# Domain generalization (2023)



$D_{tr}$ training set of $S$ domains

$$D_{tr} = \{D_1, \dots, D_S\}$$

$$\forall s = 1, \dots, S \quad D_s = \{(x_i^s, y_i^s)\}_{i=1}^{N_s}$$

↳ different distributions in the domains! (but similar)

$l(x, y; \theta)$ - loss of model $\theta$.

classical approach: empirical risk min $\mathcal{L}_{ERM} = \mathbb{E}_{D \sim D_{tr}} \mathbb{E}_{(x,y) \sim D} [l(x, y; \theta)]$

↳ fails to generalize on new domains

## (1) Pitfalls of ERM

- Binary classification
- $x \in \{0,1\}^4$, $y \in \{0,1\}$

| | $x_1$ 50% | $x_2$ 40% | $x_3$ 10% |
|---|---|---|---|
| $D_1$ | (0,0,0,0) | (1,1,0,0) | (1,0,0,0) |
| $D_2$ | (0,0,0,0) | (1,0,1,0) | (1,0,0,0) |
| $D_3$ | (0,0,0,0) | (1,0,0,1) | (1,0,0,0) |
| | $P(Y=1\|x_1)=0$ | $P(Y=1\|x_2)=1$ | $P(Y=1\|x_3)=0.3$ |

$x = (f_1, f_2, f_3, f_4)$. Accuracy of the optimal classifier considering only $f_1$?

Optimal classifier $y(f_3) = \begin{cases} 0 \text{ if } f_1 = 0 \text{ - obvious} \\ 1 \text{ if } f_1 = 1 \text{ - higher accuracy than for } y(1) = 0 \end{cases}$

-> $y^*(f_1) = f_1$

$$\text{Accuracy} = \mathbb{P}(y^*(f_1) = Y) = \mathbb{P}(Y=0|x_1)\mathbb{P}(x_1) + \mathbb{P}(Y=1|x_2)\mathbb{P}(x_2) + \dots$$

$+ \underbrace{\mathbb{P}(Y=1|x_3)}_{0.3}\mathbb{P}(x_3) = 0.93$

Non optimality of ERM principle: if we could take $x_i$ into account, we would predict $\begin{cases} y^*(f_1, x_3) = 1 - f_1 \\ y^*(f_1, x_i) = f_1 \quad i=1,2 \end{cases}$ and would get better accuracy $0.9 + 0.7 \cdot 0.1 = 0.97$.

(2) Meta-learning - alternative to ERM

$\mathcal{D}_{Tr} = \mathcal{D}_A \cup \mathcal{D}_B$ Train on $\mathcal{D}_A$ so it generalizes well on $\mathcal{D}_B$

$$\mathcal{L}_{\mathcal{D}_A}(\theta) = \mathbb{E}_{\mathcal{D}_A}[l(X,Y;\theta)] \qquad \mathcal{L}_{\mathcal{D}_B}(\theta) = \mathbb{E}_{\mathcal{D}_B}[l(X,Y;\theta)]$$

Meta-loss $\mathcal{L}_{M-L}(\theta) = \mathcal{L}_{\mathcal{D}_A}(\theta) + \mathcal{L}_{\mathcal{D}_B}(\underbrace{\theta - \alpha \nabla \mathcal{L}_{\mathcal{D}_A}(\theta)}_{\theta'})$

Gradient step $\theta'$ should minimize $\mathcal{L}_{\mathcal{D}_B}$

(2.1) Order 1 Taylor for $\mathcal{L}_{\mathcal{D}_B}(\theta') = \mathcal{L}_{\mathcal{D}_B}(\theta) + \nabla\mathcal{L}_{\mathcal{D}_B}(\theta) \cdot (-\alpha \nabla \mathcal{L}_{\mathcal{D}_A}(\theta)) + \bar{o}(|\theta - \theta'|)$

$$= \mathcal{L}_{\mathcal{D}_B}(\theta) - \alpha \langle \nabla \mathcal{L}_{\mathcal{D}_B}(\theta), \nabla \mathcal{L}_{\mathcal{D}_A}(\theta) \rangle + \bar{o}(|\theta - \theta'|)$$

