Assignment 3

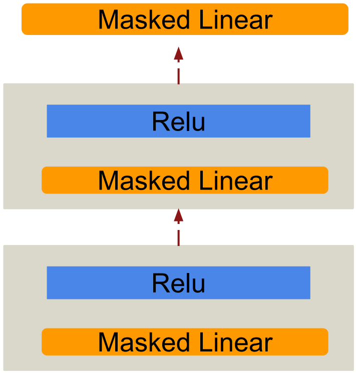
XCS236
Richard Wang

Stanford University

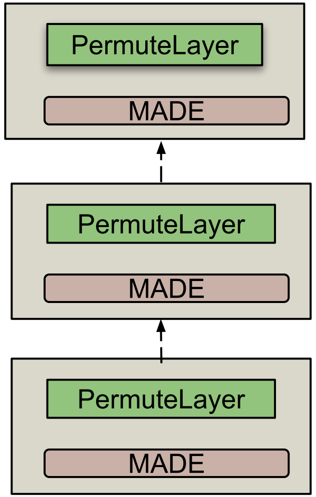
Problem 1: MAF (overview)

- **Learn complex density and sampling with multiple transformations MAF (two parts)**
  - **MADE** enforces Gaussian Autoregressive Model
  - **PermuteLayer** makes the model more expressive.

MADE (enforcing autoregressive learning)



MAF (combining multiple flows)



Problem 1: MADE and MAF in the code

MADE:

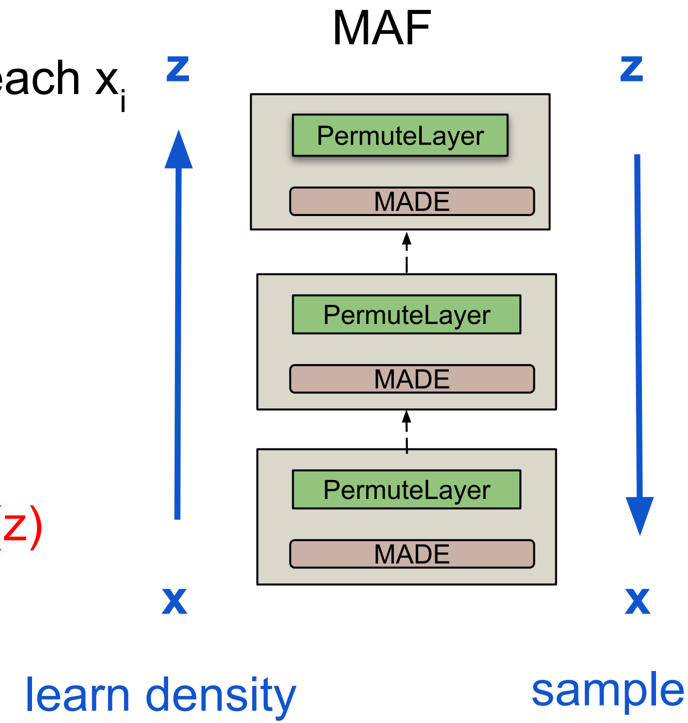
- self.net(**x**) -> Conditional mean/log SD for each x_i
- forward(z) -> x, log_det for sampling
- inverse(x) -> z, log_det for learning density

$$p(x_i \mid \boldsymbol{x}_{<i}) = \mathcal{N}\left(x_i \mid \mu_i, (\exp(\alpha_i))^2\right)$$

MAF:

self.nf: layers of flow

log_probs(x): accumulation of log_det and logp(z)



Problem 1 (a) Implement forward(z) in MADE

- How to compute conditional mean/log SD of each x_i without given x_i?

