AR 2:

Autoregressive Quantile Networks for Generative Modeling (2018)

G. Ostrovski, W. Dabney, R. Munos

Presenter:
Dachun Sun (dsun18)

TABLE OF CONTENTS

INTRODUCTION

- Problem Statement
- Related Works
- Basics
  - Quantile Regression

Problem Statement

- Notations
  - Space $\mathcal{X}$
  - Random variable $X \in \mathcal{X}$
  - Distribution $p_X \in \mathcal{P}(\mathcal{X})$ and cumulative distribution function (CDF) F_X
  - Inverse CDF (Quantile function) $Q_X = F_X^{-1}$

- **Problem: Modeling** $p(x_1, \ldots, x_D)$

- Simplest Solution: Discretization into separate values, $x_1, \ldots, x_n \in \mathcal{X}$ and parameterize the approximate $p_\theta(x_i) \propto \exp(\theta_i)$.
  - Typically, the parameter are optimized to minimize the KL divergence $\theta^* = \operatorname{argmin}_\theta D_{KL}(p_X \| p_\theta)$

Related Works

Bayesian Networks and Autoregressive Models (ARs):

- Factorize the density as a product of conditional distributions.
- Let $X = (X_1, \ldots, X_n)$, then for any permutation of the dimensions $\sigma : \mathbb{N}_n \to \mathbb{N}_n$

$$p_X(x) = \prod_{i=1}^{n} p_{X_{\sigma(i)}} \left(x_{\sigma(i)} | x_{\sigma(1)}, \ldots, x_{\sigma(i-1)} \right)$$

- Examples: PixelRNN/PixelCNN, NADE, MADE, etc.
- Limitations:
  - Need powerful conditioning to avoid having to order the dimensions
  - Essentially use KL divergence as the loss function
  - Slow in generation

Related Works

Variational Autoencoders (VAEs):

- Represent the density as the marginalization over a latent random variable.
- Let $Z \in \mathcal{Z}$, then maximize the ELBO

$$\log p_\theta(x) \geq -D_{KL}\left(q_\theta(z|x)||p(z)\right) + \mathbb{E}\left[\log p_\theta(x|z)\right]$$

- Straightforward to implement and optimize; Effective at capturing the structure in high-d spaces. However, often misses fine-grained details, and also uses KL divergence.

Related Works

Generative Adversarial Networks (GANs):

- Pose the problem of learning the generative model as a two-player zero-sum game between a discriminator and a generator. The generator is an implicit latent variable model that reparameterized samples to $\mathcal{X}$.

$$\operatorname{argmin}_G \sup_D \left[\mathbb{E}_X \left(D(X)\right) + \mathbb{E}_Z \log\left(1 - D\left(G(Z)\right)\right)\right]$$

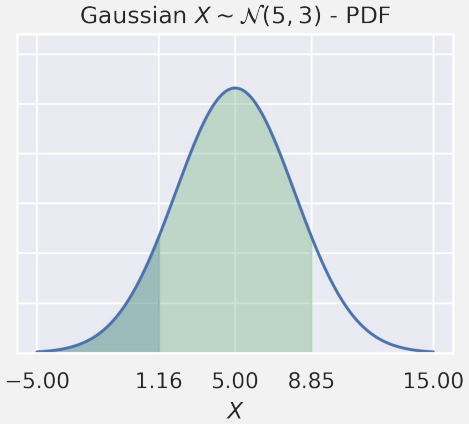
- Limitations
  - Cannot estimate the probability of a sample point
  - Essentially minimizing a lower-bound on Jensen-Shannon divergence (a function of KL divergence) $M = 0.5(P + Q) : JSD(P\|Q) = 0.5\left(D_{KL}(P\|M) + D_{KL}(Q\|M)\right)$

