdots + c_n\lambda_nA\boldsymbol{v}_n \\ &= c_1{\lambda_1}^2\boldsymbol{v}_1 + \cdots + c_n{\lambda_n}^2\boldsymbol{v}_n. \end{aligned}$$

In the similar fashion, we have, for any positive integer k,

$$A^k\boldsymbol{x} = c_1{\lambda_1}^k\boldsymbol{v}_1 + \cdots + c_n{\lambda_n}^k\boldsymbol{v}_n.$$

Among n terms $c_1{\lambda_1}^k\boldsymbol{v}_1$, $\cdots$, $c_n{\lambda_n}^k\boldsymbol{v}_n$, the first term is the dominating term, since

$$A^k\boldsymbol{x} = {\lambda_1}^k\left(c_1\boldsymbol{v}_1 + c_2\left(\frac{\lambda_2}{\lambda_1}\right)^k\boldsymbol{v}_2 + \cdots + c_n\left(\frac{\lambda_n}{\lambda_1}\right)^k\boldsymbol{v}_n\right)$$

and $\left(\frac{\lambda_i}{\lambda_1}\right)^k \to 0$ exponentially as $k \to \infty$, for $i = 2, \cdots, n$. In short, for sufficient large integer k, we have

$$A^k\boldsymbol{x} \approx c_1{\lambda_1}^k\boldsymbol{v}_1.$$

Furthermore, we can approximately think of $A^k\boldsymbol{x}$ as the eigenvector, corresponding eigenvalue of which is the dominating one. Now, we can use the following lemma, stating that if we are given an eigenvector, we can evaluate the the corresponding eigenvalue.

Lemma 3.2. ***(Rayleigh Quotient [18])*** *If $\boldsymbol{x}$ is an eigenvector of a matrix A, then its corresponding eigenvalue is given by*

$$\lambda = \frac{\langle A\boldsymbol{x}, \boldsymbol{x}\rangle}{\langle \boldsymbol{x}, \boldsymbol{x}\rangle} = \frac{\boldsymbol{x}^T A^T \boldsymbol{x}}{\boldsymbol{x}^T\boldsymbol{x}}.$$

Here, $\langle\cdot,\cdot\rangle$ is the inner product as defined in Subsection 2.3.3. The proof is straightforward if we replace $A\boldsymbol{x}$ by $\lambda\boldsymbol{x}$. For rectangular matrix F as in Subsection 3.4.1, approximating algorithm for $||F||$ is illustrated in Algorithm 2.

3.4.3 Max pooling layers

The Lipschitz constant of the max pooling layer is 1, by the following obvious reason.

Consider a usual 2×2 maxpooling, applied to 2d-array X. Suppose that X consists $4N$ numbers x_1, x_2, x_3, x_4, $\cdots$, x_{4N-3}, x_{4N-2}, x_{4N-1}, x_{4N} and

Algorithm 2: Approximating $||F||$ using the power method

```
Input: A rectangular matrix $F \in \mathbb{R}^{n\times m}$
Output: Approximation for $||F||$
$A \leftarrow F^T F$;
Set an integer $k$ sufficiently large enough;
Set a nonzero vector $x \in \mathbb{R}^m$;
$x_k \leftarrow A^k x$;
$\tilde{\lambda} \leftarrow \frac{{x_k}^T A^T x_k}{{x_k}^T x_k}$;
$||F|| \leftarrow \sqrt{\tilde{\lambda}}$
```

denote X as a $4N$-dimensional vector by

$$X = [x_1 \;\; x_2 \;\; x_3 \;\; x_4 \;\; \cdots \;\; x_{4N-3} \;\; x_{4N-2} \;\; x_{4N-1} \;\; x_{4N}]^T.$$

Let $\boldsymbol{x}^{(1)}$, $\cdots$, $\boldsymbol{x}^{(N)}$ be four-dimensional-vectors defined by

$$\boldsymbol{x}^{(1)} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}, \quad \cdots, \quad \boldsymbol{x}^{(N)} = \begin{bmatrix} x_{4N-3} \\ x_{4N-2} \\ x_{4N-1} \\ x_{4N} \end{bmatrix}$$

and write

$$X = \begin{bmatrix} \boldsymbol{x}^{(1)} \\ \vdots \\ \boldsymbol{x}^{(N)} \end{bmatrix}.$$

Lemma 3.3. *A function* $[a_1,\ a_2\, a_3,\ a_4]^T \mapsto \max a_i$ *is 1-Lipschitz function.*

Proof. Denote $a = [a_1,\ a_2\, a_3,\ a_4]^T$ and $b = [b_1,\ b_2\, b_3,\ b_4]^T$. Let $a_{i_0} = \max a_i$.