Approach: Sample x_i sequentially

$$p(x_i \mid \boldsymbol{x}_{<i}) = \mathcal{N}\left(x_i \mid \mu_i, (\exp(\alpha_i))^2\right) \qquad p(\boldsymbol{x}) = \prod_{i=1}^{n} p(x_i \mid \boldsymbol{x}_{<i})$$

1. x_1 : compute μ_1 and α_1 with *any x*: self.net(x) = (μ_1,?, α_1, ?) because μ_1 and α_1 don't depend on x and sample x_1 (using μ_1, α_1 and z_1)
2. x_2 : since μ_2 and α_2 only depend on x_1, input x = (x_1, any_number) for μ_2 and α_2 and sample x_2 (using μ_2, α_2, z_2)

- Implementation: To get (mu, alpha), split the output = self.net(x) in last dimension
- **Note**: forward(z) is for sampling; the returned log_det is not used in the main program

Problem 1 (b) Implement inverse(x) in MADE

- Returns z, log_det
- **Mean and log SD**: Obtained directly using `self.net(x)`.
- **Compute z:** Shift x by mean and scale it by SD to get x
- **log_det**: directly from log SD =(α_2,α_2).

Note: Inverse(x) is useful for computing density log p(x):

$$\log p(x) = \log p(z) + \log|\det(\partial f^{-1}/\partial x)|$$

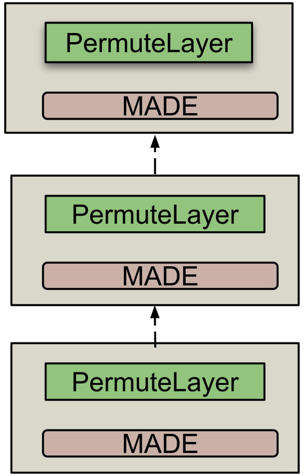
$$\log \left| \det \left(\frac{\partial f^{-1}}{\partial \boldsymbol{x}} \right) \right| = -\sum_{i=1}^{n} \alpha_i$$

standard deviation for $x_i = \exp(\alpha_i)$

Problem 1 (c) log p(x) in MAF

- **Compute logp(x)**
  - Iterate through flows in `self.nf`.
  - Apply `inverse` to get `log_det` for each flow
  - Sum log_det of all flows.
  - Add logp(z) (from last layer) to the sum.

$$\log p(\boldsymbol{x}) = \log p_z\left(f^{-1}(\boldsymbol{x})\right) + \sum_{j=1}^{k} \log \left| \det \left(\frac{\partial f_j^{-1}(\boldsymbol{x}_j)}{\partial \boldsymbol{x}_j} \right) \right|$$



self.nf

Reshape or flatten the output of PermuteLayer might be useful

Problem 2: GAN (Generative Adversarial Networks)

Minimax loss

$$D_\phi(\boldsymbol{x}) = \sigma(h_\phi(\boldsymbol{x}))$$

$$L_D(\phi;\theta) = -\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D_\phi(\boldsymbol{x})] - \mathbb{E}_{\boldsymbol{z}\sim\mathcal{N}(0,I)}[\log(1 - D_\phi(G_\theta(\boldsymbol{z})))]$$

$$L_G^{\text{minimax}}(\theta;\phi) = \mathbb{E}_{\boldsymbol{z}\sim\mathcal{N}(0,I)}[\log(1 - D_\phi(G_\theta(\boldsymbol{z})))]$$

- D_ϕ (x) : compute the *probability* that x is a real sample

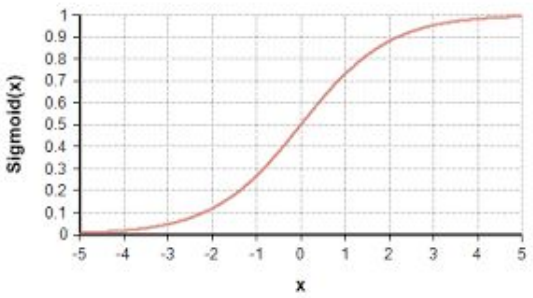
Problem 2 (a): Vanishing Gradient Problem in the generator

Show that if $D(G_\theta(z))\approx 0$, the gradient of minimax L_G with respect to θ will be approximately 0

- If the discriminator is almost certain that the generated example is fake, then the generator's gradient will be close to 0.