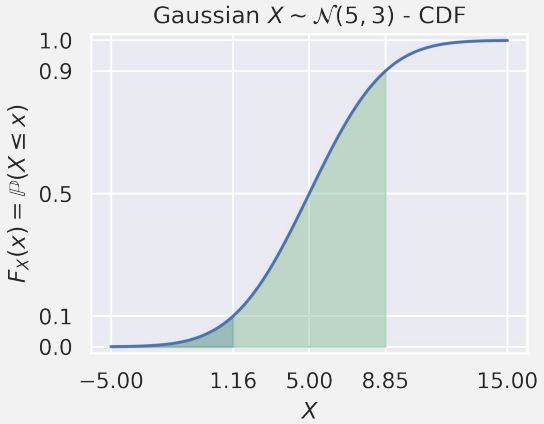
Basics - Quantile

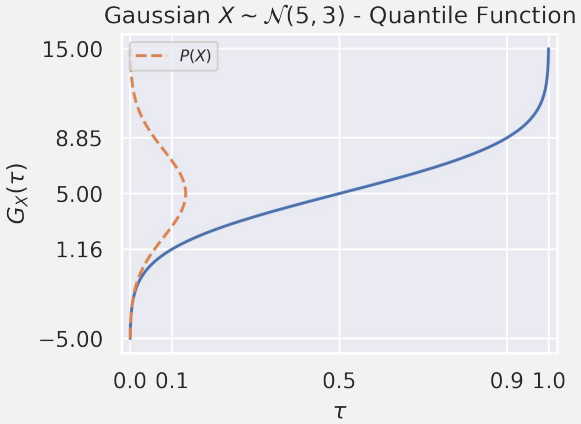
Let X be a r.v. random variable with CDF $F_X(x) = \mathbb{P}(X \leq x)$. The τ-th quantile of X is given by

$$Q_X(\tau) = F_X^{-1}(\tau) = \inf_x \{F_X(x) \geq \tau\}, \text{ where } \tau \in (0, 1)$$

Example: Let $X \sim \mathcal{N}(5, 3)$, $Q_X(0.1) \approx 1.155$, $Q_X(0.9) \approx 8.845$.







Basics - Quantile Regression

Given datasets (X, Y), and a quantile $\tau \in (0, 1)$, approximate the conditional quantile function at $\boldsymbol{\tau}$: $Q_{Y|X}(\tau) = X\beta_\tau$, with the quantile loss function:

$$\rho_\tau(u) = \begin{cases} (\tau - 1)u & u \leq 0 \\ \tau u & u > 0 \end{cases}$$

where the error $u = Y - X\beta_\tau$.

- The sign of the error term indicates the direction of correction.
  - **+** for underestimation, **-** for overestimation.
- Instead of the square of error used in most linear regression, the quantile loss is asymmetric.
  - For underestimation, **+**, the weight of penalty is $\boldsymbol{\tau}$.
  - For overestimation, **-**, the weight of penalty is ($\boldsymbol{\tau}$ - 1).

Basics - Quantile Regression

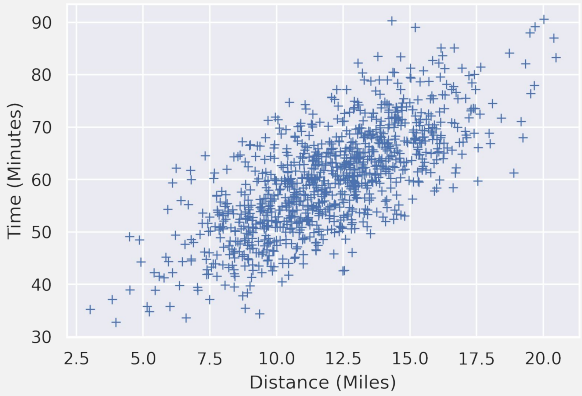
- Suppose we are trying to give an estimate of the delivery time of an order on UberEats.
- The function is obviously dependent on the distance of delivery.
- Try to make sure most users could get their meal in the time range.
  - **A Solution**: Give an estimate between times where 10% and 90% of people received their order.