Then,

$$\max a_i - \max b_i = a_{i_0} - \max b_i \leq a_{i_0} - b_{i_0} \leq \max(a_i - b_i).$$

Similarly,

$$\max b_i - \max a_i \leq \max(a_i - b_i).$$

Thus,

$$|\max a_i - \max b_i| \leq \max(a_i - b_i),$$

and Lip(max) = 1. □

Now, we are ready to evaluate Lip(MaxPool). The function

$$\text{MaxPool} : \mathbb{R}^{4N} \to \mathbb{R}^N$$

can be expressed as

$$\text{MaxPool}(X) = \begin{bmatrix} \max\left(\boldsymbol{x}^{(1)}\right) \\ \vdots \\ \max\left(\boldsymbol{x}^{(N)}\right) \end{bmatrix}.$$

Thus,

$$\begin{aligned} {||f(X) - f(Y)||_2}^2 &= \sum_{n=1}^{N} \left(\max\left(\boldsymbol{x}^{(n)}\right) - \max\left(\boldsymbol{y}^{(n)}\right)\right)^2 \\ &\leq \sum_{n=1}^{N} \left(\max\left(\boldsymbol{x}^{(n)} - \boldsymbol{y}^{(n)}\right)\right)^2 \\ &\leq {||\text{MaxPool}(X - Y)||_2}^2. \end{aligned}$$

Therefore,

$$\text{Lip}(\text{MaxPool}) = 1.$$

Chapter 4

Conclusion

For a function between metric spaces, the Lipschitz constant measures the rate of change of the function. If we conceive a neural network as a function between Euclidean spaces, the Lipschitz constant of the neural network tells us how sensitive the output is, with respect to the input. One can relate the Lipschitz constant to robustness. But the Lipschitz constant is not the same as the robustness, in that it measures the sensitivity when the parameters are fixed.

The Lipscthiz constant for relatively simple functions such as activation functions, linear functions and affine functions can be obtained either by simple calculation or the notion of matrix norm. But the actual computation for the neural networks are not that easy. Although, we can express the lipschitz constant of *any* locally Lipschitz functions by Rademacher Theorem, it is impossible (NP-hard) to evaluate the optimal constant even for a simple architecture such as 2-layer perceptron. Rather, we can think of several numerical way to find an upper bound for the constant by an algorithm called *AutoLip* or to use an approximation algorithm called the power method.

Bibliography

[1] Virmaux, A., & Scaman, K. (2018). Lipschitz regularity of deep neural networks: analysis and efficient estimation. *Advances in Neural Information Processing Systems*, 31.

[2] Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., & Fergus, R. (2014). Intriguing properties of neural networks. *Proceedings of the International Conference on Learning Representations (ICLR)*

[3] Arjovsky, M., Chintala, S., & Bottou, L. (2017, July). Wasserstein generative adversarial networks. In International conference on machine learning (pp. 214-223). PMLR.

[4] Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., & Courville, A. C. (2017). Improved training of wasserstein gans. Advances in neural information processing systems, 30.

[5] Computing The Norm of a matrix, `https://kconrad.math.uconn.edu/blurbs/linmultialg/matrixnorm.pdf`, University of Connecticut.

[6] Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep learning. MIT press.

[7] Rudin, Walter. *Real and Complex Analysis.* : McGraw-Hill Science/Engineering/Math, 1986.

[8] Herbert Federer. *Geometric measure theory.* Classics in Mathematics. Springer-Verlag Berlin Heidelberg, 1969.

[9] Villani, C. (2009). *Optimal transport: old and new* (Vol. 338, p. 23). Berlin: springer.

[10] Piotr Hajłasz, *Geometric Analysis*, `https://sites.pitt.edu/~hajlasz/Teaching/Math2304Spring2014/Part%201.pdf`, University of Pittsburgh.

[11] James T. Murphy, *An elementary proof of Rademacher's theorem*, `https://intfxdx.com/downloads/rademacher-thm-2015.pdf` THEOREM

[12] Jordan, M., & Dimakis, A. G. (2020). Exactly computing the local lipschitz constant of relu networks. *Advances in Neural Information Processing Systems*, 33, 7344-7353.

[13] Bhowmick, A., D'Souza, M., & Raghavan, G. S. (2021, September). Lipbab: computing exact lipschitz constant of relu networks. *In International Conference on Artificial Neural Networks* (pp. 151-162). Springer, Cham.

[14] Beer, G., & Hoffman, M. J. (2013). The Lipschitz metric for real-valued continuous functions. *Journal of Mathematical Analysis and Applications*, 406(1), 229-236.

[15] Carlini, N., & Wagner, D. (2017, May). Towards evaluating the robustness of neural networks. In 2017 *ieee symposium on security and privacy (sp)* (pp. 39-57). Ieee.

[16] Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.

[17] Singla, S., & Feizi, S. (2019). Robustness certificates against adversarial examples for relu networks. arXiv preprint arXiv:1902.01235.

[18] Larson, R. (2016). *Elementary linear algebra*. Cengage Learning.