$$\frac{\partial L_G^{\text{minimax}}}{\partial \theta} = \mathbb{E}_{\boldsymbol{z}\sim\mathcal{N}(0,I)}\left[-\frac{\sigma'(h_\phi(G_\theta(\boldsymbol{z})))}{1-\sigma(h_\phi(G_\theta(\boldsymbol{z})))}\frac{\partial}{\partial\theta}h_\phi\left(G_\theta\left(\boldsymbol{z}\right)\right)\right] =$$

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



Problem 2 (b): Modifying Generator Loss

- **Defining Non-Saturating Loss**: Encourages stronger gradient signals for generator learning

$$L_G^{\text{non-saturating}}(\theta;\phi) = -E_{z\sim\mathcal{N}(0,I)}[\log D_\phi(G_\theta(z))]$$

- Note: The gradient of non-saturating L_G has a different denominator compared to minimax L_G

Implementation for L_G and L_D:
- F.binary_cross_entropy_with_logits

or
- F.logsigmoid and $1 - \sigma(x) = \sigma(-x)$

Problem 3: Divergence minimization

- Understand theoretic properties for GAN

$$L_D(\phi;\theta) = -\mathbb{E}_{x\sim p_{\text{data}}(x)}\left[\log D_\phi(x)\right] - \mathbb{E}_{x\sim p_\theta(x)}\left[\log\left(1 - D_\phi\left(x\right)\right)\right]$$
$$= \int p_{\text{data}}(x)\log D_\phi(x) - p_\theta(x)\log\left(1 - D_\phi\left(x\right)\right)dx$$

- Denote fake data distribution by $p_\theta(x)$
- In practice, $p_\theta(x)$ is unknown

Problem 3 (a): Optimal Discriminator

(a) **[4 points (Written)]** Show that L_D is minimized when $D_\phi = D^*$, where

$$D^*(\boldsymbol{x}) = \frac{p_{\text{data}}(\boldsymbol{x})}{p_\theta(\boldsymbol{x}) + p_{\text{data}}(\boldsymbol{x})}$$

Hint: for a fixed $\boldsymbol{x}$,what t minimizes $f(t) = -p_{\text{data}}(\boldsymbol{x}) \log t - p_\theta(\boldsymbol{x}) \log(1-t)$?

Given x, the minimizer t^* will satisfies the following:

$$-p_{data}(x)\log t^*(x) - p_\theta(x)\log(1 - t^*(x)) \le -p_{data}(x)\log D(x) - p_\theta(x)\log(1 - D(x))$$

Integrate both sides: Leads to t*(x) minimizing L_D.

Note: if $p_\theta(x) = p_{data}(x)$, then D*(x) = 0.5

Problem 3 (b): Optimal Discriminator Logits

(b) **[3 points (Written)]** Recall that $D_\phi(\boldsymbol{x}) = \sigma(h_\phi(\boldsymbol{x}))$. Show that the logits $h_\phi(\boldsymbol{x})$ of the discriminator estimate the log of the likelihood ratio of $\boldsymbol{x}$ under the true distribution compared to the model's distribution; that is, show that if $D_\phi = D^*$, then

$$h_\phi(\boldsymbol{x}) = \log \frac{p_{\text{data}}(\boldsymbol{x})}{p_\theta(\boldsymbol{x})} \tag{14}$$

To help you get started, note that

$$D_\phi(\boldsymbol{x}) = \sigma(h_\phi(\boldsymbol{x})) = \frac{1}{1+e^{-h_\phi(\boldsymbol{x})}}$$

Setting this to the expression for $D^*(x)$ in part 3a solution, we find that

---

Solve h_ϕ by replacing D with D^* in the hint

$h_\phi = \sigma^{-1}(D^*)$

Problem 3 (c) Generator loss as KL Divergence

(c) **[3 points (Written)]** Consider a generator loss defined by the sum of the minimax loss and the non-saturating loss,

$$L_G(\theta; \phi) = \mathbb{E}_{\boldsymbol{x} \sim p_\theta(\boldsymbol{x})}[\log(1 - D_\phi(\boldsymbol{x}))] - \mathbb{E}_{\boldsymbol{x} \sim p_\theta(\boldsymbol{x})}[\log D_\phi(\boldsymbol{x})] \tag{15}$$