(2.2) $\mathcal{L}_{M-L}(\theta) \approx \mathcal{L}_{\mathcal{D}_A}(\theta) + \mathcal{L}_{\mathcal{D}_B}(\theta) - \alpha \langle \nabla \mathcal{L}_{\mathcal{D}_A}(\theta), \nabla \mathcal{L}_{\mathcal{D}_B}(\theta) \rangle$

(2.3) Interpretation:
- Minimization of loss both on $\mathcal{D}_A$ and $\mathcal{D}_B$
- At the same time maximization of the $\langle \nabla \mathcal{L}_{\mathcal{D}_A}, \nabla \mathcal{L}_{\mathcal{D}_B} \rangle$ = colinearity of this vectors → simultaneous decrease of loss on $\mathcal{D}_A$ and $\mathcal{D}_B$ → ability to generalize on domains not included in the train set.

# 3. Gradient alignment

$$G_i := \mathbb{E}_{D_i}\left[\frac{\partial \ell(x,y;\theta)}{\partial \theta}\right]$$

$$\text{New loss} = \mathcal{L}_{ERM}(D_{Tr};\theta) - \gamma \frac{2}{N_{D_{Tr}}(N_{D_{Tr}}-1)} \sum_{i,j \in D_{Tr}} \langle G_i, G_j \rangle$$

($C$ = nb of sets $i \neq j$)

(3.1) Algo 2 is computationally prohibitive because of the double sum $\sum_{i,j}$ and the necessity to calculate $\frac{\partial \hat{G}}{\partial \theta}$ (numerically or in terms of Hessians)

(3.2) For two domains: explicit expression for updates

$$\text{Algo 1:} \quad \theta = \theta - \varepsilon\alpha\left(\underbrace{\mathbb{E}_{D_{P[1]}}[\partial_\theta(x,y;\theta)]}_{g_1} + \mathbb{E}_{D_{P[2]}}[\partial_\theta \ell(x,y;\theta-\alpha g_2)]\right)$$

$$\text{Algo 2:} \quad \theta = \theta - \varepsilon\left(\frac{1}{2}\sum_{i=1}^{2} \underbrace{\mathbb{E}_{D_i}[\partial_\theta \ell(x,y;\theta)]}_{\left(\frac{\partial x_i}{\partial \theta_j}\right) G_i} - 2\gamma\, \partial_\theta \langle G_1, G_2 \rangle\right)$$

$x \in \mathbb{R}^n$, $\theta \in \mathbb{R}^d$ ; $\nabla X(\theta)$ : $n \times d$ ; $\nabla Y(\theta)$ : matrix, $n \cdot d$

$$\langle X(\theta+\Delta\theta), Y(\theta+\Delta\theta)\rangle \simeq \langle X(\theta) + \nabla X(\theta)\cdot\Delta\theta,\ Y(\theta) + \nabla Y(\theta)\cdot\Delta\theta\rangle =$$

$$= \langle X(\theta), Y(\theta)\rangle + \underbrace{\left(Y(\theta)^T \cdot \nabla X(\theta) + X(\theta)^T \nabla Y(\theta)\right)}_{\nabla_\theta \langle X(\theta), Y(\theta)\rangle^T} \cdot \Delta\theta + \tilde{o}(\|\Delta\theta\|)$$

So, in our case $\nabla_\theta \langle G_1, G_2\rangle = \mathbb{E}_{D_2}\left[\nabla^2_\theta \ell(x,y;\theta)^T\right] G_1 + \mathbb{E}_{D_1}\left[\nabla^2_\theta \ell(x,y;\theta)^T\right] G_2$ ($\nabla^2$ = Hessian)

(3.3) Inner loop of Algo 1

$L_i := L(D_{P[i]}, \tilde{\theta}_i)$, $g_i = \nabla L_i(\tilde{\theta}_i)$, $\tilde{\theta}_{i+1} = \tilde{\theta}_i - \alpha g_i$

($\theta$ at the start of the inner loop)

$g_{i,1} = \nabla L_i(\theta_1)$ $\quad h_{i,1} = \nabla^2 L_i(\theta_1)$

(1.3.1) Show $g_i = \nabla L_i(\theta_1) + \nabla^2 L_i(\theta_1)(\tilde\theta_i - \theta_1) + O(\alpha^2) =$ (symmetric)

