ErSE222: Machine learning in Geoscience

Feb. 16th, 2025

Tariq Alkhalifah and Omar Saad

Backpropagation (BCE)

General algorithm, called **backpropagation →** used to compute gradients of any complex function that be composed of simple, differentiable functions

$$z = \mathbf{x}^T\boldsymbol{\theta}, \quad a = \sigma(z), \quad \mathscr{L} = -(ylog(a) + (1-y)log(1-a))$$

$$\Big\downarrow \text{ Chain-rule}$$

$$\frac{\partial \mathscr{L}}{\partial \boldsymbol{\theta}} = \frac{\partial \mathscr{L}}{\partial a}\frac{\partial a}{\partial z}\frac{\partial z}{\partial \boldsymbol{\theta}}$$

Backpropagation (BCE)

Let’s now (for once) do the math:

$$\boxed{\mathcal{L} = -(y log(a) + (1-y) log(1-a))}$$

$$\frac{\partial \mathcal{L}}{\partial a} = -\frac{y}{a} + \frac{1-y}{1-a} = \frac{-y(1-a) + (1-y)a}{a(1-a)}$$

$$\frac{\partial a}{\partial z} = a(1-a) \quad \boxed{a = \sigma(z)}$$

Backpropagation (BCE)

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

$$u = 1 + e^{-x}$$

$$y = \frac{1}{u}$$

$$\frac{dy}{du} = -\frac{1}{u^2}$$

$$\frac{du}{dx} = -e^{-x}$$

Backpropagation (BCE)

$$\frac{dy}{dx} = \frac{dy}{du} \cdot \frac{du}{dx}$$

$$\frac{dy}{du} = -\frac{1}{u^2} \qquad \frac{du}{dx} = -e^{-x}$$

$$\frac{dy}{dx} = \frac{e^{-x}}{u^2}$$

$$\frac{dy}{dx} = \frac{e^{-x}}{(1+e^{-x})^2}$$

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

$$\sigma'(x) = \sigma(x) \cdot (1-\sigma(x))$$

Backpropagation (BCE)

Let's now (for once) do the math:

$$\boxed{\mathscr{L} = -(ylog(a) + (1-y)log(1-a))}$$

$$\frac{\partial \mathscr{L}}{\partial a} = -\frac{y}{a} + \frac{1-y}{1-a} = \frac{-y(1-a) + (1-y)a}{a(1-a)}$$

$$\frac{\partial a}{\partial z} = a(1-a) \quad \boxed{a = \sigma(z)}$$

$$\downarrow$$

$$\frac{\partial \mathscr{L}}{\partial z} = \frac{\partial \mathscr{L}}{\partial a}\frac{\partial a}{\partial z} = -y(1-a) + (1-y)a = a - y = dz$$

Backpropagation (BCE)

$$z = \mathbf{x}^T\boldsymbol{\theta}, \quad a = \sigma(z), \quad \mathscr{L} = -(ylog(a) + (1-y)log(1-a))$$

Let’s now (for once) do the math:

$$\frac{\partial \mathscr{L}}{\partial z} = \frac{\partial \mathscr{L}}{\partial a}\frac{\partial a}{\partial z} = -y(1-a) + (1-y)a = a - y = dz$$

$$\frac{\partial z}{\partial w_i} = x_i, \quad \frac{\partial z}{\partial b} = 1$$

$$\downarrow$$

$$\boxed{\frac{\partial \mathscr{L}}{\partial w_i} = dz \cdot x_i = dw_i, \quad \frac{\partial \mathscr{L}}{\partial b} = dz = db}$$

Backpropagation (BCE)

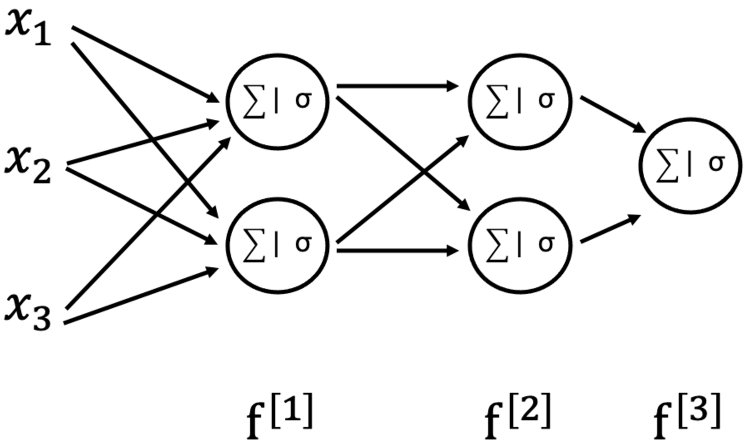
$$\left.\begin{aligned}
\mathbf{z} &= \mathbf{X}_{train}^{T}\boldsymbol{\theta} \\
\mathbf{a} &= \sigma(\mathbf{z})
\end{aligned}\right\}\ \text{Forward}$$

$$\left.\begin{aligned}
\mathbf{dz} &= \mathbf{a} - \mathbf{y} \\
\mathbf{dw} &= \frac{1}{N_s}\mathbf{X}_{train}\mathbf{dz} \\
db &= \frac{1}{N_s}\mathbf{1}^{T}\mathbf{dz}
\end{aligned}\right\}\ \text{Backward}$$

$$\left.\begin{aligned}
\mathbf{w} &\leftarrow \mathbf{w} - \alpha\mathbf{dw} \\
b &\leftarrow b - \alpha db
\end{aligned}\right\}\ \text{Update}$$

MLP

Backpropagation (BCE)



$$f_\theta(\mathbf{x}) = \mathrm{f}^{[3]}\left(\mathrm{f}^{[2]}\left(\mathrm{f}^{[1]}(\mathbf{x})\right)\right)$$

$$\downarrow$$

$$\mathcal{L}\left(\mathrm{y}, f_\theta(\mathbf{x})\right)$$

Forward Propagation $\longrightarrow$ $\mathbf{x} = \mathcal{L}\left(\mathrm{y}, f_\theta(\mathbf{x})\right)$

$\longleftarrow$ Backward Propagation $\frac{\partial \mathcal{L}}{\partial W^{\{3\}}}, \ldots, \partial \mathcal{L}/\partial b^{\{1\}}$

Backpropagation (BCE)

$$
\begin{tikzcd}
 & \mathbf{W}^{[1]} \quad \mathbf{b}^{[1]} \arrow[d] & & \mathbf{W}^{[2]} \quad \mathrm{b}^{[2]} \arrow[d] & & \\
\mathbf{x} \arrow[r] & \boxed{\mathbf{z}^{[1]} = \mathbf{W}^{[1]}\mathbf{x} + \mathbf{b}^{[1]}} \arrow[r] & \boxed{\mathbf{a}^{[1]} = \sigma\left(\mathbf{z}^{[1]}\right)} \arrow[r] & \boxed{\mathrm{z}^{[2]} = \mathbf{W}^{[2]}\mathbf{a}^{[1]} + \mathrm{b}^{[2]}} \arrow[r] & \boxed{\mathrm{a}^{[2]} = \sigma\left(\mathrm{z}^{[2]}\right)} \arrow[r] & \boxed{\mathcal{L}\left(\mathrm{y}, \mathrm{a}^{[2]}\right)}
\end{tikzcd}
$$