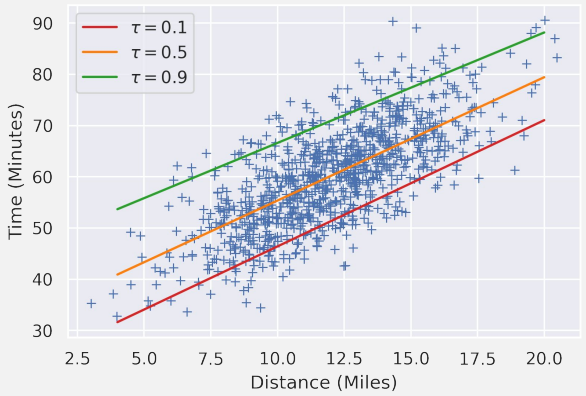
$$Q_{Y|X}(0.1) = X\beta_{0.1}$$

$$\rho_{0.1}(u) = \begin{cases} -0.9u & u \leq 0 \\ 0.1u & u > 0 \end{cases}$$

$$Q_{Y|X}(0.9) = X\beta_{0.9}$$

$$\rho_{0.9}(u) = \begin{cases} -0.1u & u \leq 0 \\ 0.9u & u > 0 \end{cases}$$





METHOD

- Autoregressive Quantiles
- Density Function
- PixelIQN

Modeling

- Let $X = (X_1, \ldots, X_n) \in \mathcal{X}_1 \times \cdots \times \mathcal{X}_n = \mathcal{X}$ be n-dim r.v.
- How do we extend the Quantile function?
  - If the CDFs all rely on the same τ, we need *comonotonic* property to ensure invertibility. Two r.v.s are comonotonic iff can be expressed as non-decreasing functions of a single r.v.
    - Assumption is too strong to be useful more broadly

$$F_X^{-1}(\tau) = \left(F_{X_1}^{-1}(\tau), F_{X_2}^{-1}(\tau), \ldots, F_{X_n}^{-1}(\tau)\right)$$

  - Use a separate value τ for each component.
    - Independence assumption too restrictive for many domains

$$F_X^{-1}(\vec{\tau}) = \left(F_{X_1}^{-1}(\tau_1), F_{X_2}^{-1}(\tau_2), \ldots, F_{X_n}^{-1}(\tau_n)\right)$$

Modeling

- Fixing an ordering of n dimensions, if the density function could be expressed as a product of conditional likelihoods, then the joint c.d.f. is:

$$F_X(x) = \mathbb{P}\left(X_1 \leq x_1, \ldots, X_n \leq x_n\right)$$
$$= \prod_{i=1}^{n} F_{X_i|X_{i-1},\ldots,X_1}(x_i)$$

- We could then write the quantile function as:

$$F_X^{-1}(\tau_{\text{joint}}) = \left(F_{X_1}^{-1}(\tau_1), \ldots, F_{X_n|X_{n-1},\ldots}^{-1}(\tau_n)\right)$$

- Denote $\mathcal{X}_{1:n} = \mathcal{X}_1 \times \cdots \times \mathcal{X}_i$, let $\tilde{\mathcal{X}} := \bigcup_{i=0}^{n} \mathcal{X}_{1:i}$ be the space of 'partial' data points. We can define the **autoregressive implicit quantiles** as a function.

$$Q_\theta : \tilde{\mathcal{X}} \times [0,1]^n \to \tilde{\mathcal{X}}$$

For generation, we can iteratively get the next partial component $x_{1:i} = Q_\theta(x_{1:i-1}, \tau_i)$.

Divergence and Metrics

- The expected quantile loss over the data distribution for a prediction q.

$$
\begin{aligned}
g_\tau(q) &= \mathbb{E}_{X\sim P}\left[\rho_\tau\left(X-q\right)\right] \\
&= \int_{-\infty}^{q}(x-q)(\tau-1)f_P(x)dx + \int_{q}^{\infty}(x-q)\tau f_P(x)dx \\
&= \int_{-\infty}^{q}(q-x)f_P(x)dx + \int_{-\infty}^{\infty}(x-q)\tau f_P(x)dx \\
&= q\int_{-\infty}^{q}f_P(x)dx - \int_{-\infty}^{q}xf_P(x)dx + \left(\mathbb{E}_{X\sim P}\left[X\right]-q\right)\tau \\
&= qF_P(q) - \left(\left[xF_P(x)\right]_{-\infty}^{q} - \int_{-\infty}^{q}F_P(x)dx\right) + \left(\mathbb{E}_{X\sim P}\left[X\right]-q\right)\tau \\
&= \int_{-\infty}^{q}F_P(x)dx + \left(\mathbb{E}_{X\sim P}\left[X\right]-q\right)\tau
\end{aligned}
$$