↑ Taylor for $\nabla L_i(\tilde\theta_i)$ — $\tilde\theta_i - \theta_1 = -\alpha \sum_{j=1}^{i-1} g_j - O(\alpha)$

(1.3.2) $= g_{i,1} - \alpha h_{i,1} \sum_{j=1}^{i-1} g_j + O(\alpha^2) = g_{i,1} - \alpha h_{i,1} \sum_{j=1}^{i-1} g_{j,1} + O(\alpha^2)$

↑ Taylor for $g_j = g_{j,1} - \alpha(\dots) + O(\alpha^2)$

Two domains

(1.3.3) Order: $\mathcal{D}_1, \mathcal{D}_2$ → $L_1 = L_1(\theta_1)$ $g_1 = \nabla L_1(\theta_1)$ $\tilde\theta_2 = \theta_1 - \alpha g_1$

$L_2 = L_2(\tilde\theta_2)$ $g_2 = \nabla L_2(\tilde\theta_2)$ $\tilde\theta_3 = \tilde\theta_2 - \alpha g_2$

Outer update

$\Delta\theta = \theta_1 - \tilde\theta_3 = \theta_1 - \tilde\theta_2 + \tilde\theta_2 - \tilde\theta_3 = \alpha(g_1 + g_2)$ (=)

$G_1(\theta_1)$ $G_2(\theta_1)$ $\nabla G_2(\theta_1)$

1.3.2.

(1.3.4) (=) $\alpha(\underbrace{g''_{1,1}}_{g_1} + \underbrace{g''_{2,1} - \alpha h_{2,1} g_{1,1}}_{g_2})$

(1.3.5) What would be the value of $g_{1,1}, g_{2,1}, h_{2,1}$ if the domains had been considered in order $\mathcal{D}_2, \mathcal{D}_1$?

$g_{1,1} = G_2(\theta_1)$ $g_{2,1} = G_1(\theta_1)$ $h_{2,1} = \nabla G_1(\theta_1)$

↑ iteration indices

(1.3.6) $\mathbb{E}_p[g_1 + g_2] = \frac{1}{2}\left(G_1(\theta_1) + G_2(\theta_1) - \alpha \nabla G_2(\theta_1) \cdot G_1(\theta_1)\right) + \frac{1}{2}(G_1(\theta_1) + G_2(\theta_1) - \alpha \nabla G_1(\theta_1) \cdot G_2(\theta_1)) + O(\alpha^2) =$

$= \frac{1}{2}(G_1 + G_2)(\theta_1) - \frac{\alpha}{2}\left(\nabla G_2(\theta_1) \cdot G_1(\theta_1) + \nabla G_1(\theta_1) G_2(\theta_1)\right) + O(\alpha^2)$

// see (3.2)

$\mathbb{E}_p[\Delta\theta] = \alpha \mathbb{E}_p[g_1 + g_2] = 2\alpha \bar G - \frac{\alpha^2}{2} \partial_{\theta_1}(G_1(\theta_1), G_2(\theta_1)) + O(\alpha^2)$

(1.3.7) For Algo 2 we had $\Delta\theta = \bar G - 2\gamma \nabla_\theta (G_1(\theta_1), G_2(\theta))$

So, for $\alpha = 2\sqrt{\gamma}$ it is actually an approximation of order $O(\gamma)$.

# Exercise (2022)

$w$ - r.v., vector of a NN

(1) Objective maximize $p(w|X,Y) \propto p(Y|X,w)p(w)$

$\mathcal{D} = (X, Y)$ training data

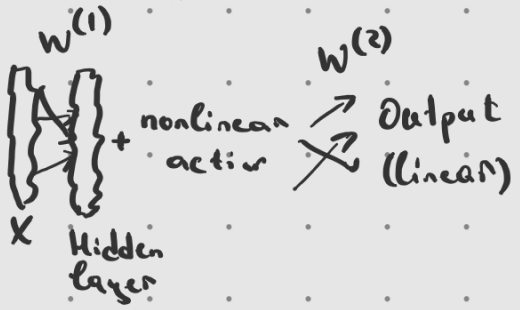
Loss $L_R(w) = -\ln p(Y|X,w) - \ln p(w) = L(w) + R(w)$