And write in reversed order (evaluated right to left):

$$
\frac{\partial \mathcal{L}}{\partial \mathbf{W}^{[1]}} = \frac{\partial \boldsymbol{z}^{[1]}}{\partial \mathbf{W}^{[1]}} \frac{\partial \boldsymbol{a}^{[1]}}{\partial \boldsymbol{z}^{[1]}} \frac{\partial z^{[2]}}{\partial \boldsymbol{a}^{[1]}} \frac{\partial a^{[2]}}{\partial z^{[2]}} \frac{\partial \mathcal{L}}{\partial a^{[2]}}
$$

$$\longleftarrow$$

Effective order of operations

Backpropagation (BCE)

$$\mathbf{z} = \mathbf{X}_{train}^T \boldsymbol{\theta}$$

$$\mathbf{a} = \sigma(\mathbf{z})$$

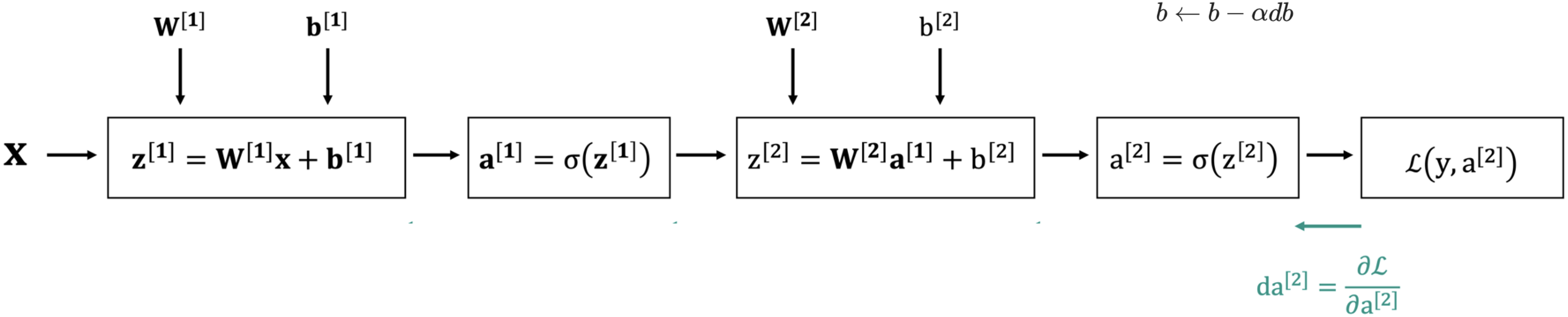
$$\mathbf{dz} = \mathbf{a} - \mathbf{y}$$

$$\mathbf{dw} = \frac{1}{N_s} \mathbf{X}_{train} \mathbf{dz}$$

$$db = \frac{1}{N_s} \mathbf{1}^T \mathbf{dz}$$

$$\mathbf{w} \leftarrow \mathbf{w} - \alpha \mathbf{dw}$$

$$b \leftarrow b - \alpha db$$



Backpropagation (BCE)

$$\mathbf{z} = \mathbf{X}_{train}^T \boldsymbol{\theta}$$

$$\mathbf{a} = \sigma(\mathbf{z})$$

$$\mathbf{dz} = \mathbf{a} - \mathbf{y}$$

$$\mathbf{dw} = \frac{1}{N_s} \mathbf{X}_{train} \mathbf{dz}$$

$$db = \frac{1}{N_s} \mathbf{1}^T \mathbf{dz}$$

$$\mathbf{w} \leftarrow \mathbf{w} - \alpha \mathbf{dw}$$

$$b \leftarrow b - \alpha db$$

$$\frac{\partial \mathscr{L}}{\partial z} = \frac{\partial \mathscr{L}}{\partial a} \frac{\partial a}{\partial z} = -y(1-a) + (1-y)a = a - y = dz$$

$$\begin{array}{ccccccccc}
 & \mathbf{W}^{[1]} \quad \mathbf{b}^{[1]} & & & & \mathbf{W}^{[2]} \quad \mathrm{b}^{[2]} & & & \\
 & \downarrow \qquad \downarrow & & & & \downarrow \qquad \downarrow & & & \\
\mathbf{X} \rightarrow & \boxed{\mathbf{z}^{[1]} = \mathbf{W}^{[1]}\mathbf{x} + \mathbf{b}^{[1]}} & \rightarrow & \boxed{\mathbf{a}^{[1]} = \sigma(\mathbf{z}^{[1]})} & \rightarrow & \boxed{\mathrm{z}^{[2]} = \mathbf{W}^{[2]}\mathbf{a}^{[1]} + \mathrm{b}^{[2]}} & \rightarrow & \boxed{\mathrm{a}^{[2]} = \sigma(\mathrm{z}^{[2]})} & \rightarrow \boxed{\mathcal{L}(\mathrm{y}, \mathrm{a}^{[2]})} \\
 & & & & \leftarrow & & \leftarrow & &
\end{array}$$

$$\mathrm{dz}^{[2]} = \frac{\partial \mathrm{a}^{[2]}}{\partial \mathrm{z}^{[2]}} \mathrm{da}^{[2]} = \mathrm{a}^{[2]} - y \quad *$$

$$\mathrm{da}^{[2]} = \frac{\partial \mathcal{L}}{\partial \mathrm{a}^{[2]}}$$

* Assuming $\mathcal{L}$=BCE

Backpropagation (BCE)

$$\boxed{\frac{\partial \mathcal{L}}{\partial w_i} = dz \cdot x_i = dw_i, \qquad \frac{\partial \mathcal{L}}{\partial b} = dz = db}$$

$$\mathbf{z} = \mathbf{X}_{train}^T \boldsymbol{\theta}$$

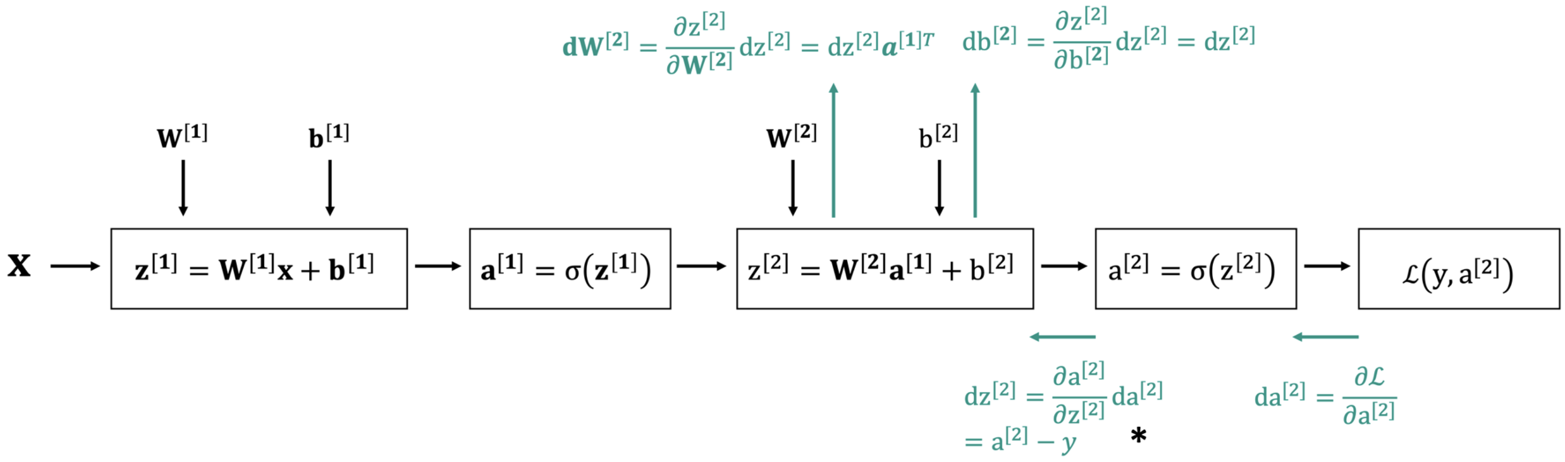
$$\mathbf{a} = \sigma(\mathbf{z})$$

$$\mathbf{dz} = \mathbf{a} - \mathbf{y}$$

$$\mathbf{dw} = \frac{1}{N_s}\mathbf{X}_{train}\mathbf{dz}$$

$$db = \frac{1}{N_s}\mathbf{1}^T\mathbf{dz}$$

$$\mathbf{w} \leftarrow \mathbf{w} - \alpha\mathbf{dw}$$



* Assuming $\mathcal{L}$=BCE

Backpropagation (BCE)

