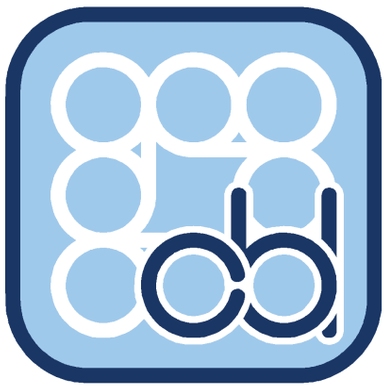
Evaluating Deep Generative Models on Out-of-Distribution Inputs

Eric Nalisnick

OxCSML Seminar

31.5.19



Computational and Biological Learning
University of Cambridge

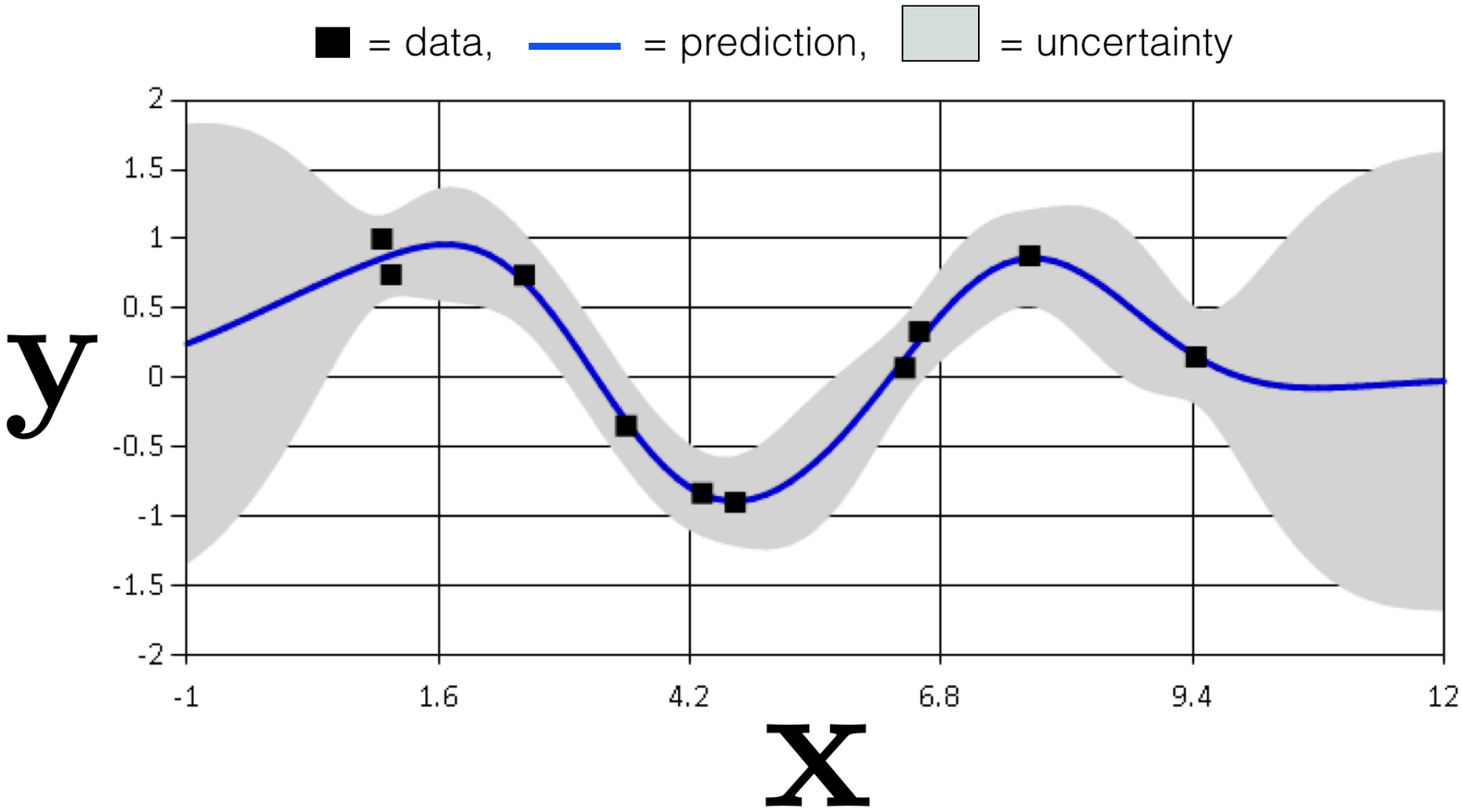
DeepMind

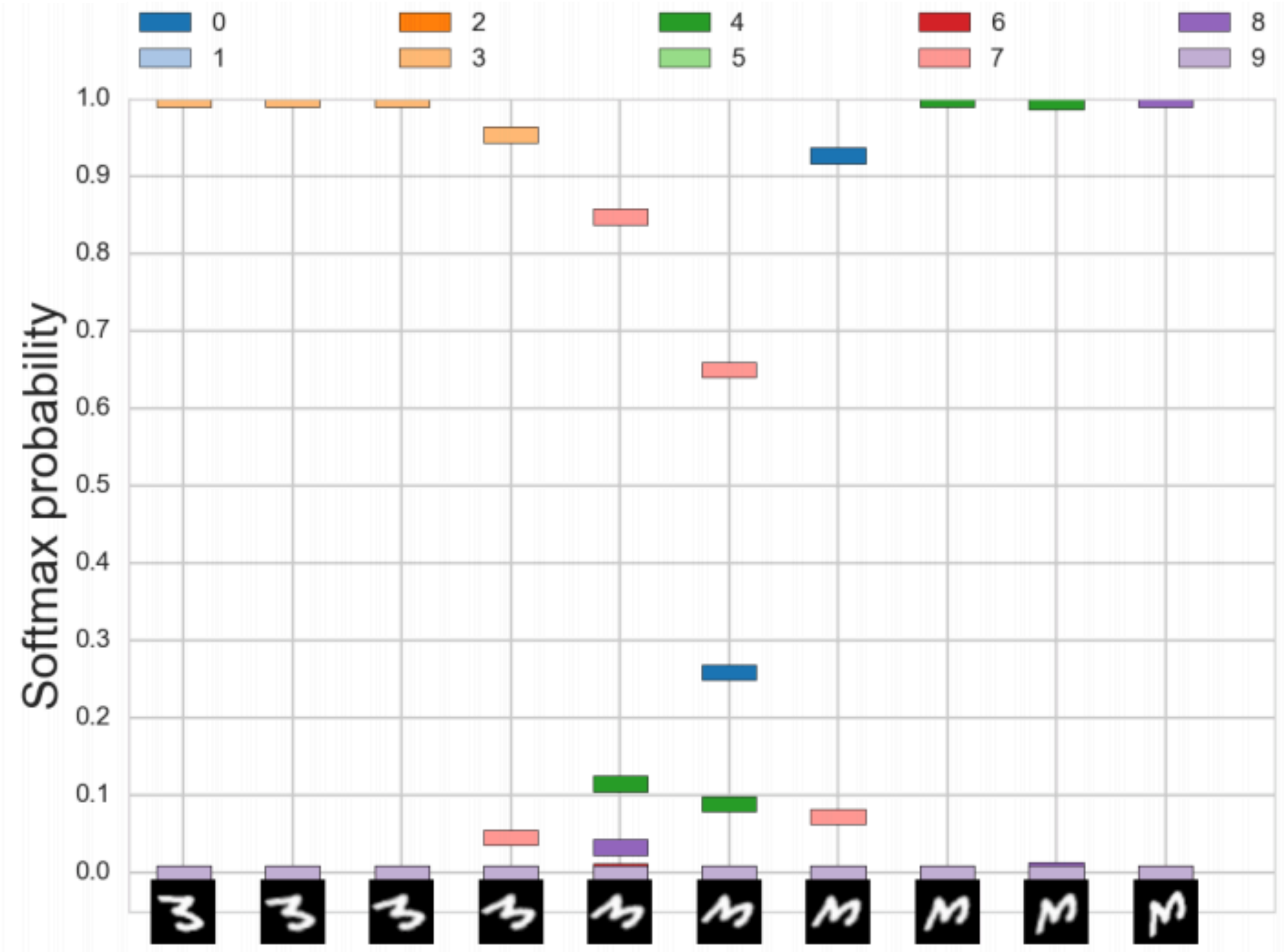
PART #1

Validating Predictive Models