Show that if $D_\phi = D^*$, then

$$L_G(\theta; \phi) = \mathrm{KL}(p_\theta(\boldsymbol{x}) \mid\mid p_{\text{data}}(\boldsymbol{x})) \tag{16}$$

To get started

$$\begin{aligned} L_G(\theta; \phi) &= \mathbb{E}_{p_\theta(\boldsymbol{x})}[\log(1 - D_\phi(\boldsymbol{x}))] - \mathbb{E}_{p_\theta(\boldsymbol{x})}[\log D_\phi(\boldsymbol{x})] \\ &= \mathbb{E}_{p_\theta(\boldsymbol{x})}\left[\log \frac{1 - D_\phi(\boldsymbol{x})}{D_\phi(\boldsymbol{x})}\right] \end{aligned}$$

- **Theoretically**, optimizing GAN is equivalent to minimizing the KL divergence between the fake distribution and real data distribution.

- **Note**: D^* depends on the generator's distribution $p_\theta(x)$.

Problem 3 (d)

(d) **[3 points (Written)]** Recall that when training VAEs, we minimize the negative ELBO, an upper bound to the negative log likelihood. Show that the negative log likelihood, $-\mathbb{E}_{\boldsymbol{x} \sim p_{\text{data}}(\boldsymbol{x})}[\log p_\theta(\boldsymbol{x})]$, can be written as a KL divergence plus an additional term that is constant with respect to $\boldsymbol{\theta}$. We are asking if the KL divergence is equal to L_G, so after finding the expression, you will be able to deduce that. Note that the constant term is constant with respect to θ, so it can be another expectation.

Does this mean that a VAE decoder trained with ELBO and a GAN generator trained with the L_G defined in the previous part 3c are implicitly learning the same objective? Explain.

Show that $-E_{X \sim p\ \text{data}(x)} \log p_\theta(x)$ = KL + a term independent of θ

Hints:

- No need for a latent variable z in this case
- Try adding a term to make the negative log likelihood look like a KL

Problem 4 Conditional GAN (CGAN)

Goal: Learn conditional generation $p_\theta(\boldsymbol{x} \mid y)$

- Data includes images x and the corresponding labels y:
- labels are uniformly distributed

Discriminator Loss:

$$L_D(\phi;\theta) = -\mathbb{E}_{(\boldsymbol{x},y)\sim p_{\text{data}}(\boldsymbol{x},y)}[\log D_\phi(\boldsymbol{x},y)] - \mathbb{E}_{(\boldsymbol{x},y)\sim p_\theta(\boldsymbol{x},y)}[\log(1 - D_\phi(\boldsymbol{x},y))]$$

$$= -\mathbb{E}_{(\boldsymbol{x},y)\sim p_{\text{data}}(\boldsymbol{x},y)}[\log D_\phi(\boldsymbol{x},y)] - \mathbb{E}_{y\sim p_\theta(y)}[\mathbb{E}_{\boldsymbol{z}\sim\mathcal{N}(0,I)}[\log(1 - D_\phi(G_\theta(\boldsymbol{z},y),y))]]$$

- Model join distribution for the generator:

$$p_\theta(\boldsymbol{x},y) = p_\theta(\boldsymbol{x} \mid y)p_\theta(y)$$

- Conditional generation p_θ(x|y) depends on both y and latent variable z

$$G_\theta(\boldsymbol{z},y), \text{ where } \boldsymbol{z} \sim \mathcal{N}(0,I)$$