$$\boxed{\frac{\partial \mathscr{L}}{\partial w_i} = dz \cdot x_i = dw_i, \qquad \frac{\partial \mathscr{L}}{\partial b} = dz = db}$$

$$\mathbf{z} = \mathbf{X}_{train}^T \boldsymbol{\theta}$$

$$\mathbf{a} = \sigma(\mathbf{z})$$

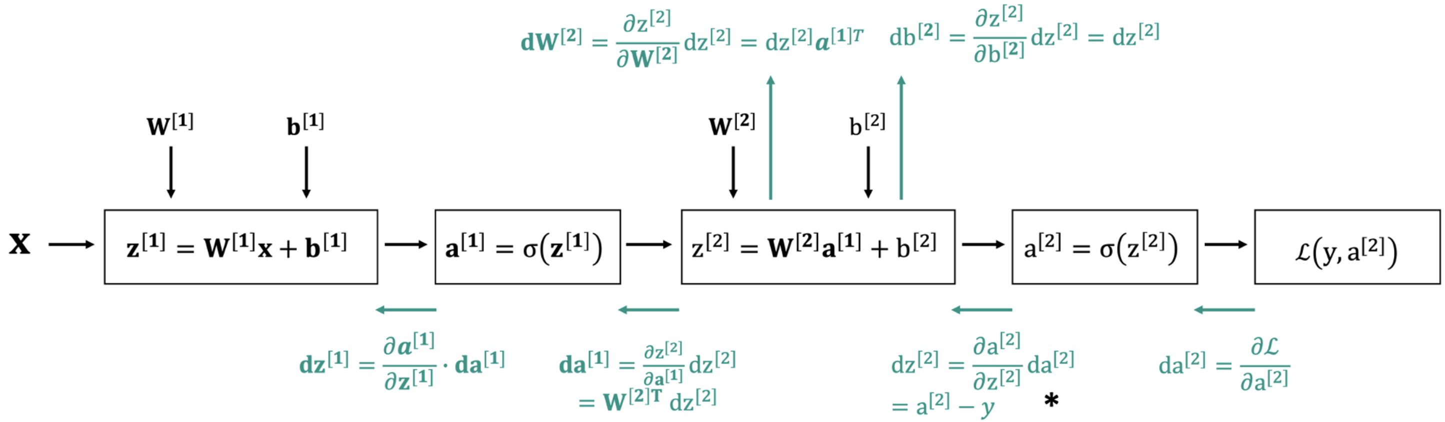
$$\mathbf{dz} = \mathbf{a} - \mathbf{y}$$

$$\mathbf{dw} = \frac{1}{N_s}\mathbf{X}_{train}\mathbf{dz}$$

$$db = \frac{1}{N_s}\mathbf{1}^T\mathbf{dz}$$

$$\mathbf{w} \leftarrow \mathbf{w} - \alpha\mathbf{dw}$$

$$b \leftarrow b - \alpha db$$



* Assuming $\mathcal{L}$=BCE

Backpropagation (BCE)

$$\boxed{\frac{\partial \mathscr{L}}{\partial w_i} = dz \cdot x_i = dw_i, \qquad \frac{\partial \mathscr{L}}{\partial b} = dz = db}$$

$$\mathbf{z} = \mathbf{X}_{train}^T \boldsymbol{\theta}$$

$$\mathbf{a} = \sigma(\mathbf{z})$$

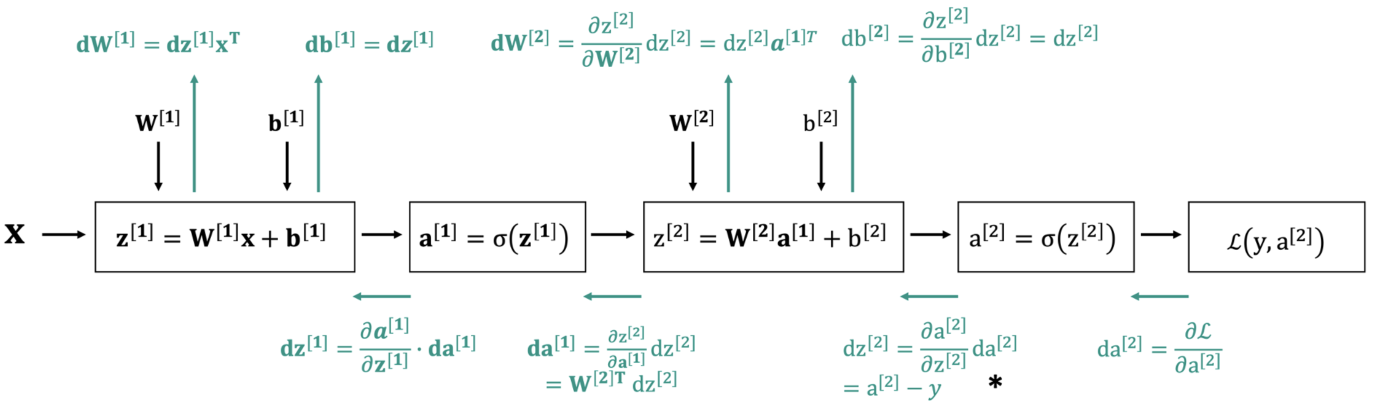
$$\mathbf{dz} = \mathbf{a} - \mathbf{y}$$

$$\mathbf{dw} = \tfrac{1}{N_s} \mathbf{X}_{train} \mathbf{dz}$$

$$db = \tfrac{1}{N_s} \mathbf{1}^T \mathbf{dz}$$

$$\mathbf{w} \leftarrow \mathbf{w} - \alpha \mathbf{dw}$$

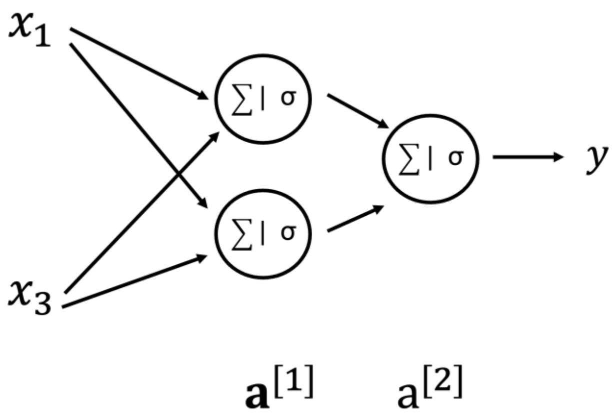
$$b \leftarrow b - \alpha db$$



* Assuming $\mathcal{L}$=BCE

Initialization

- **Zero (or constant) initialization**: symmetry problem (aka symmetric gradients)



Initialization:

$$\left\{\mathbf{W}^{[1]} = c^{[1]}\mathbf{1}, \mathbf{b}^{[1]} = c^{[1]}\mathbf{1}, \mathbf{w}^{[2]} = c^{[2]}\mathbf{1}, b^{[2]} = c^{[2]}\right.$$

$$\downarrow$$

$$a_1^{[1]} = a_2^{[1]} \longrightarrow dz_1^{[1]} = dz_2^{[1]}$$

$$\longrightarrow dW_{11}^{[1]} = dW_{21}^{[1]}, dW_{12}^{[1]} = dW_{22}^{[1]}$$

$$\longrightarrow W_{11}^{[1]} = W_{21}^{[1]}, W_{12}^{[1]} = W_{22}^{[1]}$$

Initialization

- **Random initialization**: general idea

$$w_{i,j}^{[.]} \sim \mathcal{N}(0, 0.01)$$

Not too small: to avoid slow training
Not too high: to avoid unstable training

$$b_i^{[.]} = 0$$ (or constant - e.g. for ReLU)

