Lecture 10

Backprop

Sol 1,2,3

Q1 : 4:00 Thu 2/12
15 min

L IV - 2

Lecture Notes IV – Neural Networks, Part 2

Marina Meilă
mmp@uwaterloo.ca

With Thanks to Pascal Poupart & Gautam Kamath
Cheriton School of Computer Science
University of Waterloo

February 8, 2026

Backpropagation

Train n.n. by G.D.

Training a single unit

Training a 2-layer network

Training a L-layer network

Reading HTF Ch.: 11.3 Neural networks, Murphy Ch.: (16.5 neural nets), Bach Ch.: –, Deep Learning Book (Goodfellow, Bengio, Courville) 6.1-4, ResNet 7.6, ConvNet 9., Autoencoders 14.1, Dive Into Deep Learning 4.1-4.3.

Training a neural network

BIG PICTURE

- Model is multilayer network.
- Layers $l = 1, 2, \ldots L$, with $x^{(l)} \in \mathbb{R}^{m_l}$, for $l = 1 : L-1$, $m_0 = d$, $m_L = 1$ (for regression and binary classification).

$$
\begin{aligned}
x^{(0)} &= x && (1)\\
x^{(L)} &= f(x) && (2)\\
x^{(l)} &= \phi(z^{(l)}) \quad \text{for } l = 1 : L-1 && (3)\\
x^{(L)} &= z^{(L)} \equiv \phi_{\text{out}}(z^{(L)1}) && (4)\\
z^{(l)} &= W^{(l)} x^{(l-1)} && (5)\\
& && (6)
\end{aligned}
$$

- Parameters $\boxed{\mathbb{W} = \{W^{(l)} \in \mathbb{R}^{m_l \times m_{l=1}}\}}$
- $W_k^{(l)}$ is row k of $W^{(l)}$ and corresponds to unit k of layer l
- $\boxed{\mathcal{D} = \{(x^1, y^1), \ldots (x^n, y^n)\}}$

$W_i^{(l)} \sim N(0, \frac{\sigma^2 I}{m_l})$

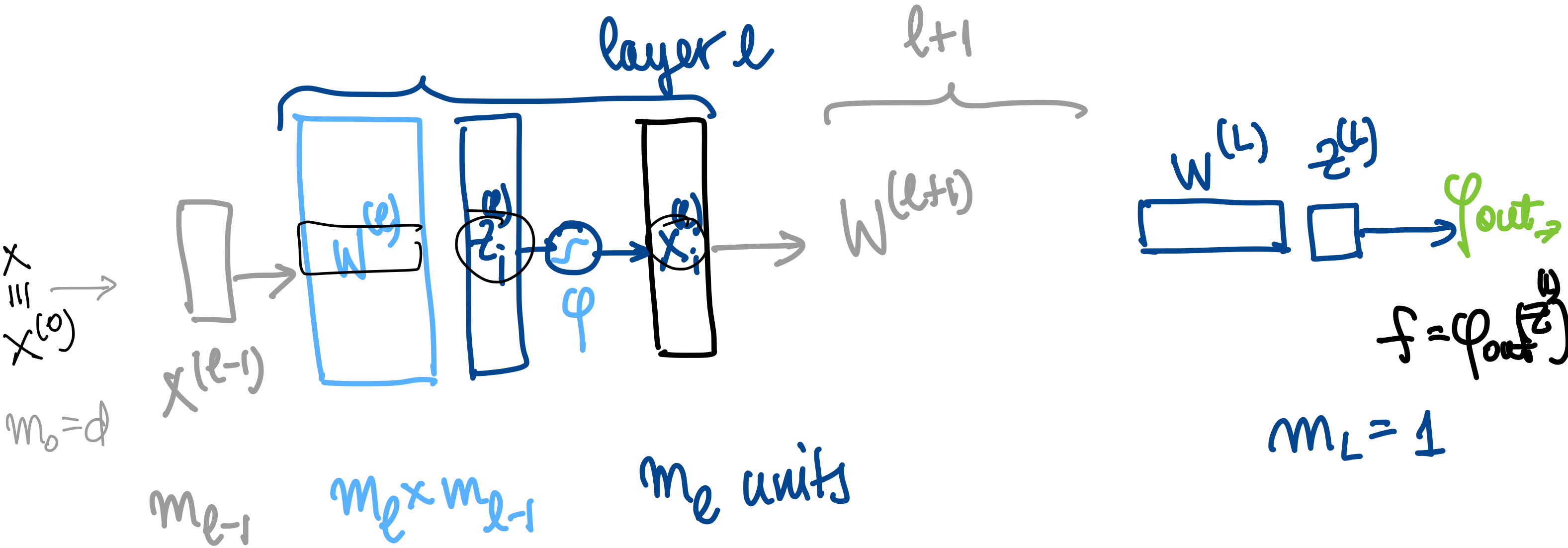
↑

$\mathbb{R}^{m_l}$

- **How to train this network?**
- Minimize $\mathcal{L}(\mathbb{W})$ (remember ϕ_{out} is associated with $\mathcal{L}$)