Prior is Gaussian $p(w) = \mathcal{N}(w; 0, \sigma^2 I)$

(1.1) $R(w) = -\ln \frac{1}{(2\pi)^{d/2}} e^{-\frac{\sigma^2}{2}\|w\|^2} = \underbrace{\frac{d}{2}\ln(2\pi)}_{c} + \underbrace{\frac{\sigma^2}{2}}_{\lambda>0}\|w\|^2 = C + \lambda w^T w$

(1.2) Effect of $R(w)$ = regularization. Prevents $w$ from being to big -> controls the model complexity.
From Bayesian point of view it's a prior assumption that $\|w\|$ is of order $\sigma$.

(2)




(2.1) Scale inputs $x' = ax$ -> outputs for the first layer $w'^{(1)} = \frac{1}{a} w^{(1)}$ -> $R(w) = \lambda w^{(1)T} w^{(1)} + \lambda w^{(2)T} w^{(2)}$ ~>
$w^{(2)}$ doesn't change ~> $R(w') = \lambda \cdot \frac{1}{a^2}\|w^{(1)}\|^2 + \lambda\|w^{(2)}\|^2$
Same situation when scaling outputs

(2.2) $p(w) = p(w^{(1)})\,p(w^{(2)})$ $\quad p(w^{(i)}) = \mathcal{N}(w^{(i)}, 0, \sigma_i^2 I)$

$$R(W) = -\ln p(w^{(1)})\,p(w^{(2)}) = \underbrace{\frac{d_1+d_2}{2}\ln(2\pi)}_{\text{const}} + \underbrace{\frac{\sigma^2}{2}}_{\lambda_1}\|w^{(1)}\|^2 + \underbrace{\frac{\sigma^2}{2}}_{\lambda_2}\|w^{(2)}\|^2$$

**(2.3)** same problem as in (2.1).

$w = (w_1, \dots, w_d)$

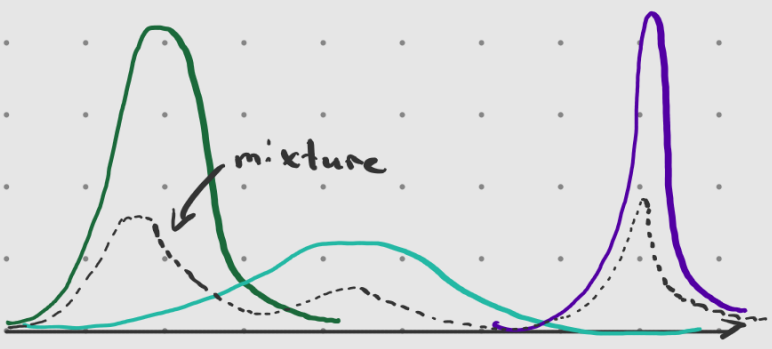
**(3)** Gaussian mixture $p(w) = \prod_i p(w_i)$ $\quad w_i = \sum_{j=1}^{m} \pi_j \mathcal{N}(w_i; \mu_j, \sigma_j^2)$ ← 1D-Gaussian

$\sum \pi_j = 1 \quad \pi_j > 0$

$$R(W) = -\sum_i \ln\left(\sum_{j=1}^{m} \pi_j \mathcal{N}(w_i; \mu_j, \sigma_j^2)\right)$$

$$L_R(W) = \underbrace{L(W)}_{\text{classical loss}} + \lambda R(W), \quad \lambda \in \mathbb{R}^+$$

**(3.1)**




$\pi_i \approx \frac{1}{3}$

**(3.2)** $\mu_j, \sigma_j^2, \pi_j^2$ fixed $\quad L_R(W) \to \min_W$

Effect of the constraint on the weights?

$$L_R(W) = L(W) + \underbrace{\lambda}_{\neq 0} R(W) \to \min_W$$

can be considered as a conditional minimization of $L(W)$ under constraints $R(W) \leq 0$ (in our case $\sum_i \ln\left(\sum \pi_j \mathcal{N}(w_i; \mu_j, \sigma_j^2)\right) \geq C$)

The value of the constraint is greater when all the weights are close to one of the mixture centers $\mu_i$.