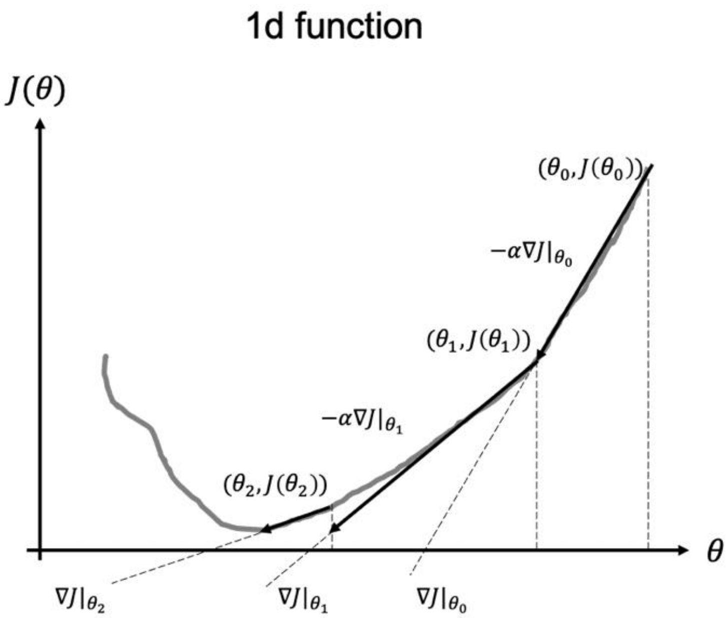
Initialization

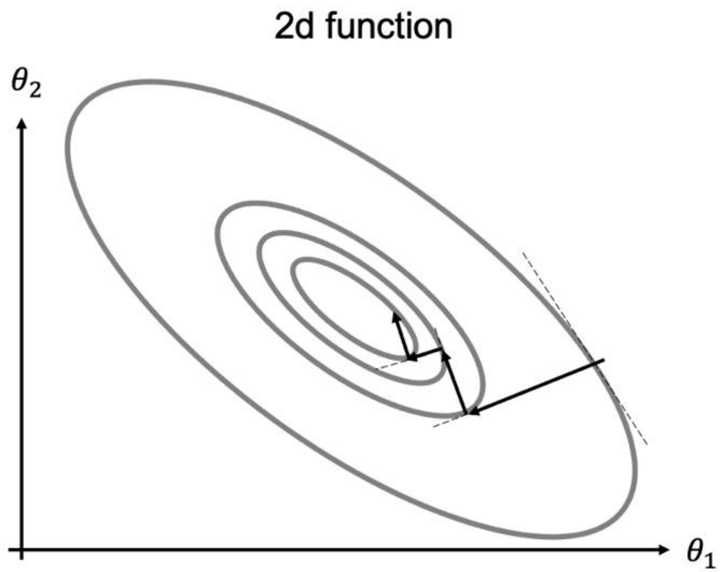
- **Random initialization**: more appropriate choices (depend on #units in layer)

- Uniform distributior $w_{ij}^{[k]} \sim \mathcal{U}(-1/\sqrt{N^{[k]}}, 1/\sqrt{N^{[k]}})$
- Xavier initialization $w_{ij}^{[k]} \sim \mathcal{N}(0, 1/N^{[k]})$
- He initialization $w_{ij}^{[k]} \sim \mathcal{N}(0, 2/N^{[k]})$

Optimization

Gradient-based optimization





Use local information of $f(x)$ to find its minimum/maximum

Gradient-based optimization

Gradient descent algorithm:

Initialization: choose $\theta \in \mathbb{R}$

For $i = 0, \ldots N-1$;

1. Compute update direction $d_i = -\nabla J|_{\theta_i}$
2. Estimate step-lenght α_i
3. Update $\theta_{i+1} = \theta_i + \alpha_i d_i$

Choice of step-length:

- Constant or with some schedule (in DL)
- Exact
- Backtracking

Extension to n-dimensional problems:

In DL applications, $\theta = [\theta_1, \theta_2, \ldots, \theta_N]$

$$\nabla J = [\delta J/\delta\theta_1, \delta J/\delta\theta_2, \ldots, \delta J/\delta\theta_N].$$

Stochastic gradient descent (SGD)

Batched gradient descent:

$$\boldsymbol{\theta}_{i+1} = \boldsymbol{\theta}_i - \alpha_i \nabla J = \boldsymbol{\theta}_i - \frac{\alpha_i}{N_s} \sum_{j=1}^{N_s} \nabla \mathscr{L}_j$$

Stochastic gradient descent:

$$\boldsymbol{\theta}_{i+1} = \boldsymbol{\theta}_i - \alpha_i \nabla \mathscr{L}_j$$

Mini-batch gradient descent:

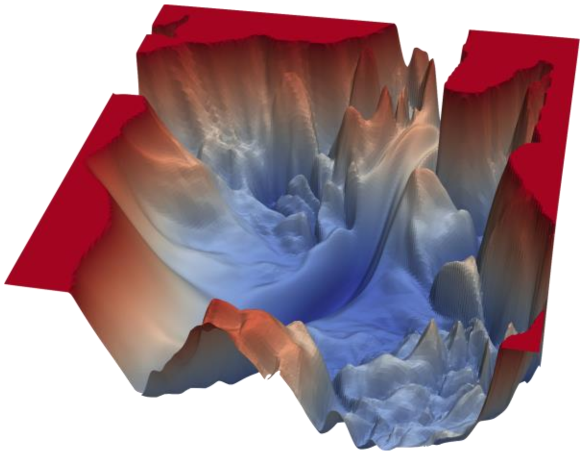
Mini-batch size

$$\boldsymbol{\theta}_{i+1} = \boldsymbol{\theta}_i - \frac{\alpha_i}{N_b} \sum_{j=1}^{N_b} \nabla \mathscr{L}_j$$

Limitations of SGD

- **Ill-conditioning:** In an ill-conditioned problem, the gradients may change dramatically with small changes in the parameter values. This results in steep gradients in certain directions and shallow gradients in others.

- **Local minima**: the landscape of NN functionals is generally non-convex and populated with a multitude of local minima



Model identifiability

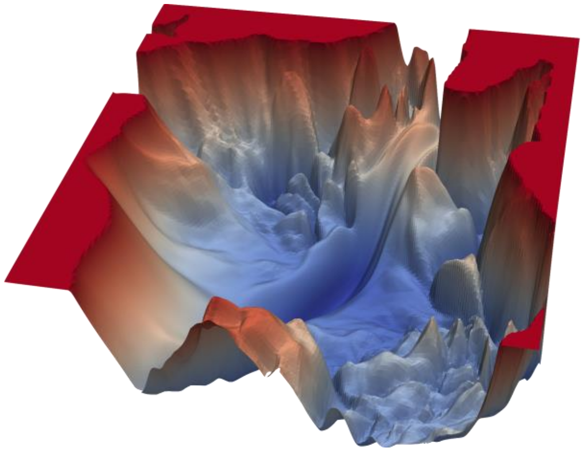
$\boldsymbol{\theta}_{gm}$ *Single set of params for optimal performance*

$\boldsymbol{\theta}_{gm}, \boldsymbol{\theta}_{lm,1}, \cdots$ *Multiple models with similar performance*

https://www.cs.umd.edu/~tomg/projects/landscapes/

Limitations of SGD

- **Saddle points (and other flat regions)**: landscapes associated to the training of deep neural networks may have much fewer local minima than we tend to believe… the main hinder to slow training is represented by saddle points