- Minimization by Gradient Descent
  - Initialize $\mathbb{W}$ to small random values 1.
  - for $t = 1, 2, \ldots$ do $\mathbb{W}^{t} \leftarrow \mathbb{W}^{t} - \eta \frac{\partial \mathcal{L}}{\partial \mathbb{W}}$ 2. (W^t)

- **We need to compute gradient** $\frac{\partial \mathcal{L}(\mathbb{W})}{\partial W_{ij}^{(l)}}$ for all $i = 1 : m_l$, $j = 1 : m_{l-1}$, $l = 1 : L$.



$l = 1 : L$

$$z^{(l)} = W^{(l)} x^{(l-1)}$$

$$\varphi(z_i^{(l)}) = x_i^{(l)}$$

$\left[\text{parameters } \mathbb{W} = \{ W^{(1)}, W^{(2)}, \dots W^{(L)} \} \right]$

Training a single unit

single (x, y)

$$\underset{\mathbb{R}^d}{x} \to \overset{W}{\boxed{}} \to \underset{\mathbb{R}^m}{z} \to \varphi_{out} \to f$$

- The function $f(x; \mathbb{W}) = \phi_{\text{out}} \underbrace{\left(\sum_{j=1}^{d} w_j x_j \right)}_{z}$ with parameters $\mathbb{W} = w \in \mathbb{R}^d$.
- The loss function $\mathcal{L}(y, f(x; \mathbb{W})) =$ Least Squares, or Logistic (Max Likelihood) binary or multiclass.
  Fact For each $\mathcal{L}$, there exists a mapping $y \to y_*$, $z \to \phi_{\text{out}}(z)$ such that
  $\boxed{-\frac{\partial \mathcal{L}(y, f(x; \mathbb{W}))}{\partial f} = y_* - \phi_{\text{out}}}$ = target output − network output
- Let's use chain rule

$z = Wx$

$y \to y_*$

$$L_{LS} = \frac{1}{2}(y_* - \varphi_{out})^2$$

$$\frac{\partial L}{\partial \varphi_{out}} = \varphi_{out} - y_*$$

$$-\frac{\partial \mathcal{L}}{\partial f} = y_* - \phi_{\text{out}}(z) \quad (7)$$

$$\frac{\partial f}{\partial z} = \phi'_{\text{out}}(z) \quad (8)$$

$$\frac{\partial f}{\partial w} = \phi'_{\text{out}}(z) x \quad (9)$$

$$\frac{\partial f}{\partial x} = \phi'_{\text{out}}(z) w \quad (10)$$

$$-\frac{\partial \mathcal{L}}{\partial w} = (y_* - \phi_{\text{out}}(z)) \phi'_{\text{out}}(z) x$$

$\frac{\partial z}{\partial W}$

$\frac{\partial z}{\partial x}$

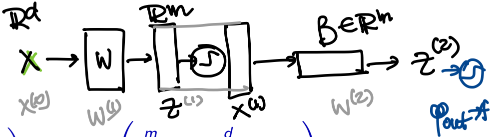
- Remember $\phi'_{\text{out}}(z) = z$ for $\mathcal{L}_{LS}$, $\phi'_{\text{out}}(z) = \phi(x)(1 - \phi(z))$ for $\mathcal{L}_{\text{logit}}$.
- Notice that $\partial \mathcal{L} / \partial w$ is a vector collinear with x.

Training a single unit, n data points

- $\mathcal{D} = \{(x^1, y^1), \dots (x^n, y^n)\}$
- For single pair (x, y), $\frac{\partial L}{\partial w} = (y - \phi_{\text{out}}(z))\phi'_{\text{out}}(z)x$
- For entire $\mathcal{D}$, $\mathcal{L}(\mathbb{W}) = \frac{1}{n}\sum_{i=1}^{n} \mathcal{L}(y^i, f(x^i; \mathbb{W}))$.
- The gradient of this loss is

$$-\frac{\partial L}{\partial w} = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{\partial}{\partial w}\mathcal{L}(y^i, f(x^i; \mathbb{W}))\right) = \frac{1}{n}\sum_{i=1}^{n}(y^i - \phi_{\text{out}}(z^i))\phi'_{\text{out}}(z^i)x^i. \tag{11}$$