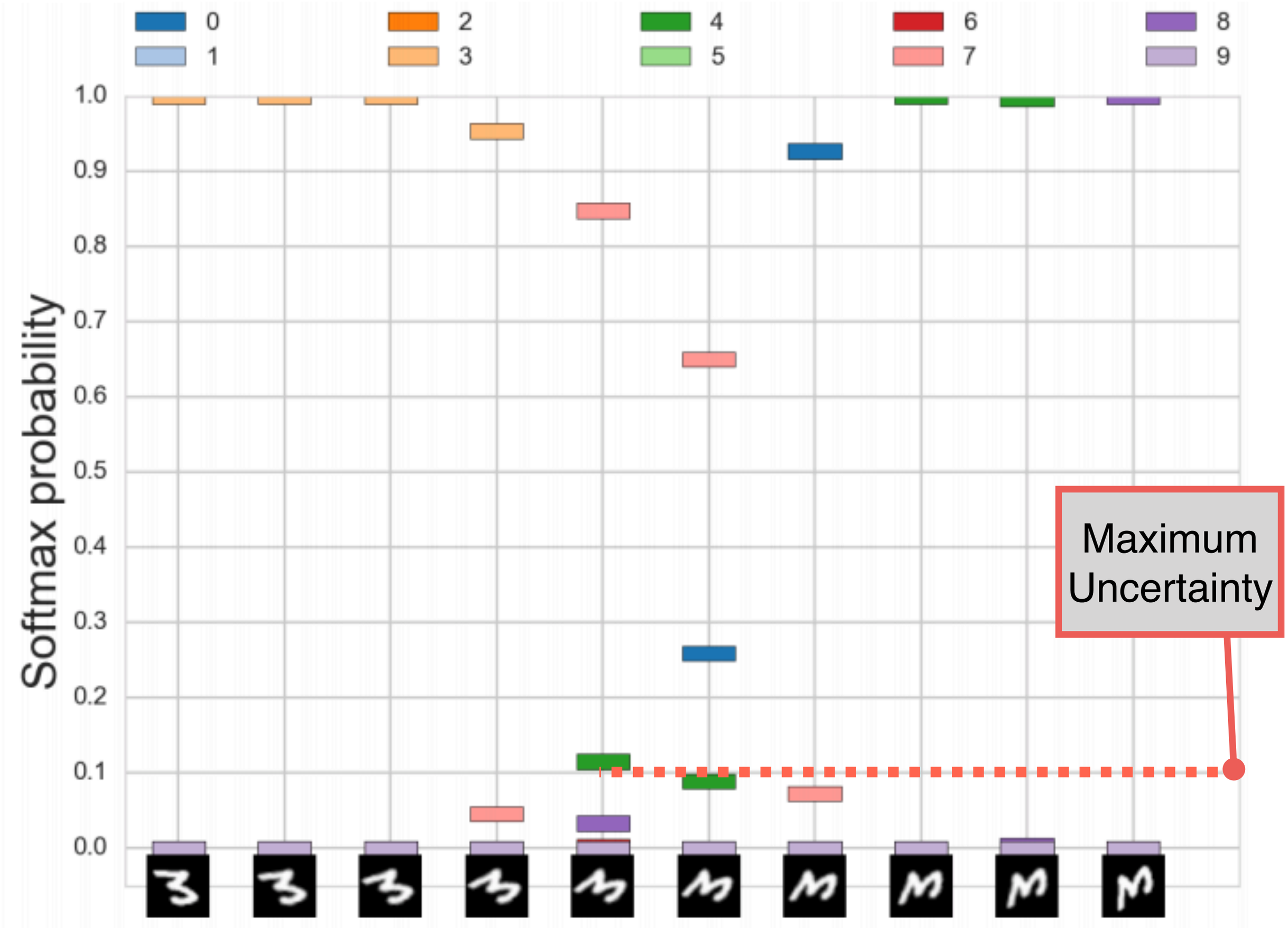
$$p(\underline{\mathbf{y}}|\underline{\mathbf{X}};\underline{\boldsymbol{\theta}})$$

Labels **Features** **Parameters**





[Louizos & Welling, ICML 2017]