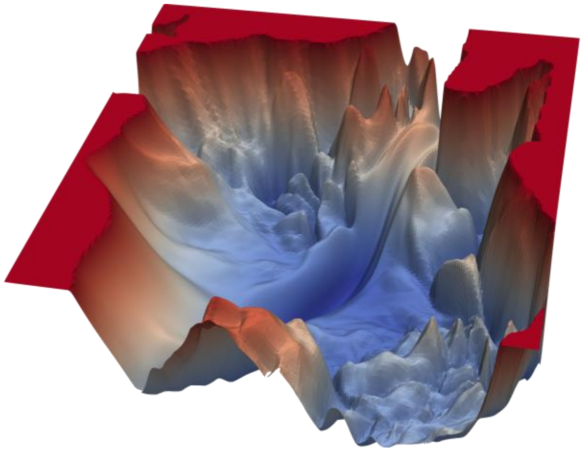


https://www.cs.umd.edu/~tomg/projects/landscapes/

$J(\theta) \to 0$ Vanishing gradients

Limitations of SGD

- **Cliffs:** steep regions in the NN landscape with high gradient



https://www.cs.umd.edu/~tomg/projects/landscapes/

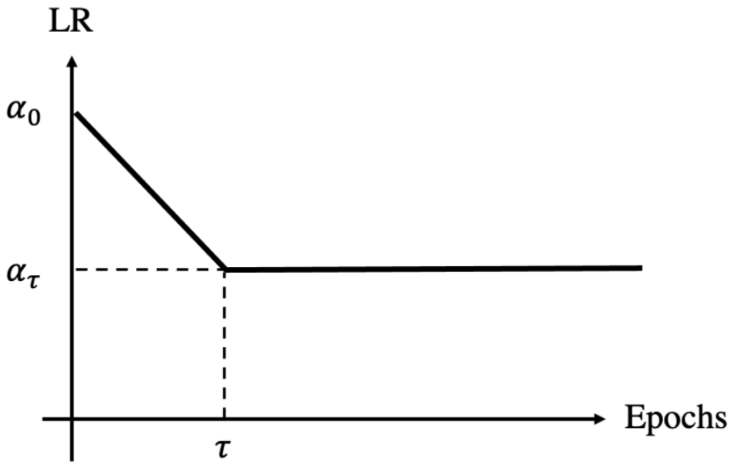
$J(\theta) \to \infty$ Exploding gradients

$\nabla J(\theta_i) = min(\nabla J(\theta_i), th)$ Gradient clipping

Cooling

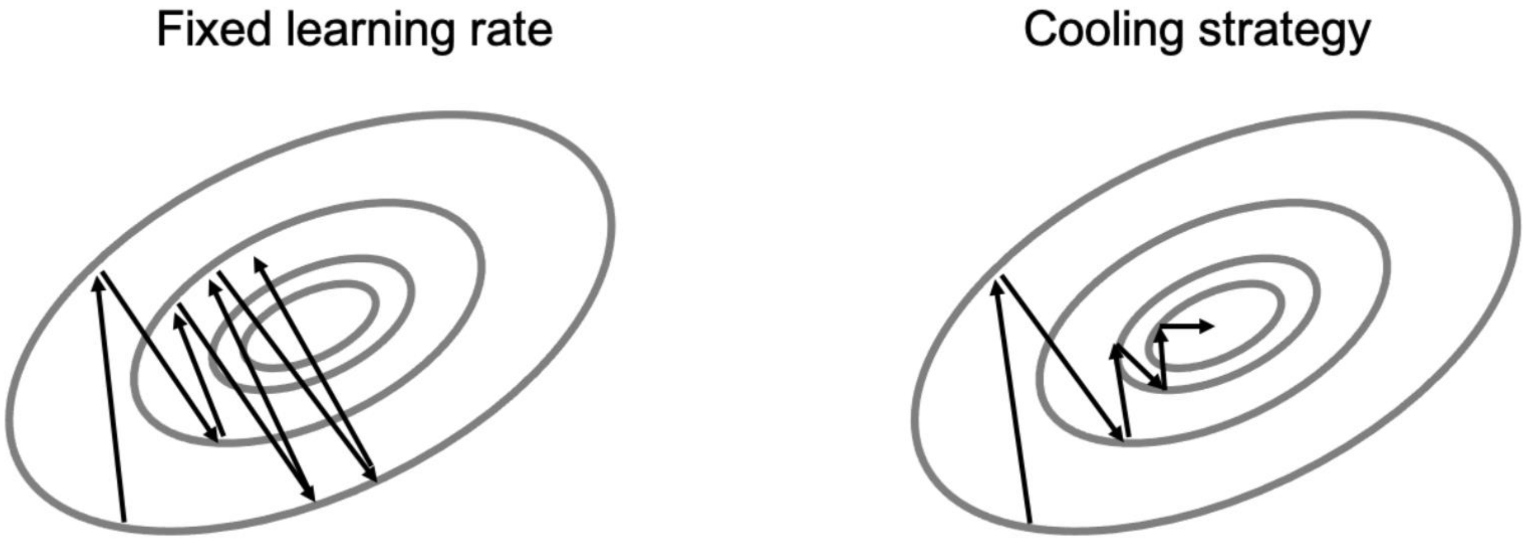
- Linearly decaying learning rate

$$\alpha_i = (1-\beta)\alpha_0 + \beta\alpha_\tau \qquad i < \tau$$
$$\alpha_i = \alpha_\tau \qquad i \geq \tau$$



$$\beta = i/\tau$$
$$\tau \approx 100 N_{epochs}, \alpha_\tau = \alpha_0/100$$

Cooling



SGD with momentum

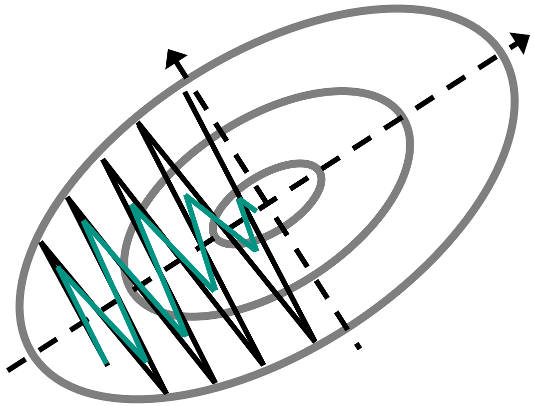
Idea of momentum (from Polyak and Nesterov in 60'): use update that is an exponentially decaying moving average of the past gradients.