**(3.3.)** Training concerns all the parameters.

$$p(j|w) = \gamma_j(w) = \frac{\pi_j \mathcal{N}(w; \mu_j, \sigma_j^2)}{\sum_{k=1}^{m} \pi_k \mathcal{N}(w; \mu_k, \sigma_k^2)}$$

$w$ is scalar!

(3.3.1) $\frac{\partial R}{\partial w_i} = -\partial_{w_i} \ln \sum_j \pi_j \mathcal{N}(w_i; \mu_j, \sigma_j^2) = -\frac{\sum_j \pi_j \partial_{w_i} \mathcal{N}(w_i; \mu_j, \sigma_j^2)}{\sum_j \pi_j \mathcal{N}(w_i; \mu_j, \sigma_j^2)} =$

$= \left\{ \begin{array}{l} p = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \\ p' = -\frac{(x-\mu)}{\sigma^2} \cdot p \end{array} \right\} = \sum_j \frac{(w_i - \mu_j)}{\sigma_j^2} \underbrace{\gamma_j(w_i)}_{>0} \rightarrow$

$\rightarrow w_i$ tends to approach to $\mu_j$ with greater posterior proba $\gamma_j(w_i)$ and lower variance $\sigma_j^2$

(3.3.2) $\frac{\partial R}{\partial \mu_j} = -\sum_i \partial_{\mu_j} \ln\left(\sum_k \pi_k \mathcal{N}(w_i; \mu_k, \sigma_k^2)\right) =$

$= -\sum_i \frac{\pi_j \partial_{\mu_j} \mathcal{N}(w_i; \mu_j, \sigma_j^2)}{\sum_k \pi_k \mathcal{N}(w_i; \mu_k, \sigma_k^2)} = \sum_i \frac{(\mu_j - w_i)}{\sigma_j^2} \gamma_j(w_i) \rightarrow$

$\rightarrow \mu_j$ will approach to the weights $w_i$ such that $j^{th}$ component of mixture is probable for them.

(3.3.3) $\frac{\partial R}{\partial \sigma_j} = -\sum_i \frac{\pi_j \partial_{\sigma_j} \mathcal{N}(w_i; \mu_j, \sigma_j^2)}{\sum_k \pi_k \mathcal{N}(w_i; \mu_k, \sigma_k^2)} \textcircled{=}$

$-\left(\frac{x-\mu}{\sigma}\right)\varphi\left(\frac{x-\mu}{\sigma}\right)$

$\frac{d}{d\sigma}\left(\underbrace{\frac{1}{\sigma}\varphi\left(\frac{x-\mu}{\sigma}\right)}_{p}\right) = -\frac{1}{\sigma^2}\varphi\left(\frac{x-\mu}{\sigma}\right) + \frac{1}{\sigma}\overbrace{\varphi'\left(\frac{x-\mu}{\sigma}\right)}^{''} \cdot \left(-\frac{x-\mu}{\sigma^2}\right) =$

$= -\frac{1}{\sigma} p + \frac{(x-\mu)^2}{\sigma^3} p = \frac{p}{\sigma}\left(\frac{(x-\mu)^2}{\sigma^2} - 1\right)$

$\textcircled{=} \sum_i \frac{1}{\sigma_j}\left(1 - \left(\frac{w_i - \mu_j}{\sigma_j}\right)^2\right) \gamma_j(w_i)$

$\sigma_j$ will decrease if there are many points very close to $\mu_j$.

$\sigma_j$ should be positive.

Possible solutions:

- Set $\sigma_{min} > 0$ and at each step k take $\sigma_k \vee \sigma_{min}$ in order to keep it positive
- Transform $\sigma_j = e^{\rho_j}$. Optmization w.r.t. $\rho_j$

# Convolutional NN (2021)



Image $X \in \mathbb{R}^{3\times3}$, $W \in \mathbb{R}^{2\times2}$ convolution filter, $Z \in \mathbb{R}^{2\times2}$ transformed image.