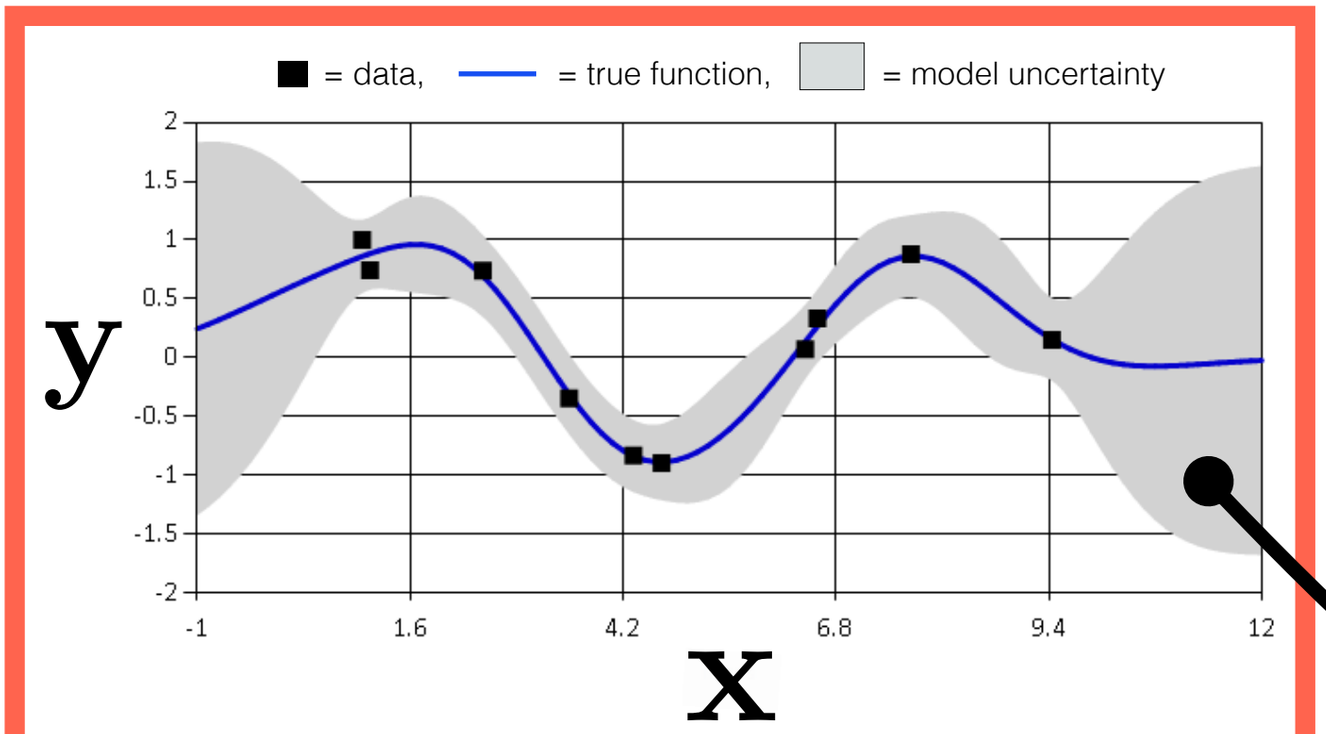


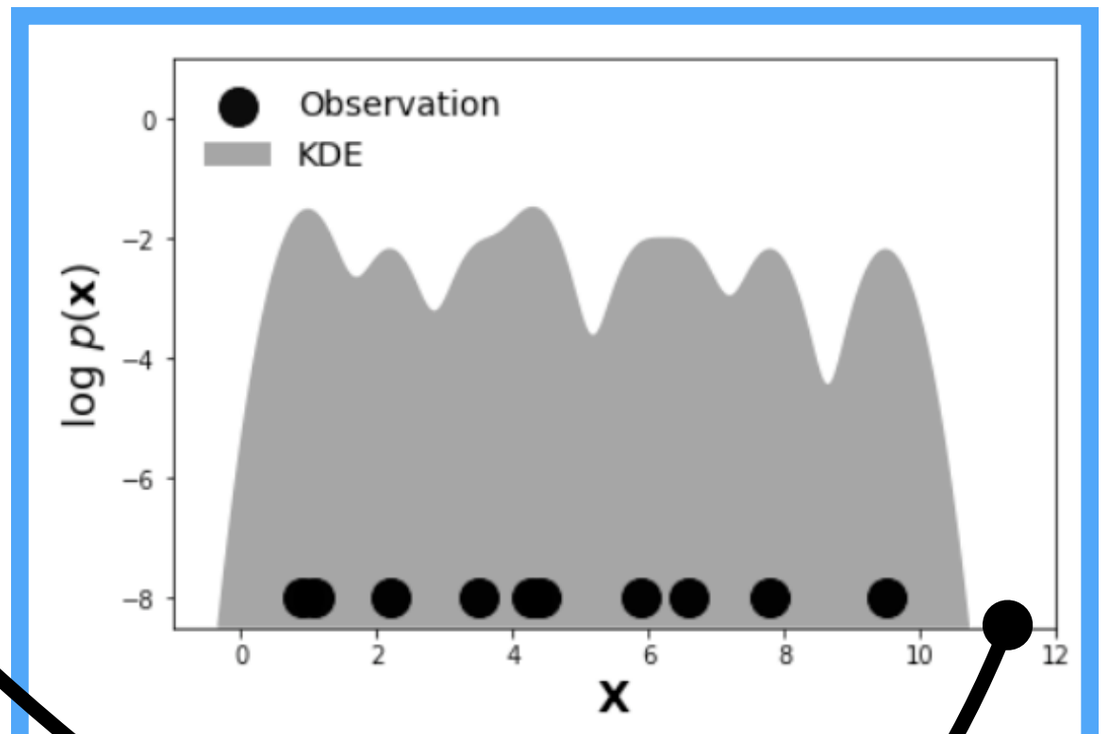
[Louizos & Welling, ICML 2017]