Divergence and Metrics

- The relative loss between q and ground truth quantile function

$$
\begin{aligned}
g_\tau(q) - g_\tau(F_P^{-1}(\tau)) &= \int_{F_P^{-1}(\tau)}^{q} F_P(x)dx + \left(F_P^{-1}(\tau) - q\right)\tau \\
&= \int_{F_P^{-1}(\tau)}^{q} \left(F_P(x) - \tau\right) dx
\end{aligned}
$$

Divergence and Metrics

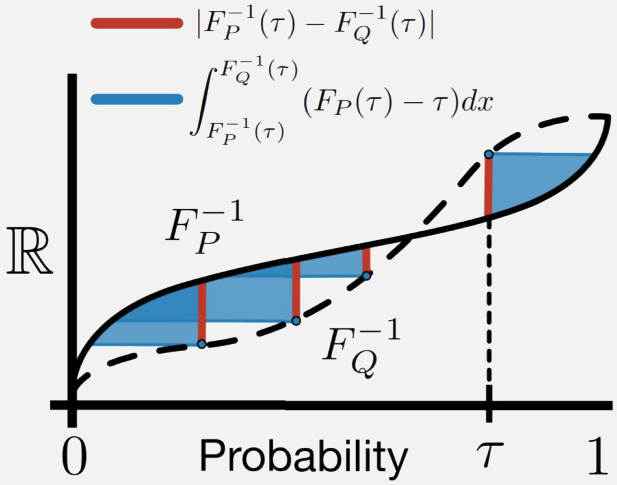
- Take the expectation over τ, we found that relative quantile loss between a quantile function Q and the quantile function of P leads us to a new divergence between P and Q.

$$\mathbb{E}_{\tau\sim\mathcal{U}([0,1])}\left[g_\tau\left(F_Q^{-1}(\tau)\right)-g_\tau\left(F_P^{-1}(\tau)\right)\right]=\int_0^1\left[\int_{F_P^{-1}(\tau)}^{F_Q^{-1}(\tau)}\left(F_P(x)-\tau\right)dx\right]d\tau$$

$$\mathbb{E}_{\tau\sim\mathcal{U}([0,1])}\left[g_\tau\left(F_Q^{-1}(\tau)\right)\right]=\underbrace{\int_0^1\left[\int_{F_P^{-1}(\tau)}^{F_Q^{-1}(\tau)}\left(F_P(x)-\tau\right)dx\right]d\tau}_{\text{Quantile divergence } q(P,Q)}+\underbrace{\mathbb{E}_{\tau\sim\mathcal{U}([0,1])}\left[g_\tau\left(F_P^{-1}(\tau)\right)\right]}_{\text{does not depend on } Q}$$

Divergence and Metrics

- If P and Q are univariate distributions,
  - Red line segment is the 1-Wasserstein metric



Divergence and Metrics

- Minimizing quantile loss indeed minimizes some divergence between distributions.
- Take the gradient of quantile loss w.r.t. parameters is an unbiased estimate of the gradient of the divergence.