- Assume uniform label distribution

$$p_\theta(y) = \frac{1}{m}$$

Problem 4 (a) Optimal Discriminator Logits for CGAN

Assume $\varphi(\boldsymbol{x})$ transform data into a mixtures of *m* unit Gaussians

$$\frac{p_{\text{data}}(\boldsymbol{x} \mid y)}{p_\theta(\boldsymbol{x} \mid y)} = \frac{\mathcal{N}(\varphi(\boldsymbol{x}) \mid \boldsymbol{\mu}_{\boldsymbol{y}}, I)}{\mathcal{N}(\varphi(\boldsymbol{x}) \mid \hat{\boldsymbol{\mu}}_{\boldsymbol{y}}, I)}$$

Show $$h^*(\boldsymbol{x}, y) = \boldsymbol{y}^T(A\varphi(\boldsymbol{x}) + \boldsymbol{b})$$

The optimal discriminator logits have the same form as 3 (b), given in the hint. So we can compute the following:

$$h_\phi(x, y) = \log \frac{p_{\text{data}}(\boldsymbol{x}, y)}{p_\theta(\boldsymbol{x}, y)}$$
$$= \log \frac{p_{\text{data}}(\boldsymbol{x}|y)}{p_\theta(\boldsymbol{x}|y)} + \log \frac{p_{\text{data}}(y)}{p_\theta(y)}$$
$$= \log \frac{p_{\text{data}}(\boldsymbol{x}|y)}{p_\theta(\boldsymbol{x}|y)} =$$

Stack the result for a fixed class label y together for all y, then multiple the one-hot vector

Problem 4 (b) Implement CGAN

- Implement discriminator loss

$$L_D(\phi;\theta) = -\mathbb{E}_{(\boldsymbol{x},y)\sim p_{\text{data}}(\boldsymbol{x},y)}[\log D_\phi(\boldsymbol{x},y)] - \mathbb{E}_{(\boldsymbol{x},y)\sim p_\theta(\boldsymbol{x},y)}[\log(1-D_\phi(\boldsymbol{x},y))]$$

$$= -\mathbb{E}_{(\boldsymbol{x},y)\sim p_{\text{data}}(\boldsymbol{x},y)}[\log D_\phi(\boldsymbol{x},y)] - \mathbb{E}_{y\sim p_\theta(y)}[\mathbb{E}_{\boldsymbol{z}\sim\mathcal{N}(0,I)}[\log(1-D_\phi(G_\theta(\boldsymbol{z},y),y))]]$$

- Implement non-saturating generator loss (similar to 2 (b))

Implementation:

(x_{real}, y_{real}) for the real data term
(z, y_{real}) for the generated data term

Problem 5 Wasserstein GAN

- **Problem 3 (c):** shows that $L_G = \mathrm{KL}(p_\theta(x) \,//\, p_{data}(x))$ when using the optimal discriminator D*, given $p_\theta(x)$ and $p_{data}(x)$.
- **Learning GAN**: can be viewed as minimizing the KL divergence between $p_\theta(x)$ and $p_{data}(x)$
- **KL divergence issue:** $\mathrm{KL}(p_\theta(x) \,//\, p_{\text{data}}(x))$ and its gradient may become large when the distributions have minimal overlapping
- **Solution:** Implement alternative losses to address the issue.

Problem 5 (a) Divergence of two Normals with same var

(a) **[3 points (Written)]** Let $p_\theta(x) = \mathcal{N}(x \mid \theta, \epsilon^2)$ and $p_{\text{data}}(x) = \mathcal{N}(x \mid \theta_0, \epsilon^2)$ be normal distributions with standard deviation ϵ centered at $\theta \in \mathbb{R}$ and $\theta_0 \in \mathbb{R}$ respectively. Show that

$$\mathrm{KL}(p_\theta(x) \,||\, p_{\text{data}}(x)) = \frac{(\theta - \theta_0)^2}{2\epsilon^2} \tag{21}$$

To help you get started:

$$\mathsf{KL}(p_\theta(x) \,||\, p_{\text{data}}(x)) = \mathbb{E}_{x \sim \mathcal{N}(\theta, \epsilon^2)}\left[\log \frac{\exp(-\frac{1}{2\epsilon^2}(x-\theta)^2)}{\exp(-\frac{1}{2\epsilon^2}(x-\theta_0)^2)}\right] =$$

Hint: X ~p(x), E(X) = ∫ x·p(x)dx

Problem 5 (b) Asymptotic behavior of KL

Assume $\theta \neq \theta_0$, What happens to $KL(p_\theta(x) \,//\, p_{data}(x))$ and its derivatives as $\varepsilon \rightarrow 0$?

