CS 4540/CS 8803: Algorithmic Theory of Intelligence Fall 2023

Lecture 12: Neural Networks

Instructor: Santosh Vempala *Lecture date: 11/6, 11/8*

Scribed by: Mirabel Reid, Xinyuan Cao

1 Architecture

A neural network is a function $f : \mathbb{R}^n \to \mathbb{R}^k$. Let $\sigma : \mathbb{R} \to \mathbb{R}$ be some activation function. The output of a neural network can be defined as follows:

$$y = \sigma(W^{(l)}\sigma \dots \sigma(W^{(2)}\sigma(W^{(1)}x)) \dots)$$

where $W^{(i)} \in \mathbb{R}^{n_i \times n_{i+1}}$.

The choice of activation function σ is important for the behavior of the neural network. If σ is linear (for example, $\sigma(x) = x$), then f is also a linear function.

$$y = W^{(l)}W^{(l-1)} \dots W^{(2)}W^{(1)}x = Wx$$

So, the choice of σ should be nonlinear.

2 Training Algorithms

2.1 Defining the Loss Function

Suppose we are trying to predict from a sequence of labelled data points $\{(x_i, y_i)\}_{i=1}^M$ drawn from a distribution. For each (x, y), the neural network outputs a prediction $\tilde{y}$.

A natural idea for a loss function is zero-one loss, which counts the number of data points that the neural network predicts incorrectly. Note that when the data and the neural network architecture are given, the loss is defined as a function of the parameters of the network, the weights $W^{(1)}, \dots, W^{(l)}$.

$$L_{0,1}(W; x, y) = \sum_{i=1}^{M} 1\{y_i \neq \tilde{y}_i\}$$

However, this loss function is restrictive if σ is a continuous function. It is also possible to define the loss as an error function - the distance between the prediction and the true label.

$$L(W; x, y) = \frac{1}{M}\sum_{i=1}^{M} \|y_i - \tilde{y}_i\|^2$$

2.2 Gradient Descent

Let $L(w; x, y)$ be the loss function. Consider the goal of finding the value of w^* which minimizes the loss. At the minimum, $\nabla L(w^*; x, y) = 0$. (Keep in mind that $\nabla f(x) = 0$ does not imply that x is the global minimum in general- it could be a local minimum, maximum, or saddle point).

To find a point where $\nabla f = 0$, we can use the following update function, parameterized by η

$$x^{(k+1)} = x^{(k)} - \eta \nabla f(x^{(k)}) \tag{1}$$

Theorem 1. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be differentiable, and ∇f be L-Lipschitz. Then gradient descent (Equation 1) with $\eta = \frac{1}{L}$ converges to an $\hat{x}$ with $\|\nabla f(\hat{x})\|_2 \leq \epsilon$ in at most $\frac{2L(f(x^{(0)})-f(x^*))}{\epsilon^2}$ steps.*

Proof. To prove this, we will use Taylor's theorem for 1-dimensional functions.

Lemma 2. *Let $g : \mathbb{R} \to \mathbb{R}$ be continuous and twice-differentiable on the interval $[a, b]$. Then, there exists a $z \in [a, b]$ such that $g(b) = g(a) + (b - a)g'(a) + \frac{1}{2}g''(z)(b - a)^2$.*

Now, let $g(t) = f((1 - t)x + ty)$ for a fixed $x, y \in \mathbb{R}^n$. Then, $g'(t) = \langle y - x, \nabla f((1 - t)x + ty)\rangle$. By Taylor's theorem, there exists a $z \in [0, 1]$ such that:

$$g(1) = g(0) + g'(0) + g''(z)$$

Substituting the definition of g, and letting $u = (1 - z)x + zy$:

$$f(y) = f(x) + \langle y - x, \nabla f(x)\rangle + \frac{1}{2}(y - x)^\top \nabla^2 f(u)(y - x)$$

Now, we can analyze the gradient update step, setting $y = x - \eta\nabla f(x)$.

$$\begin{aligned} f(x - \eta\nabla f(x)) &= f(x) - \langle \eta\nabla f(x), \nabla f(x)\rangle + \frac{1}{2}\eta^2 \nabla f(x)^\top \nabla^2 f(u)\nabla f(x) \\ &\le f(x) - \eta\|\nabla f(x)\|^2 + \frac{1}{2}\eta^2 L\|\nabla f(x)\|^2) && \text{using the Lipschitz property} \\ &= f(x) - (\eta - \frac{1}{2}\eta^2 L)\|\nabla f(x)\|^2 \\ &\le f(x) - \frac{1}{2L}\|\nabla f(x)\|^2 && \text{substituting } \eta = \frac{1}{L} \\ \|\nabla f(x)\|^2 &\le 2L(f(x) - f(x - \eta\nabla f(x))) \end{aligned}$$

This implies that after $T = \frac{2L(f(x^{(0)}) - f(x^*))}{\epsilon^2}$ steps, $\|\nabla f(x^{(T)})\|^2 \le \epsilon$ □