$$X = \begin{pmatrix} x_1 & x_2 & x_3 \\ x_4 & x_5 & x_6 \\ x_7 & x_8 & x_9 \end{pmatrix} \quad W = \begin{pmatrix} w_1 & w_2 \\ w_3 & w_4 \end{pmatrix} \quad Z = \begin{pmatrix} z_1 & z_2 \\ z_3 & z_4 \end{pmatrix}$$

(1)
$$z_1 = w_1x_1 + w_2x_2 + w_3x_4 + w_4x_5$$
$$z_2 = w_1x_2 + w_2x_3 + w_3x_5 + w_4x_6$$
$$z_3 = w_1x_4 + w_2x_5 + w_3x_7 + w_4x_8$$
$$z_4 = w_1x_5 + w_2x_6 + w_3x_8 + w_4x_9$$

(2)
$$z = \begin{pmatrix} z_1 \\ z_2 \\ z_3 \\ z_4 \end{pmatrix} = \text{Flat}(Z) \quad x = \text{Flat}(X), \quad w = \text{Flat}(W) \quad z = wx$$

$$\begin{pmatrix} z_1 \\ z_2 \\ z_3 \\ z_4 \end{pmatrix} = \underbrace{\begin{pmatrix} w_1 & w_2 & 0 & w_3 & w_4 & 0 & 0 & 0 & 0 \\ 0 & w_1 & w_2 & 0 & w_3 & w_4 & 0 & 0 & 0 \\ 0 & 0 & 0 & w_1 & w_2 & 0 & w_3 & w_4 & 0 \\ 0 & 0 & 0 & 0 & w_1 & w_2 & 0 & w_3 & w_4 \end{pmatrix}}_{W^T} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_9 \end{pmatrix}$$

(3) $y = W^Tz$ transposed convolution

$$W^Tz = \begin{pmatrix} w_1z_1 \\ w_2z_1 + w_1z_2 \\ w_2z_2 \\ w_3z_1 + w_1z_3 \\ w_4z_1 + w_3z_2 + w_2z_3 + w_1z_4 \\ w_4z_2 + w_2z_4 \\ w_3z_3 \\ w_4z_3 + w_3z_4 \\ w_4z_4 \end{pmatrix}$$


(4) How it helps? To build a decoder and increase the shape from 2x2 to 3x3.

(5) Padding Z, Stride = 1

$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & z_1 & z_2 & 0 \\ 0 & z_3 & z_4 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} * \underset{W'}{\begin{bmatrix} w_4 & w_3 \\ w_2 & w_1 \end{bmatrix}} = \begin{bmatrix} y_1 & y_2 & y_3 \\ y_4 & y_5 & y_6 \\ y_7 & y_8 & y_9 \end{bmatrix}$$

So, transposed convolution in (3) = Padding + Convolution with reversed $w' = (w_4, w_3, w_2, w_1)$

(6) Stride = 2

$$\begin{bmatrix} x_1 & x_2 & x_3 & x_4 \\ x_5 & x_6 & x_7 & x_8 \\ x_9 & x_{10} & x_{11} & x_{12} \\ x_{13} & x_{14} & x_{15} & x_{16} \end{bmatrix} * \begin{bmatrix} w_1 & w_2 \\ w_3 & w_4 \end{bmatrix} = \begin{bmatrix} z_1 & z_2 \\ z_3 & z_4 \end{bmatrix}$$

(6.1)

$$z = \underbrace{\begin{pmatrix} w_1 & w_2 & 0 & 0 & w_3 & w_4 & 0 & 0 \\ 0 & 0 & w_1 & w_2 & 0 & 0 & w_3 & w_4 \\ \cdots & & & & & & & \cdots \\ w_1 & w_2 & 0 & 0 & w_3 & w_4 & 0 & 0 \\ 0 & 0 & w_1 & w_2 & 0 & 0 & w_3 & w_4 \end{pmatrix}}_{W} \cdot x$$

(6.2) Decoder

$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & z_1 & z_2 & 0 \\ 0 & z_3 & z_4 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} * \begin{bmatrix} w_1 & w_2 \\ w_3 & w_4 \end{bmatrix} = \begin{bmatrix} \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \end{bmatrix}$$