Training a 2-layer network



- Consider a two layer neural network

$$f(x) = \phi_{\text{out}}\left(\sum_{i=1}^{m} \beta_i x_i^{(1)}\right) = \phi_{\text{out}}\left(\sum_{i=1}^{m} \beta_i \phi\left(\sum_{j=1}^{d} w_{ij} x_j\right)\right) \quad (12)$$

The parameters $\mathbb{W}$ are $\boxed{\beta}$ and $\boxed{W} = [w_{ij}]_{i=1:m, j=1:d}$

Gradient w.r.t. β (same as before! $w \leftarrow \beta$)

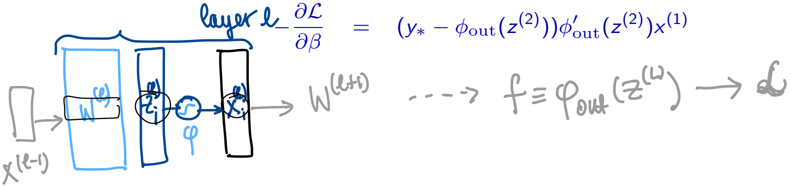
$$-\frac{\partial \mathcal{L}}{\partial f} = y_* - \phi_{\text{out}}(z^{(2)}) \quad (13)$$

$$\frac{\partial f}{\partial x^{(1)}} = \phi'_{\text{out}}(z^{(2)})\beta \quad (14)$$

$$\frac{\partial f}{\partial x_i^{(1)}} = \phi'_{\text{out}}(z^{(2)})\beta_i \quad (15)$$

$$\frac{\partial f}{\partial \beta} = \phi'_{\text{out}}(z^{(2)}) x^{(1)} \quad (16)$$

$$-\frac{\partial \mathcal{L}}{\partial \beta} = (y_* - \phi_{\text{out}}(z^{(2)}))\phi'_{\text{out}}(z^{(2)}) x^{(1)}$$



Trainining a 2-layer network – hidden layer parameters

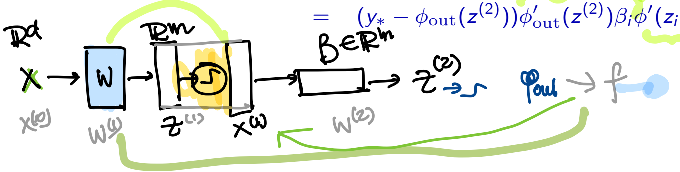
Gradient w.r.t. W

- Let's break $W \in \mathbb{R}^{m\times d}$ into rows. $W_i = [w_{ij}]_{j=1:d}$ is the column vector of weights for unit i in hidden layer.
- We have $z_i = W_i^T x$ and $x_i^{(1)} = \phi(z_i)$.
- Again, as before, with $\phi_{\text{out}} \leftarrow \phi$, $w \leftarrow W_i$, $z \leftarrow z_i$, $z^{(2)} = \beta^T x_i^{(1)}$, $f = \phi_{\text{out}}(z^{(2)})$we have

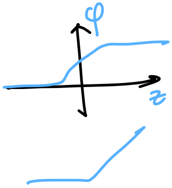
$$
\begin{aligned}
\frac{\partial x_i^{(1)}}{\partial z_i} &= \phi'(z_i) && (17)\\
\frac{\partial x_i^{(1)}}{\partial W_i} &= \phi'(z_i)x && (18)\\
\frac{\partial f}{\partial W_i} &= \frac{\partial f}{\partial x_i^{(1)}}\frac{\partial x_i^{(1)}}{\partial W_i} && (19)\\
&= \phi'_{\text{out}}(z^{(2)})\beta_i\phi'(z_i)x && (20)\\
-\frac{\partial \mathcal{L}}{\partial W_i} &= \frac{\partial \mathcal{L}}{\partial f}\frac{\partial f}{\partial W_i} && (21)\\
&= (y_* - \phi_{\text{out}}(z^{(2)}))\phi'_{\text{out}}(z^{(2)})\beta_i\phi'(z_i)x && (22)\\
& && (23)
\end{aligned}
$$