$$p(\mathbf{y}, \mathbf{X}) = p(\mathbf{y}|\mathbf{X}; \boldsymbol{\theta})\ p(\mathbf{X}; \boldsymbol{\phi})$$

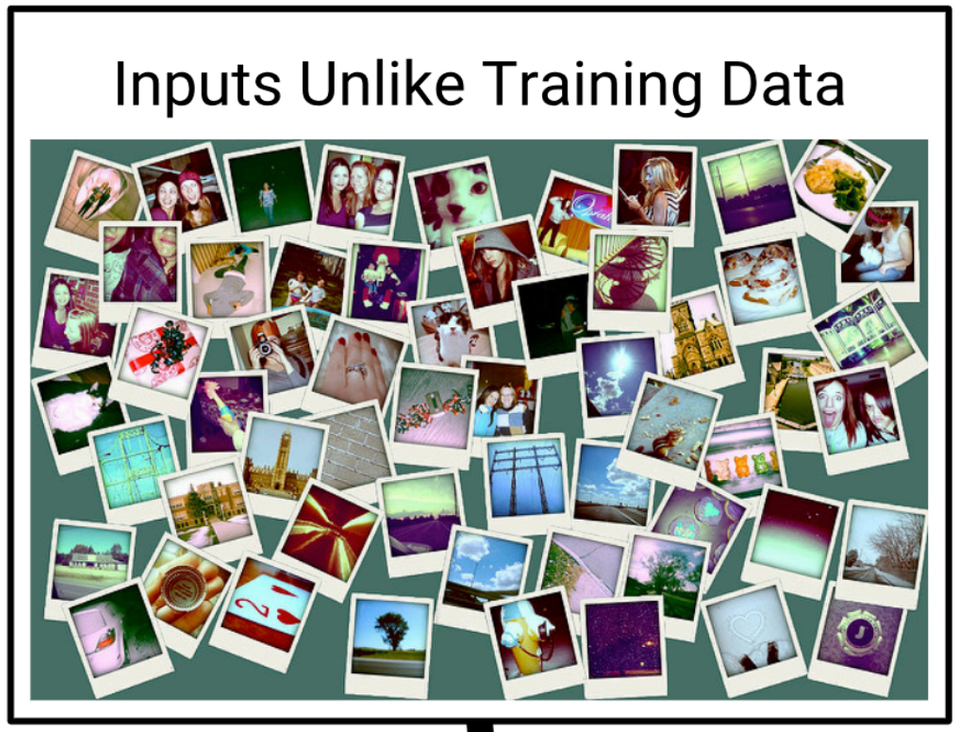
Predictive Model

Generative Model





Threshold-Based Rejection