Velocity

Momentum term [0,1)

$$\mathbf{v}_{i+1} = \gamma \mathbf{v}_i - \mathbf{g}_{i+1} = \gamma \mathbf{v}_i - \frac{\alpha}{N_b} \sum_{j=1}^{N_b} \nabla \mathscr{L}_j$$

$$\boldsymbol{\theta}_{i+1} = \boldsymbol{\theta}_i + \mathbf{v}_{i+1}$$

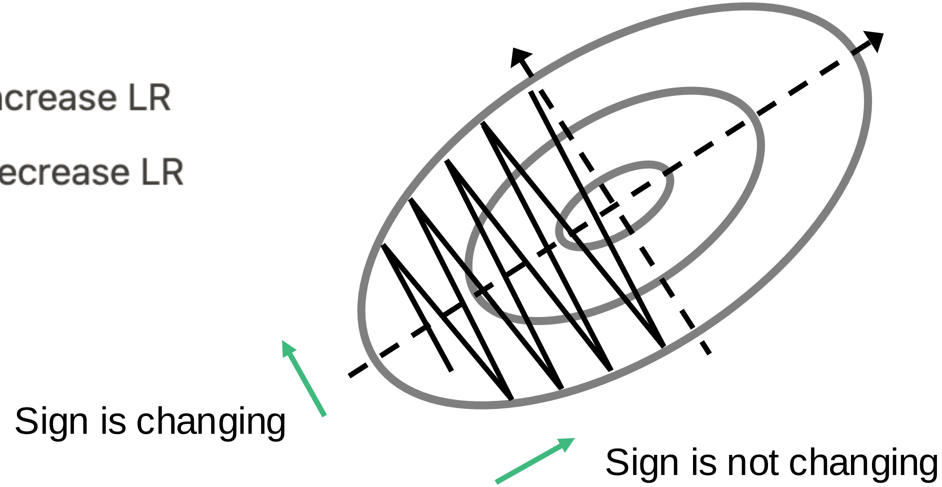


Adaptive Learning rates

Idea of adaptive LR : instead of scaling the entire gradient, different learning rates are adaptively applied to different directions

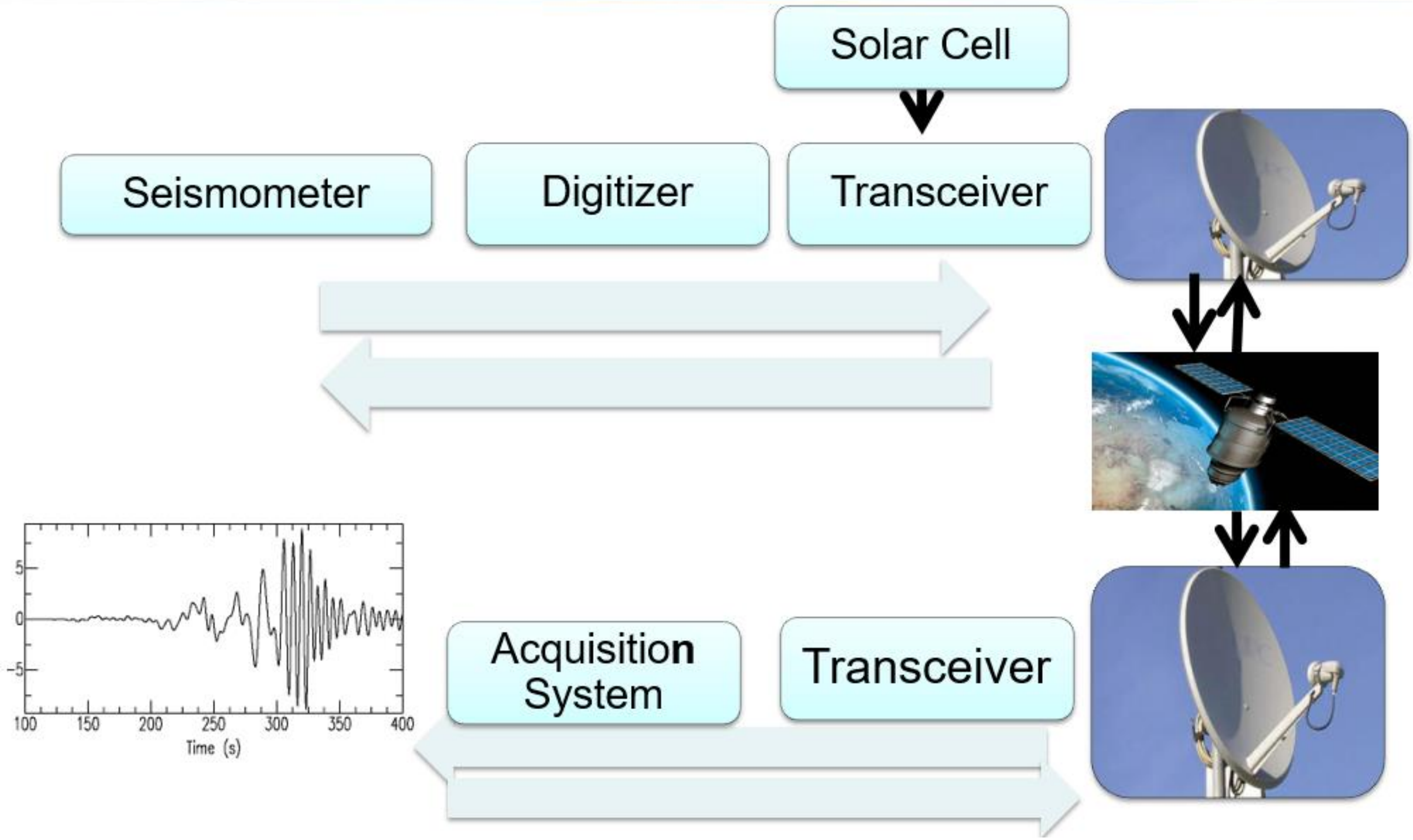
Delta-Bar-Delta algorithm

- if $sign\{g_{i+1}^{j}\} = sign\{g_{i}^{j}\}$, increase LR
- if $sign\{g_{i+1}^{j}\} \neq sign\{g_{i}^{j}\}$, decrease LR