$z_i = W_i x$

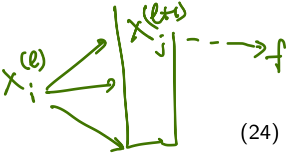
$\frac{\partial z_i}{\partial x} = W_i$



From 2 layers to L layers



$|z|$ large $\Rightarrow \varphi' \approx 0$ "saturation"



$$\frac{\partial x_i^{(l)}}{\partial z_i^{(l)}} = \phi'(z_i^{(l)}) \qquad (24)$$

$$\frac{\partial x_i^{(l)}}{\partial W_i^{(l)}} = \phi'(z_i^{(l)})x^{(l-1)} \qquad (25)$$

$z_i^{(\ell)} = W_i^{(\ell)} x^{(\ell-1)}$

used for $\dfrac{\partial \mathcal{L}}{\partial W^{(\ell-1)}}$

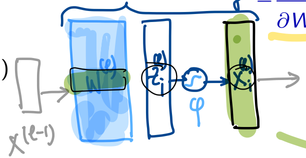
$$\frac{\partial x_i^{(l)}}{\partial x_i^{(l-1)}} = \phi'(z_i^{(l)})W_i^{(l)} \qquad (26)$$

$$\frac{\partial f}{\partial x_i^{(l)}} = \sum_{j=1}^{m_{\ell+1}} \frac{\partial f}{\partial x_j^{(l+1)}} \frac{\partial x_j^{(l+1)}}{\partial x_i^{(l)}} \qquad (27)$$

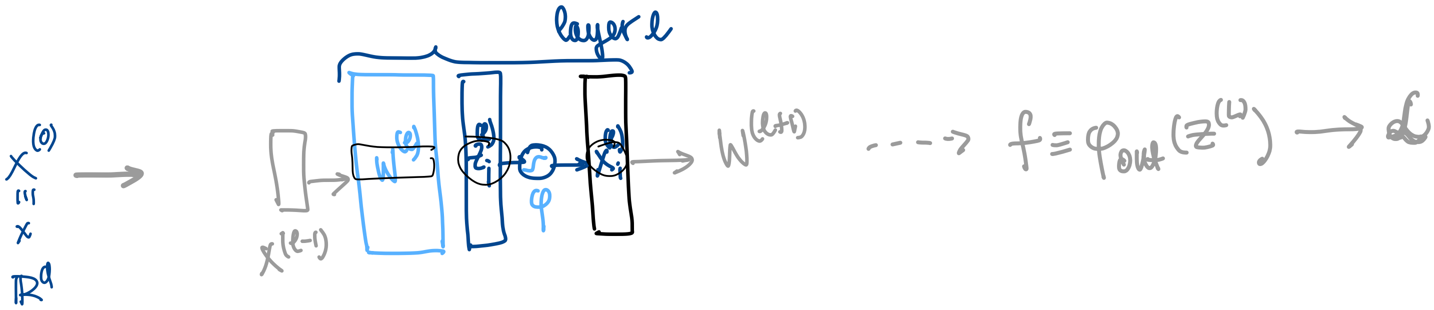
$$\frac{\partial f}{\partial W_i^{(l)}} = \frac{\partial f}{\partial x_i^{(l)}} \frac{\partial x_i^{(l)}}{\partial W_i^{(l)}} \qquad (28)$$

layer ℓ

$$\frac{\partial \mathcal{L}}{\partial W_i^{(l)}} = (y_* - \phi_{\text{out}}) \frac{\partial f}{\partial W_i^{(l)}} \qquad (29)$$



$\to W^{(\ell+1)} \dashrightarrow f \equiv \varphi_{out}(z^{(L)}) \to \mathcal{L}$



for $t = 1, 2, \dots$

for $k = 1:n$ (examples)

FORWARD (Prediction)

for $\ell = 1:L$ — store!

compute $x^{(\ell)}$, $z^{(\ell)}$ from $x^{(\ell-1)}$, $W^{(\ell)}$

$f = \varphi_{out}(z^{(L)})$, $\mathcal{L}^{train} \mathrel{+}= \frac{1}{n}\mathcal{L}(y^i, f(x^i))$ → used for stopping not $\frac{\partial \mathcal{L}}{\partial W}$

BACK PROP.

for $\ell = L:-1:1$

compute $\frac{\partial \mathcal{L}}{\partial W_i^{(\ell)}}$ from $x^{(\ell)}$, $\frac{\partial f}{\partial x^{(\ell)}}$, $W^{(\ell)}$, $\frac{\partial x^{(\ell+1)}}{\partial x^{(\ell)}}$ $\quad \frac{\partial \mathcal{L}}{\partial f}$

$i = 1:m_\ell$

$\frac{\partial x^{(\ell)}}{\partial x^{(\ell-1)}}$ → for $\frac{\partial \mathcal{L}}{\partial W^{(\ell-1)}}$