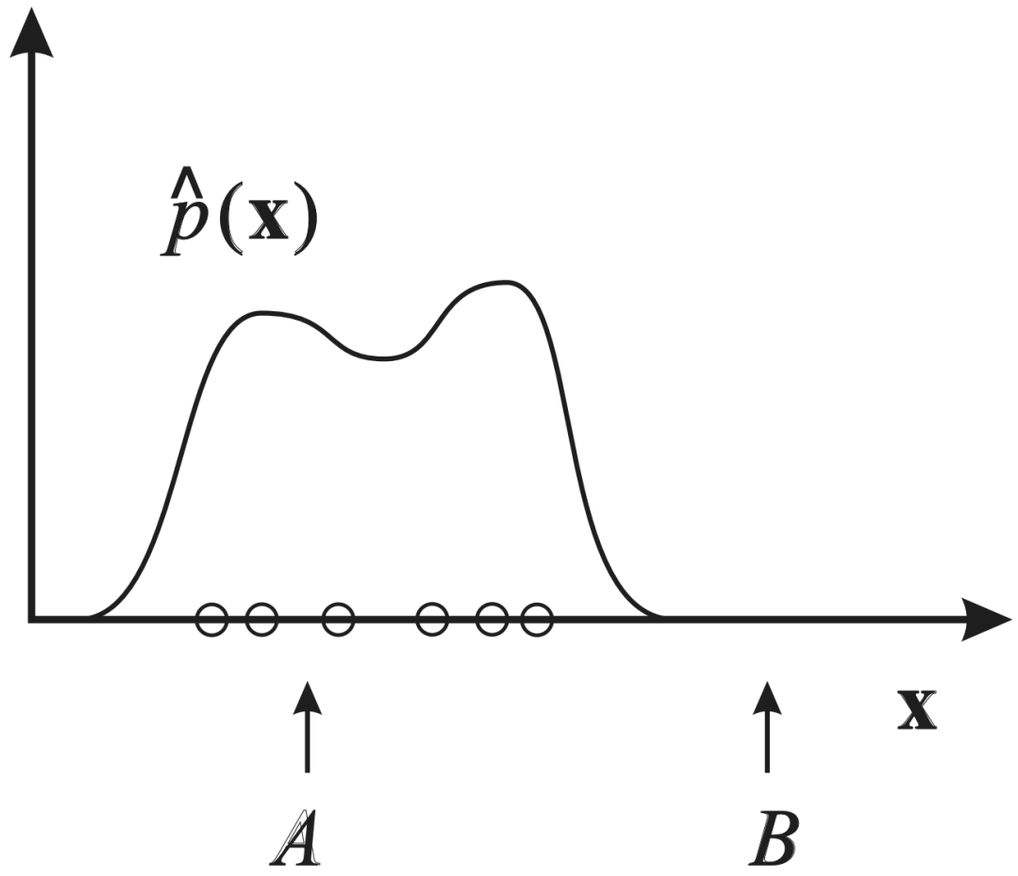
$$\text{if } p(\boldsymbol{x}^*; \boldsymbol{\phi}) < \tau,$$

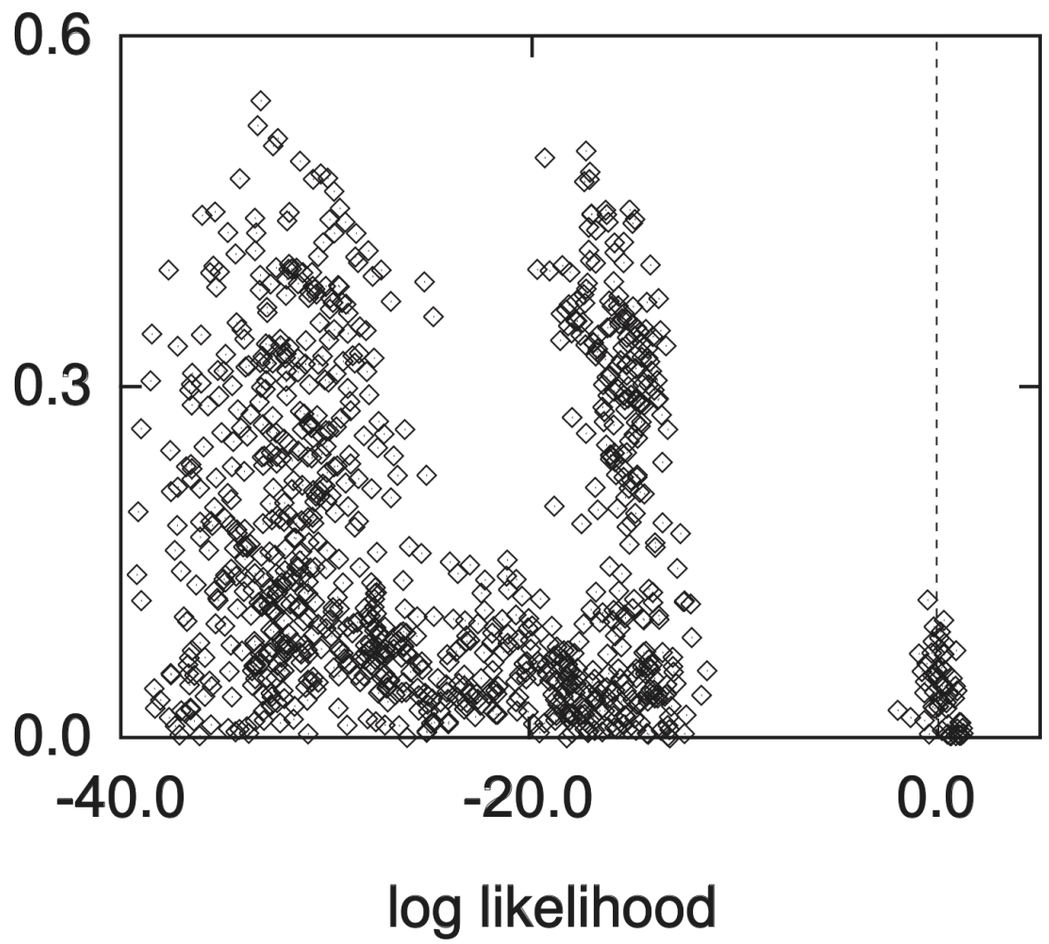
$$\text{then } \texttt{reject}\ \boldsymbol{x}^*$$