When f is convex as well, we can show that this approaches a global minimum $f(x^*)$. If a function $f : \Omega \to \mathbb{R}$ is convex, it has the following properties:

1. $\forall x, y \in \Omega, \forall \alpha \in [0, 1]$,
$$f(\alpha x + (1 - \alpha)y) \le \alpha f(x) + (1 - \alpha)f(y)$$

2. If f is differentiable, $\forall x, y \in \Omega$,
$$f(y) \ge f(x) + \langle \nabla f(x), y - x\rangle$$

3. If f is twice differentiable, $\forall x \in \Omega$,
$$\nabla^2 f(x) \ge 0$$

Theorem 3. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be differentiable and convex, and ∇f be L-Lipschitz. Then, after k steps, gradient descent (Equation 1) with $\eta = \frac{1}{L}$ converges to*

$$f(x^{(k)}) - f(x^*) \le \frac{2LR^2}{k + 4}$$

where $R^2 = \max_{x : f(x) \le f(x^*)} \|x - x^*\|^2$.

Proof. We have shown that

$$f(x^{(k+1)}) \le f(x^{(k)}) - \frac{1}{2L}\|\nabla f(x^{(k)})\|^2$$

That is

$$f(x^{(k+1)}) - f(x^*) \le f(x^{(k)}) - f(x^*) - \frac{1}{2L}\|\nabla f(x^{(k)})\|^2$$

Denote $\epsilon_k := f(x^{(k)}) - f(x^*)$. That is,

$$\epsilon_{k+1} \le \epsilon_k - \frac{1}{2L}\|\nabla f(x^{(k)})\|^2$$

By convexity of f,

$$f(x^*) \ge f(x_k) + \langle \nabla f(x^{(k)}), x^* - x^{(k)} \rangle$$

This implies

$$\epsilon_k \le \langle \nabla f(x^{(k)}), x^{(k)} - x^* \rangle \le \|\nabla f(x^{(k)})\| \cdot \|x^{(k)} - x^*\|$$

Square both sides,

$$\|\nabla f(x^{(k)})\|^2 \ge \frac{\epsilon_k^2}{\|x^{(k)} - x^*\|^2}$$

Substituting and we have

$$\epsilon_{k+1} \le \epsilon_k - \frac{1}{2L}\|\nabla f(x^{(k)})\|^2 \le \epsilon_k - \frac{1}{2L}\frac{\epsilon_k^2}{\|x^{(k)} - x^*\|^2} \le \epsilon_k - \frac{\epsilon_k^2}{2LR^2}$$

Then we have

$$\frac{1}{\epsilon_{k+1}} - \frac{1}{\epsilon_k} = \frac{\epsilon_k - \epsilon_{k+1}}{\epsilon_k \cdot \epsilon_{k+1}} \ge \frac{\epsilon_k - \epsilon_{k+1}}{\epsilon_k^2} \ge \frac{\frac{\epsilon_k^2}{2LR^2}}{\epsilon_k^2} = \frac{1}{2LR^2}$$

Summing up over k, we have

$$\frac{1}{\epsilon_k} \ge \frac{1}{\epsilon_0} + \frac{k}{2LR^2}$$

On the other hand, we can bound ϵ_0 by Taylor expansion. There exists $z \in (x^{(0)}, x^*)$ such that

$$\begin{aligned}
\epsilon_0 = f(x^{(0)}) - f(x^*) =& \langle \nabla f(x^*), x^{(0)} - x^* \rangle + \frac{1}{2}(x^{(0)} - x^*)^\top \nabla^2 f(z)(x^{(0)} - x^*) \\
=& \frac{1}{2}(x^{(0)} - x^*)^\top \nabla^2 f(z)(x^{(0)} - x^*) && \text{By optimality of } x^* \\
\le& \frac{1}{2}L\|x^{(0)} - x^*\|^2 && \text{By Lipschitz} \\
\le& \frac{LR^2}{2}
\end{aligned}$$

This implies that

$$\frac{1}{\epsilon_k} \ge \frac{1}{\epsilon_0} + \frac{k}{2LR^2} \ge \frac{2}{LR^2} + \frac{k}{2LR^2} = \frac{k+4}{2LR^2}$$

That is,

$$\epsilon_k \le \frac{2LR^2}{k+4}$$

□

2.3 Backpropagation

Given a feed forward network with parameter w, we use gradient descent to optimize over w to achieve small training error. Given data $(x_i, y(x_i)), 1 \le i \le m$ and its estimator $\tilde{y}_i := \tilde{y}(x_i, w)$ with parameter w, we would like to minimize the empirical risk.