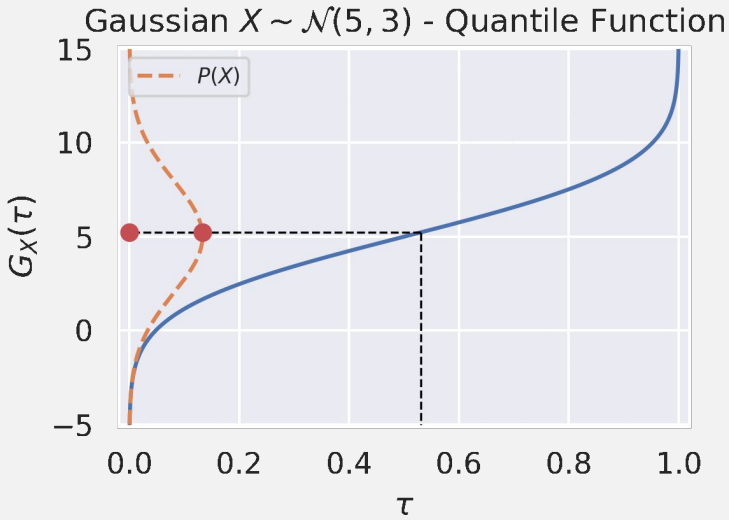
$$\begin{aligned}
&\nabla_\theta \mathbb{E}_{\tau\sim\mathcal{U}([0,1])}\left[g_\tau\left(\bar{Q}_\theta(\tau)\right)\right] \\
&\qquad = \mathbb{E}_{\tau\sim\mathcal{U}([0,1])}\mathbb{E}_{X\sim P}\left[\nabla_\theta \rho\left(X-\bar{Q}_\theta(\tau)\right)\right] \\
&\qquad = \nabla_\theta q\left(P,\bar{Q}_\theta\right)
\end{aligned}$$

- Use Huber quantile loss instead, because the gradient scales with the magnitude of the error.

$$\rho_\tau(u)=\begin{cases}(\tau-1)u & u\le 0\\ \tau u & u>0\end{cases} \qquad \rho_\tau^\kappa(u)=\begin{cases}\frac{|\tau-\mathbb{I}\{u\le 0\}|}{2\kappa}u^2 & |u|\le\kappa\\ |\tau-\mathbb{I}\{u\le 0\}|\left(|u|-\frac{\kappa}{2}\right) & \text{otherwise}\end{cases}$$

Reparameterization

- Previously, the source of randomness comes from $\epsilon \sim \mathcal{N}(0, 1)$.
- Now, we sample from Quantile function, the source of randomness comes from $\tau \sim \mathcal{U}([0, 1])$.



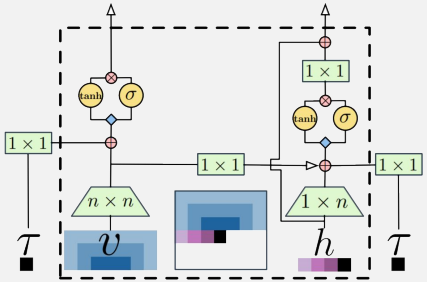
Querying the Density Function

- IQN does not directly model the log-likelihood of the data distribution
- But we can query the implied density at a point:

$$\frac{\partial}{\partial \tau} F_X^{-1}(\tau) = \frac{1}{p_X\left(F_X^{-1}(\tau)\right)}$$

- A single step of back-propagation calculates the above formula
- Getting general likelihoods is *inefficient* because it would require finding the value of $\boldsymbol{\tau}$ that produces the closest approximation to the query point.

PixelIQN



Gated PixelCNN

$$p(x|s) = \prod_{i=1}^{3n^2} p(x_i|x_1, \dots, x_{i-1}, s_i)$$

$$\sum_i D_{KL}(\delta_{x_i}, p(\cdot|x_1, \dots, x_{i-1}))$$

PixelIQN

$$\tau = (\tau_1, \dots, \tau_{3n^2}) \sim \mathcal{U}\left([0,1]^{3n^2}\right)$$