Use p(**X**) model to reject inputs with density below some threshold [Bishop, 1994].

Novelty Detection and Neural Network Validation

Chris M. Bishop (May 1994)



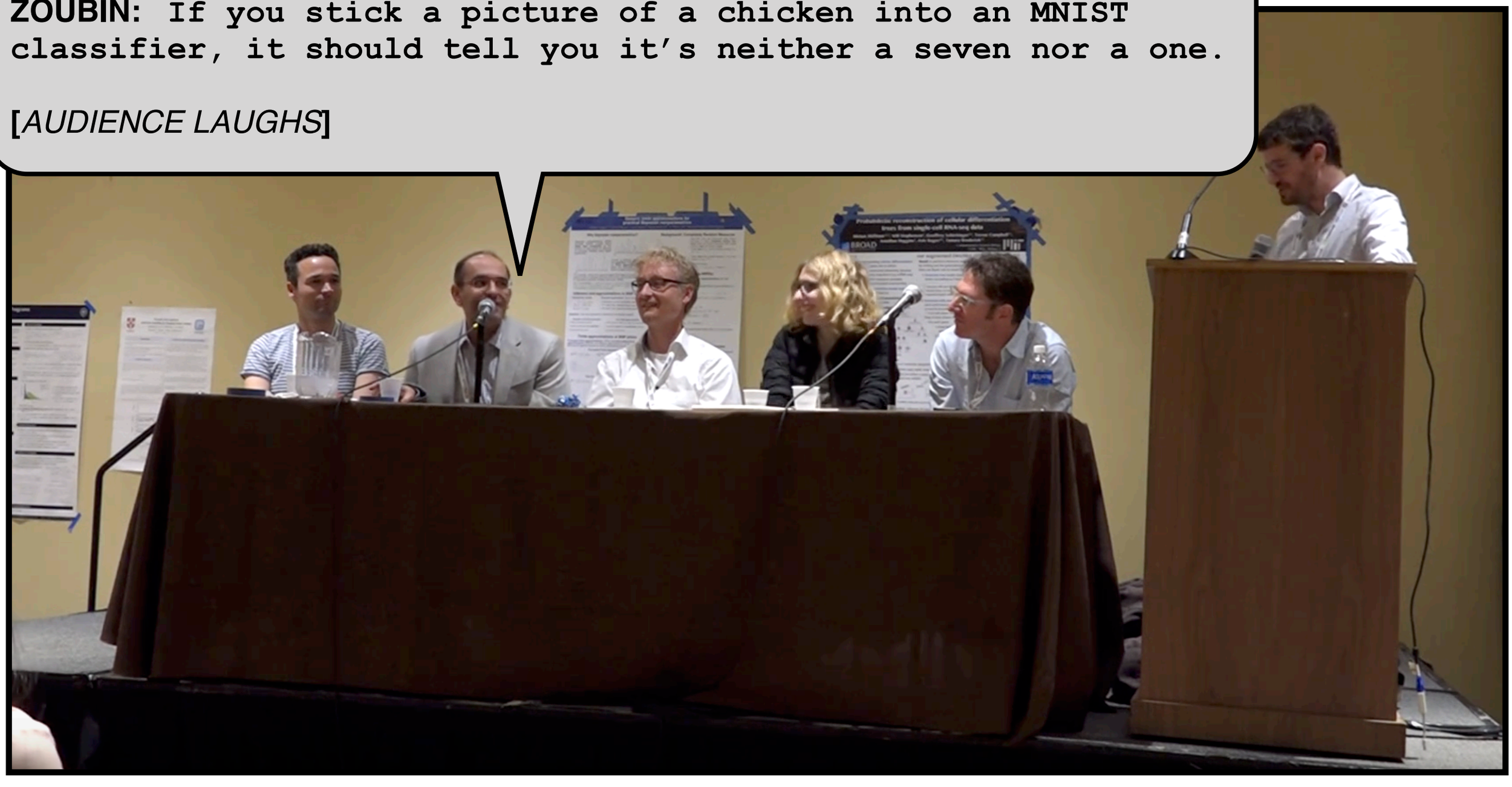


ZOUBIN: [The Bishop (1994) procedure] should be built into the software.