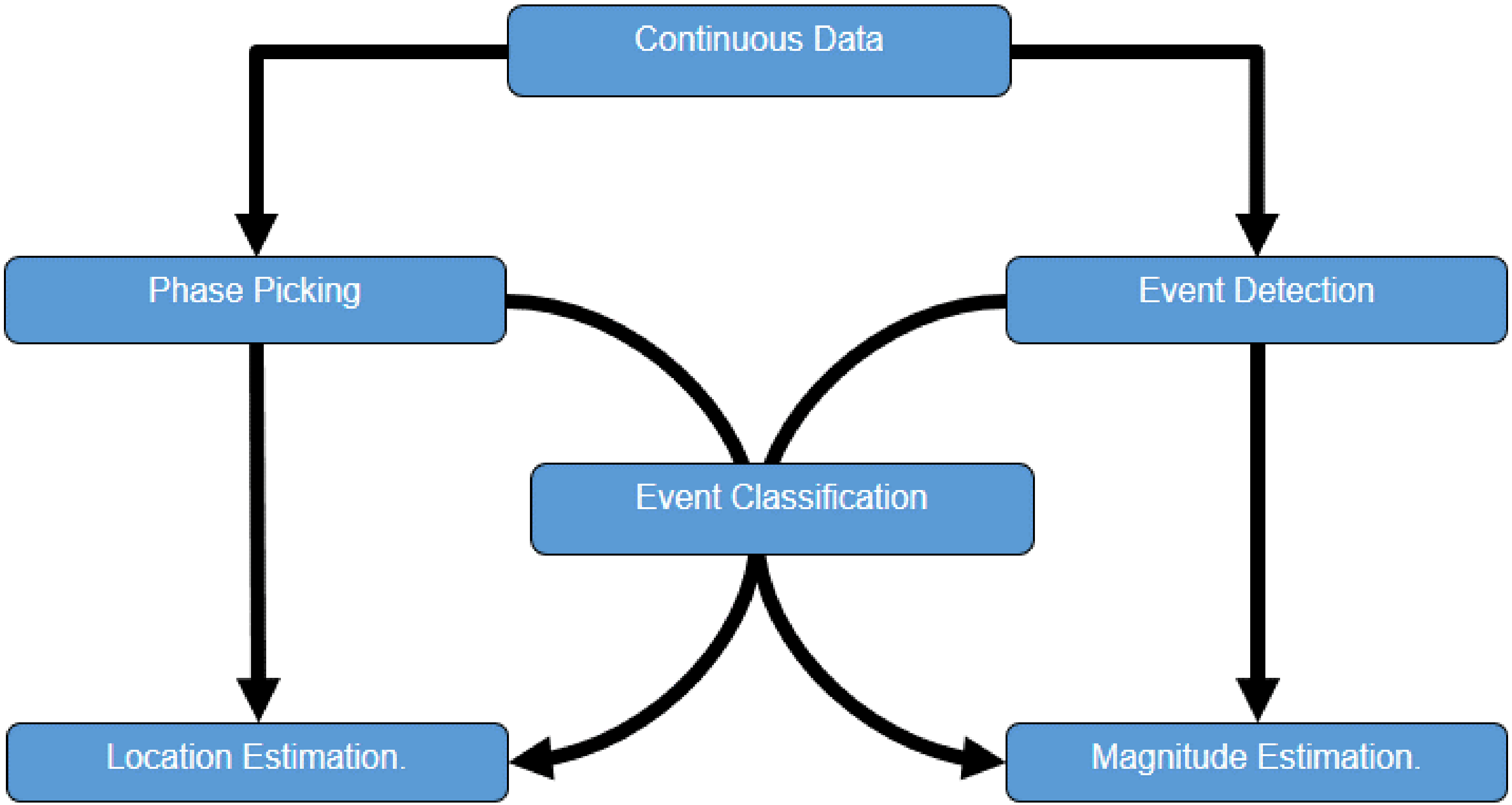


Application

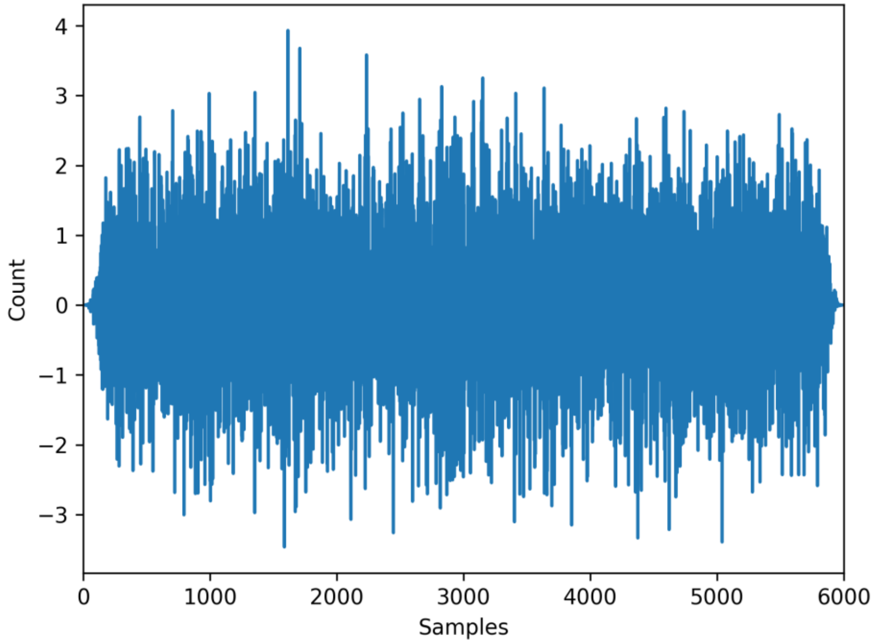
Earthquake Monitoring System

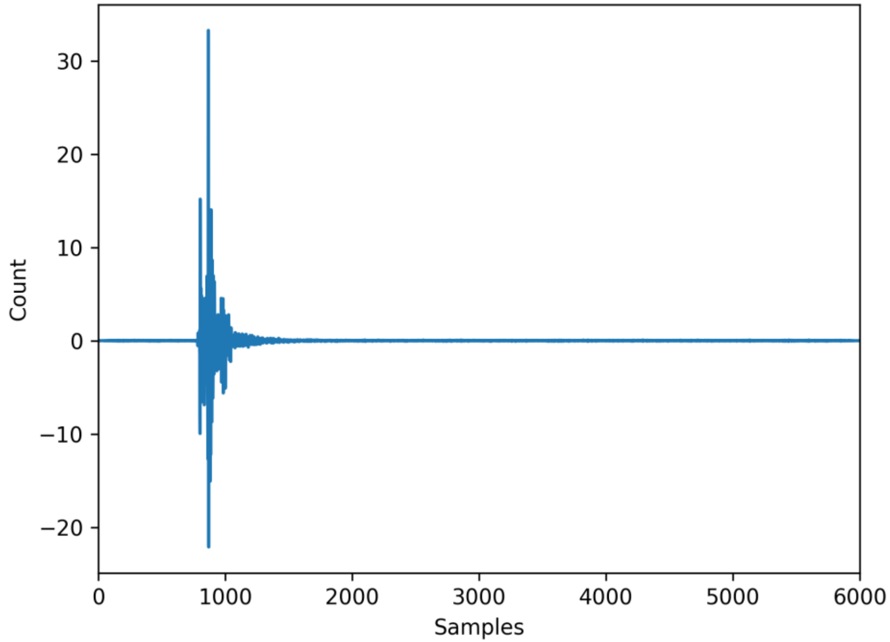


Earthquake Monitoring System

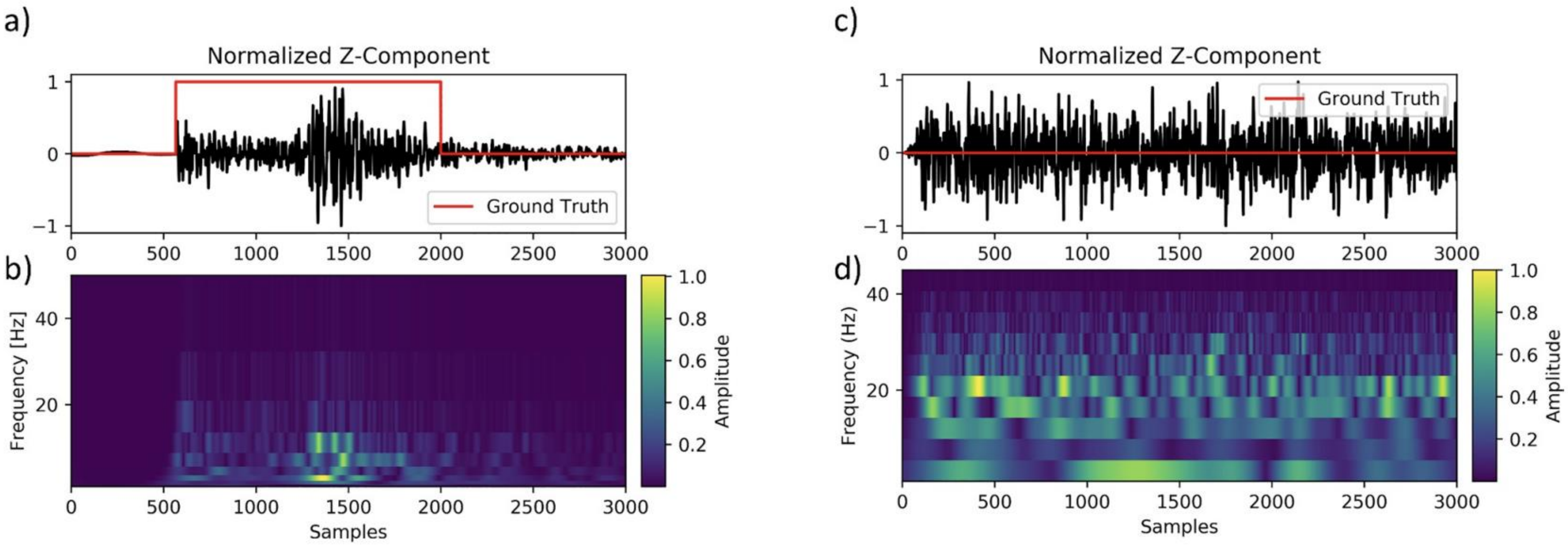