$$\text{err}(w) = \frac{1}{m}\sum_{i=1}^{m} \|y(x_i) - \tilde{y}(x_i, w)\|^2$$

To minimizing $\text{err}(w)$, we do the gradient descent with step size η. The update rule is

$$w \leftarrow w - \eta \nabla \text{err}(w)$$

By chain rule,

$$\frac{\partial \mathrm{err}(w)}{\partial w_{ij}} = \frac{\partial \mathrm{err}(w)}{\partial \tilde{y}} \cdot \frac{\partial \tilde{y}}{\partial w_{ij}} = -2(y-\tilde{y}) \cdot \frac{\partial \tilde{y}}{\partial w_{ij}} \tag{2}$$

Now if we describe explicitly the operation from the second to last layer to the last layer. $\tilde{y} = \sigma(\sum_j \alpha_j z_j)$, where σ is the activation function, z_j is the logit of the second to last layers and α_j is the weight on the last layer. Then we can apply chain rule again, and get

$$\frac{\partial \tilde{y}}{\partial \alpha_j} = \sigma'(\sum_k \alpha_k z_k) z_j \tag{3}$$

By combining Equation (2) and Equation (3), we get the partial derivative of $\mathrm{err}(w)$ with respect to the last layer parameter w_j.

$$\frac{\partial \mathrm{err}(w)}{\partial w_j} = -2(y-\tilde{y})\sigma'(\sum_k \alpha_k z_k) z_j$$

In the same way, we can pass the derivative message from the top layer of the neural network to the bottom layer and get the partial derivative of $\mathrm{err}(w)$ with respect to the weights in all layers efficiently. This process is called backpropagation.

2.4 Activation Function

The activation function $\sigma : \mathbb{R} \to \mathbb{R}$ can add the nonlinearty of the function to be represented. We list two common activation functions with their derivative here.

1. Sigmoid.

$$\sigma(x) = \frac{e^x}{1+e^x}, \quad \sigma'(x) = \sigma(x)(1-\sigma(x))$$

2. ReLU.

$$\sigma(x) = \max(x,0), \quad \sigma'(x) = \begin{cases} 1 & \text{for } x \geq 0 \\ 0 & \text{for } x < 0 \end{cases}$$

3 Expressiveness

We would like to study the expressiveness of the neural networks. The Universal Approximation Theorem states that a feed-forward network with a single hidden layer containing a finite number of neurons can approximate continuous functions on compact subsets of $\mathbb{R}^n$, under mild assumptions on the activation function. This means that such a network can approximate any function to a desired degree of accuracy given enough neurons in the hidden layer. Here we would like to study the expressiveness of some simpler function class. Specifically, we would like to approximate a function with the basis from some particular function class.

3.1 Polynomials

For a function $f : \mathbb{R} \to \mathbb{R}, \forall x, a \in \mathbb{R}$, by Taylor expansion, there exists $z \in [a, x]$ such that

$$f(x) = f(a) + f'(a)(x-a) + \frac{1}{2}f''(a)(x-a)^2 + \cdots + \frac{1}{k!}f^{(k)}(a)(x-a)^k + \frac{1}{(k+1)!}f^{(k+1)}(z)(x-a)^{k+1}$$

In other words, we can approximate $f(x)$ with a linear combination of polynomials up to order k (which we denote as $f_k(x)$) with approximation error

$$|f(x) - f_k(x)| \leq \frac{1}{(k+1)!}f^{(k+1)}(z)(x-a)^{k+1}$$

3.2 Fourier Series

For any function $f : [-\pi, \pi] \to \mathbb{R}$. We can decompose $f(x)$ with respect to $\cos(jx)$ and $\sin(jx), j \in \mathbb{Z}$ as follows.

$$f(x) = a_0 + \sum_{j=1}^{\infty} a_j \cos(jx) + \sum_{j=1}^{\infty} b_j \sin(jx)$$

where $a_j = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) \cos(jx)\, dx, b_j = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) \sin(jx)\, dx$. The coefficient a_j can be seen as the inner product of $f(x)$ and $\cos(jx)$ and b_j can be seen as the inner product of $f(x)$ and $\sin(jx)$. $\{\cos(jx), \sin(jx)\}$ constructs an orthogonal basis.

3.3 Sigmoidal Function

Theorem 4. *A two-layer neural network with sigmoidal (or similar) activation functions gives a universal approximation.*

We call a function $\sigma : \mathbb{R} \to \mathbb{R}$ to be sigmoidal if

$$\sigma(x) = \begin{cases} 0 & \text{for } x \to -\infty \\ \text{monotonically non-decreasing} & \text{for } -\infty < x < \infty \\ 1 & \text{for } x \to \infty \end{cases}$$

The theorem can be shown by representing $\cos(jx)$ using sigmoidal functions. Specifically, we partition the domain into small intervals $[x_1, x_2], [x_2, x_3], \cdots, [x_{t-1}, x_t]$. Then we can approximate $\cos(jx)\mathbb{1}_{x \in [x_i, x_{i+1}]}$ using sigmoidal functions. So we can approximate

$$\cos(jx) = \sum_{i=1}^{t-1} \cos(jx)\mathbb{1}_{x \in [x_i, x_{i+1}]}$$

using sigmoidal functions (we let $t \to \infty$).