$$Q_x(\tau) \in \mathbb{R}^{3n^2}, Q_x(\tau)_i = Q_X(\tau_i|x_{i-1}, \dots)$$

$$\sum_i \rho_{\tau_i}^{\kappa}\left(x_i - Q_X(\tau_i|x_{i-1}, \dots)\right)$$

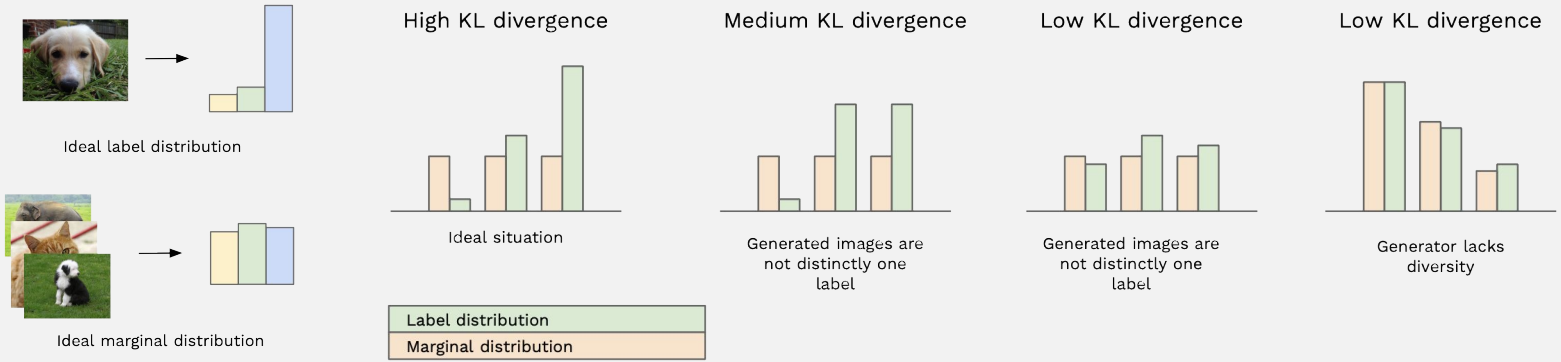
- The location-dependent conditioning was used to condition on class labels in Gated PixelCNN. Used to condition on $\boldsymbol{\tau}$ in PixelIQN.
- PixelIQN directly output 3 color channels without the final softmax activation in Gated PixelCNN.

RESULTS

- Dataset and Metrics
- Experiments

Datasets & Metrics

- Datasets: CIFAR-10 and ImageNet 32x32
- Metrics:
  - Fréchet inception distance (FID): Squared Wasserstein metric between two multidimensional Gaussian distributions. *Lower is better.*
  - Inception Score (IS): KL-divergence between similar label distribution and marginal distribution. *Higher is better.*



Credit to https://medium.com/octavian-ai/a-simple-explanation-of-the-inception-score-372dff6a8c7a

Experiments

Method	CIFAR-10		ImageNet (32x32)	
	Inception	FID	Inception	FID
WGAN	3.82	-	-	-
WGAN-GP	6.5	36.4	-	-
DC-GAN	6.4	37.11	7.89	-
PixelCNN	4.60	65.93	7.16	40.51
PixelIQN	5.29	49.46	8.68	26.56
PixelIQN(l)	-	-	7.29	37.62
PixelCNN*	-	-	8.33	33.27
PixelIQN*	-	-	10.18	22.99

Table 1. Inception score and FID for CIFAR-10 and ImageNet. WGAN and DC-GAN results taken from (Arjovsky et al., 2017; Radford et al., 2015). PixelIQN(l) is the small 15-layer version of the model. Models marked $*$ refer to class-conditional training.

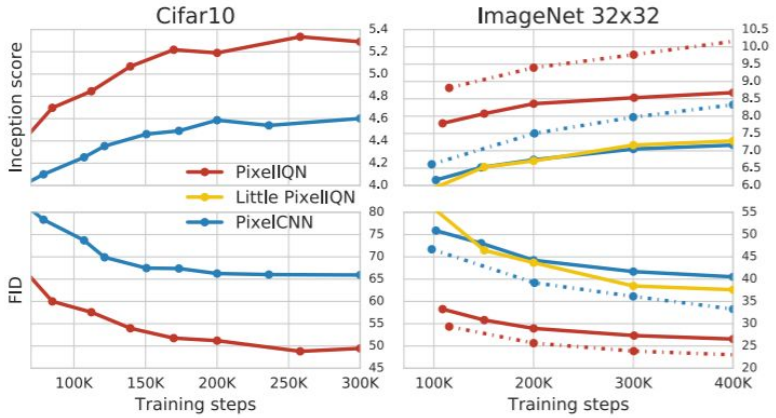


Figure 4. Evaluations by Inception score (higher is better) and FID (lower is better) on CIFAR-10 and ImageNet 32x32. Dotted lines correspond to models trained with class-label conditioning.