## NN & conditional density mixture

$$\mathcal{D} = (x_i, y_i)_{i=1}^{N} \quad x_i \in \mathbb{R}^n, \; y_i \in \mathbb{R}$$

$$X \in \mathbb{R}^{n \times N} \qquad X = \begin{pmatrix} | & | & & | \\ x_1 & x_2 & \cdots & x_N \\ | & | & & | \end{pmatrix} \qquad Y = \begin{pmatrix} y_1 \\ \vdots \\ y_N \end{pmatrix}$$

Multimodal $p(y|x) = \sum_{k=1}^{K} \pi_k(x) p_k(y|x)$, $p_k(y|x) = \mathcal{N}(y | \mu_k(x), \sigma_k^2(x))$

$\pi_k(x) \in [0,1]$ $\sum \pi_k(x) = 1$.

$\mu_k(\cdot), \sigma_k^2(\cdot), \pi_k(\cdot)$ will be computed by a NN

$$\pi_k(x) = \frac{e^{a_k^\pi}}{\sum_c e^{a_c^\pi}} \qquad \sigma_k(x) = e^{a_k^\sigma} > 0 \qquad \mu_k(x) = a_k^\mu$$

(1) How can be implemented?

$x \to (w^{(1)}) \to \dots \to (w^{(n)}) \to \begin{pmatrix} a_k^\pi \\ a_k^\sigma \\ a_k^\mu \end{pmatrix} \to \begin{pmatrix} \pi_k \\ \sigma_k \\ \mu_k \end{pmatrix}$ ← Loss can be estimated via likelihood of the data

+ activations

(2) $L(w) = \sum_{i=1}^{N} L_i(w)$ $\qquad L_i(w) = \log\left( \sum_{k=1}^{K} \underbrace{\pi_k(x_i)}_{\pi_k} \underbrace{p_k(y_i|x_i)}_{p_k} \right)$

Derive $\partial_{a_j^\alpha} L_i(w)$ for $\alpha \in \{\pi, \sigma, \mu\}$

$$\partial_{a_j^\pi} L_i(w) = \frac{1}{\sum_{k=1}^{K} \pi_k(x_i) p_k(y_i|x_i)} \cdot \sum_k p_k \left( \underbrace{\frac{-e^{a_k^\pi}}{(\sum_c e^{a_c^\pi})^2} \cdot e^{a_j^\pi}}_{-\pi_k \cdot \pi_j} + \underbrace{\frac{e^{a_j^\pi}}{\sum_c e^{a_c^\pi}}}_{\pi_j} \delta_{kj} \right)$$

$$= \frac{\sum_k p_k (\delta_{kj} \cdot \pi_j - \pi_k \pi_j)}{\sum_k \pi_k p_k}$$

$$\partial_{a_j^\sigma} L_i(w) = \frac{1}{\sum_{k=1}^{K} \pi_k p_k} \pi_j \cdot \partial_\sigma p_k \cdot e^{a_j^\sigma}$$

cf. les exo de 2022

$$\partial_{a_j^\mu} L_i(w) = \frac{1}{\sum_{k=1}^{K} \pi_k p_k} \cdot \pi_j \underbrace{\partial_\mu p_k}_{\tau}$$

(3) Conditional mean $\mathbb{E}[y|x] = ?$ conditional variance $S^2(x) = ?$

$$\mathbb{E}[y|x] = \sum \pi_k(x) \mathbb{E}\mathcal{N}(\mu_k(x), \sigma_k^2(x)) = \sum \pi_k(x) \mu_k(x)$$

$$\sigma^2(x) = \mathbb{E}\left[(y - \mathbb{E}[y|x])^2 \,|\, x\right] = \sum_k \pi_k(x)\, \mathbb{E}\left[(\xi_k - \mathbb{E}[y|x])^2\right] = \quad \xi_k \sim \mathcal{N}(\mu_k(x), \sigma_k^2(x))$$

$$= \sum_k \pi_k(x) \Big( \underbrace{\mathbb{E}\left[(\xi_k - \mu_k(x))^2\right]}_{\sigma_k^2(x)} + \left(\mu_k(x) - \mathbb{E}[y|x]\right)^2 \Big) = \sum_k \pi_k(x) \left(\sigma_k^2(x) + (\mu_k(x) - \mathbb{E}[y|x])^2\right)$$