Seismic Event Classification

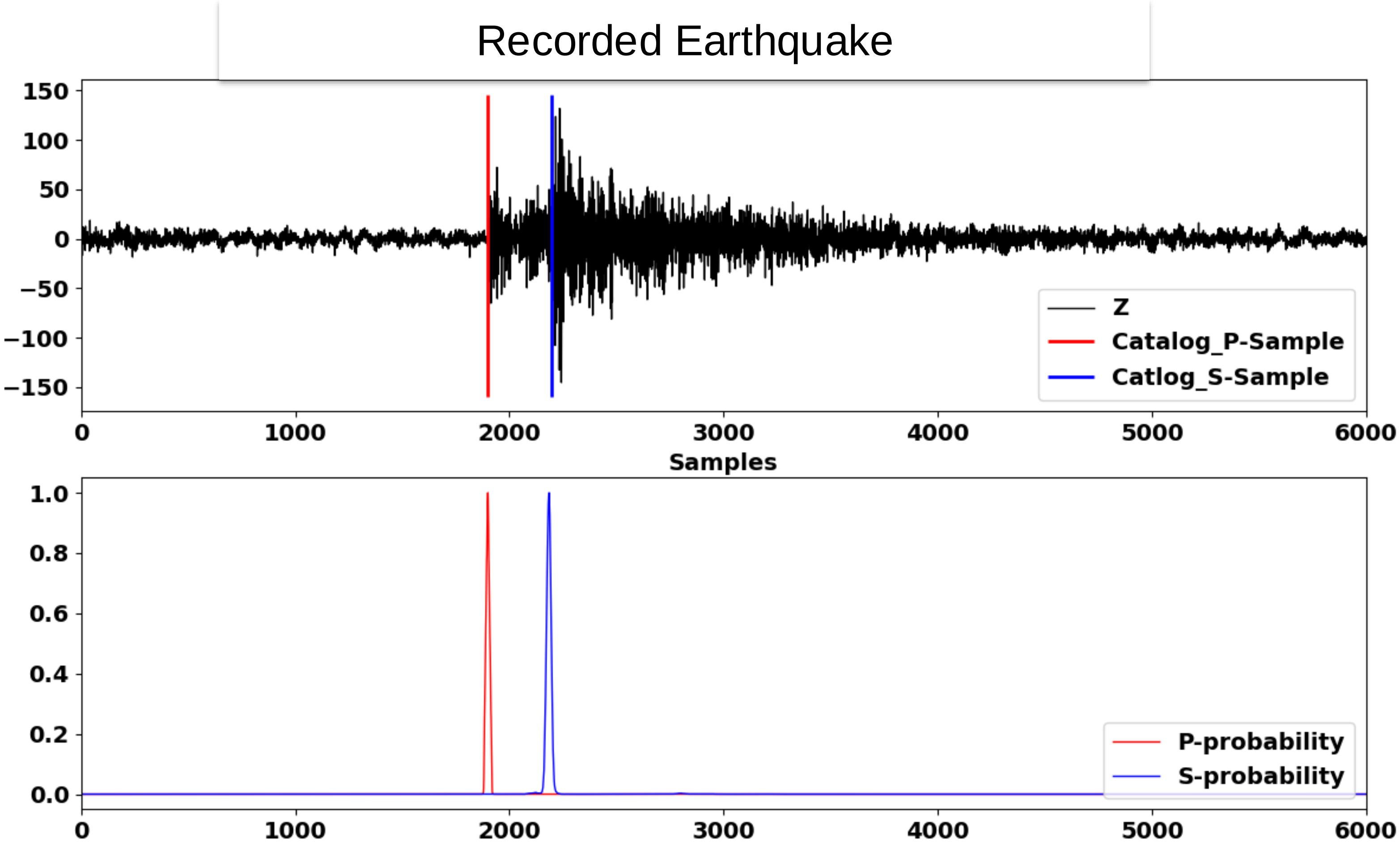




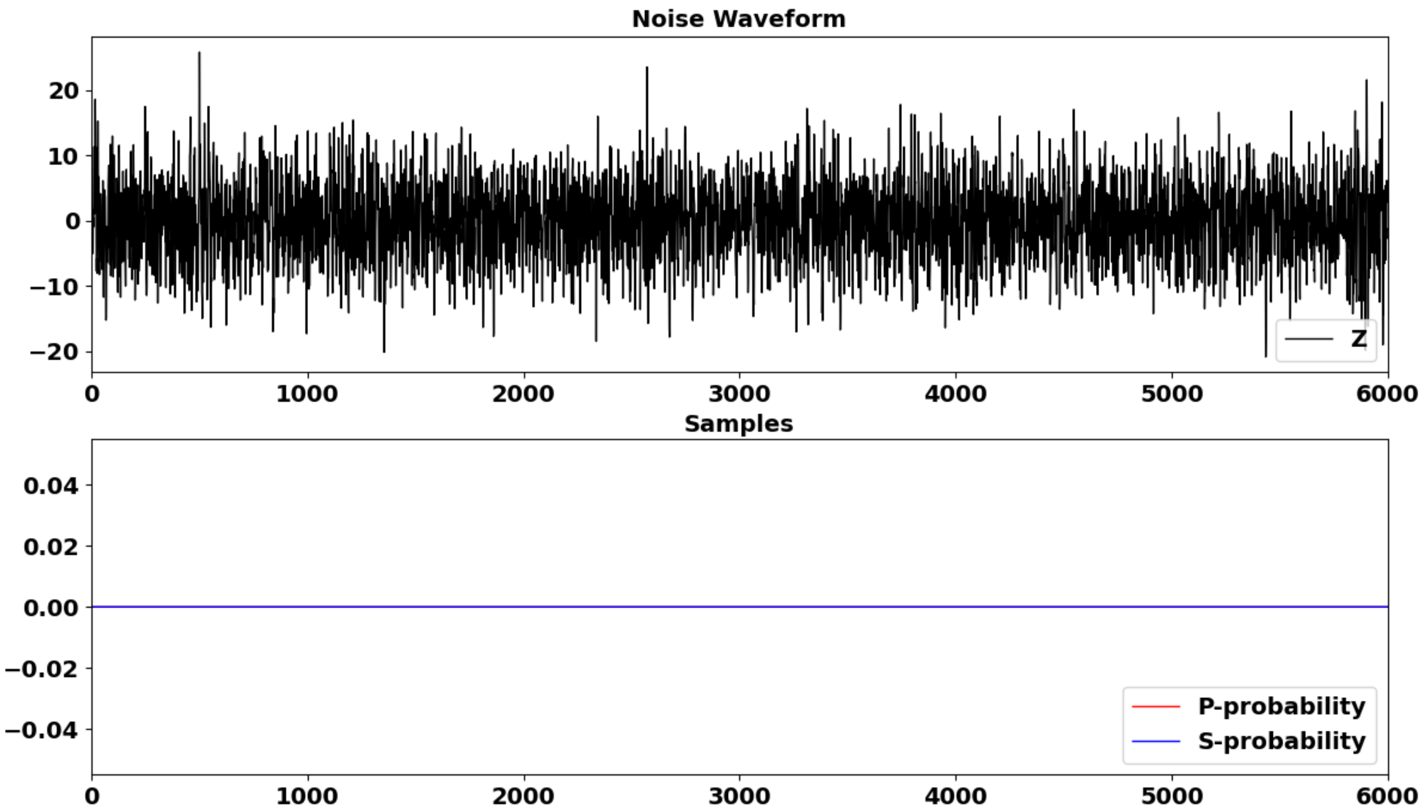
Sample-based Classification (Deteciton)



Sample-based Classification (Picking)



Sample-based Classification (Picking)




Thank You
30 17:50