Experiments

Figure 5. ImageNet 32x32: Real example images (left), samples generated by PixelCNN (center), and samples generated by PixelIQN (right). Neither of the sampled image sets were cherry-picked. More samples by PixelIQN in the Appendix.

Experiments

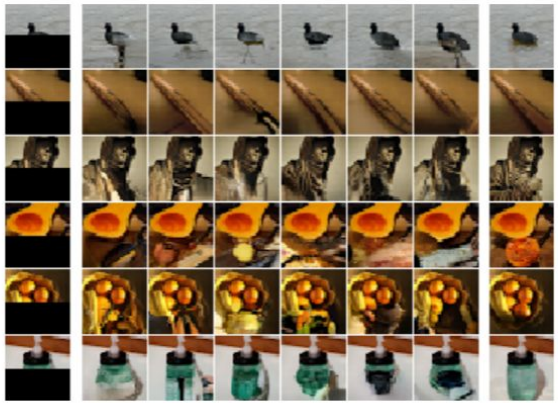


Figure 6. Small ImageNet inpainting examples. Left image is the input provided to the network at the beginning of sampling, right is the original image, columns in between show different completions. More examples in the Appendix.

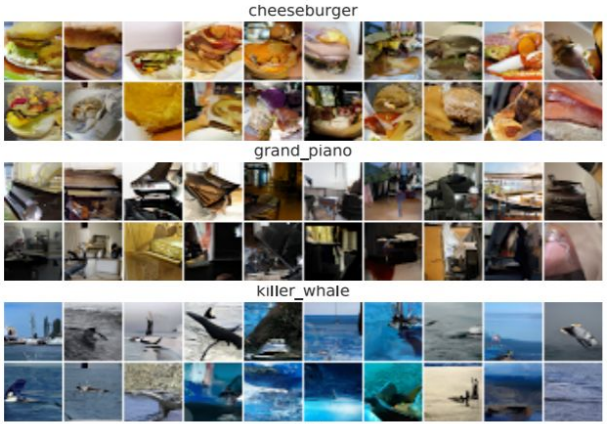


Figure 7. Class-conditional samples from PixelIQN. More samples of each class and more classes in the Appendix.

CONCLUSION

01

KL-Divergence

Most existing works use KL-divergence

02

Quantile Regression

Borrow the quantile loss, modeling CDFs, and derive a divergence on CDFs.

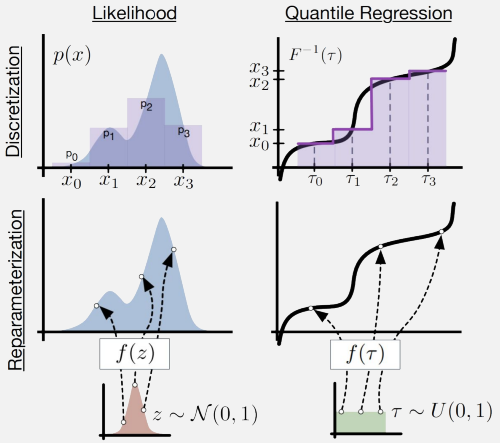
03

Application

Replace the reparameterization and merge with more existing models.

Summary

- Joint-quantile function are factorized as products of conditional quantile functions
- Replace KL-divergence with Quantile loss
- Reparameterization is now on $\tau \sim \mathcal{U}(0,1)$
- The technique can also be used in VAEs:
  - Let $e : \mathbb{R}^n \to \mathbb{R}^m$ and $d : \mathbb{R}^m \to \mathbb{R}^n$ be the encoder and decoder. Let Q_τ be an AIQN on the space $\mathbb{R}^m$.
  - Loss: $\mathcal{L}(x) = \mathcal{L}_{\text{VAE}}(x) + \mathbb{E}_{\tau \sim \mathcal{U}([0,1]^m)} \left[\rho_\tau^\kappa \left(e(x) - Q_\tau \right) \right]$
  - Decoding process: $y = d(Q_\tau)$
- Limitations:
  - Sampling is slow, similar to PixelRNN/PixelCNN
  - Querying density is possible, but not easy to find corresponding τ.



THANKS