- Problem 3 c) shows $L_G(x) = KL(p_\theta(x) \,//\, p_{data}(x))$ when $D = D^*$
- **Issue:** in this case, what happens to gradient of L_G?

Problem 5 (c) Alternative objectives

Solution: Define new losses to avoid large gradient of L_G

$$L_D(\phi;\theta) = \mathbb{E}_{x\sim p_\theta(x)}[D_\phi(x)] - \mathbb{E}_{x\sim p_{\text{data}}(x)}[D_\phi(x)]$$

$$L_G(\theta;\phi) = -\mathbb{E}_{x\sim p_\theta(x)}[D_\phi(x)]$$

***$D_\phi(x) \in R$* is a real value function**, not a probability

Question: Consider limit of $\varepsilon \to 0$ i.e. $p_\theta(x=\theta)=1$ and $p_{data}(x=\theta_0) = 1$
Why is there no D_ϕ minimize L_D ?

$L_D = D_\phi(\theta) - D_\phi(\theta_0)$
If D_ϕ can be any real-valued function, you can find D_ϕ that make L_D arbitrarily small

Problem 5 (d) Impose smoothness constraint

$D_\phi(x) \in R$ is a real value function and $|D'_\phi(x)| \leq 1$

$$L_D(\phi;\theta) = \mathbb{E}_{x\sim p_\theta(x)}[D_\phi(x)] - \mathbb{E}_{x\sim p_{\mathrm{data}}(x)}[D_\phi(x)]$$

Question: Consider limit of $\varepsilon \to 0$ i.e. $p_\theta(x=\theta)=1$ and $p_{data}(x=\theta_0) = 1$

Do we have D_ϕ that minimize L_D ?

By Mean Value Theorem:

$|D_\phi(\theta) - D_\phi(\theta_0)| \leq |D'_\phi(\theta^*)||\theta-\theta_0| \leq |\theta-\theta_0|$

$-|\theta-\theta_0| \leq D_\phi(\theta) - D_\phi(\theta_0) \leq |\theta-\theta_0|$

Note: L_D is Wasserstein distance between $p_\theta(x)$ and $p_{data}(x)$

Problem 5 (e) implementation

$$L_D(\phi;\theta) = \mathbb{E}_{\boldsymbol{x}\sim p_\theta(\boldsymbol{x})}[D_\phi(\boldsymbol{x})] - \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[D_\phi(\boldsymbol{x})] + \lambda\mathbb{E}_{\boldsymbol{x}\sim r_\theta(\boldsymbol{x})}[(\|\nabla D_\phi(\boldsymbol{x})\|_2 - 1)^2]$$

$$L_G(\theta;\phi) = -\mathbb{E}_{\boldsymbol{x}\sim p_\theta(\boldsymbol{x})}[D_\phi(\boldsymbol{x})]$$

Enforce **$|\nabla_x D_\phi(x)| \leq 1$** using a regularization term

$X_1 \sim p_\theta(x)$, $x_2 \sim p_{data}(x)$ and $r_\theta(x) = \alpha x_1 + (1 - \alpha)x_2$, $\alpha \sim \text{Uniform}(0,1)$

- D_ϕ is not log prob
- The order of X_1 and X_2 matters for L_D in the gradescope test!
- Sum D over the batch for gradient computation to avoid loop

$$X = \begin{bmatrix} \boldsymbol{x}^{(1)} \\ \vdots \\ \boldsymbol{x}^{(m)} \end{bmatrix} \qquad \texttt{grad} = \frac{\partial}{\partial X}\sum_{j=1}^{m} D(\boldsymbol{x}^{(j)})$$

Reference: Improved Training of Wasserstein GANs