MODERATOR: Isn't that hard?

ZOUBIN: If you stick a picture of a chicken into an MNIST classifier, it should tell you it's neither a seven nor a one.

[*AUDIENCE LAUGHS*]

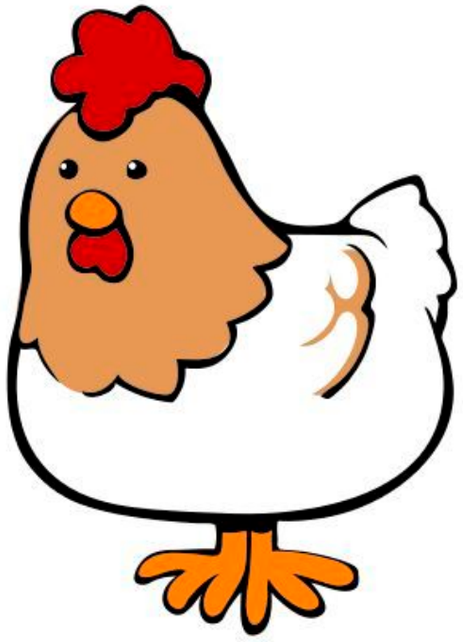


Panel Discussion

Advances in Approximate Bayesian Inference, Dec 2017

PART #2

Chicken or Seven?



Deep Generative Models

- [ ] **Autoregressive Models** (e.g. WaveNet)

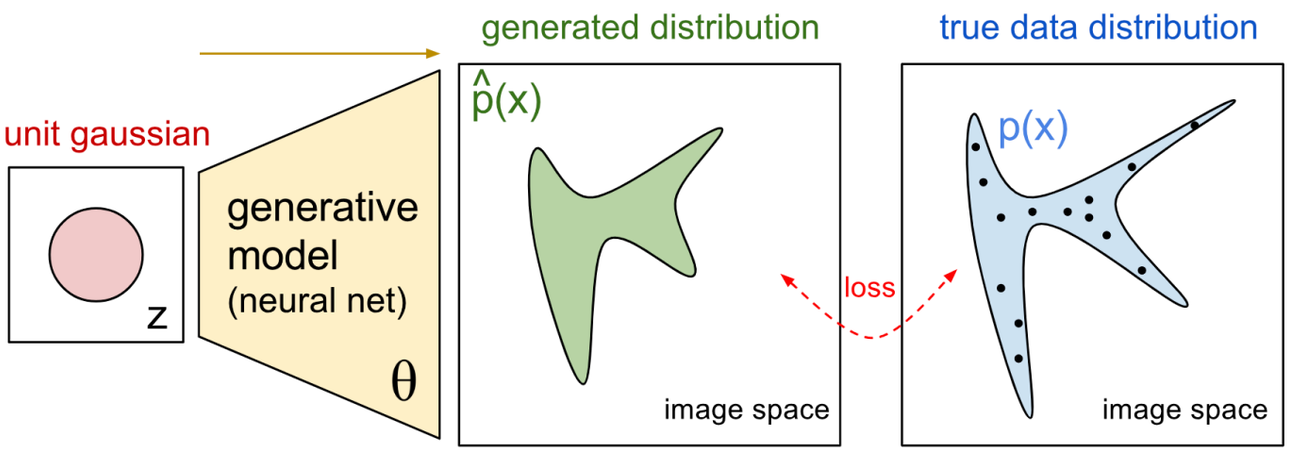
[Larochelle & Murray, AISTATS 2011; van den Oord, ICLM 2016; van den Oord, NeurIPS 2016]

- [ ] **Variational Autoencoders** (VAEs)

[Kingma & Welling, ICLR 2014; Rezende et al., ICML 2014]

- [ ] **Normalizing Flows**

[Tabak & Turner, 2013; Rezende & Mohamed, ICML 2015; Dinh et al., ICLR 2017]

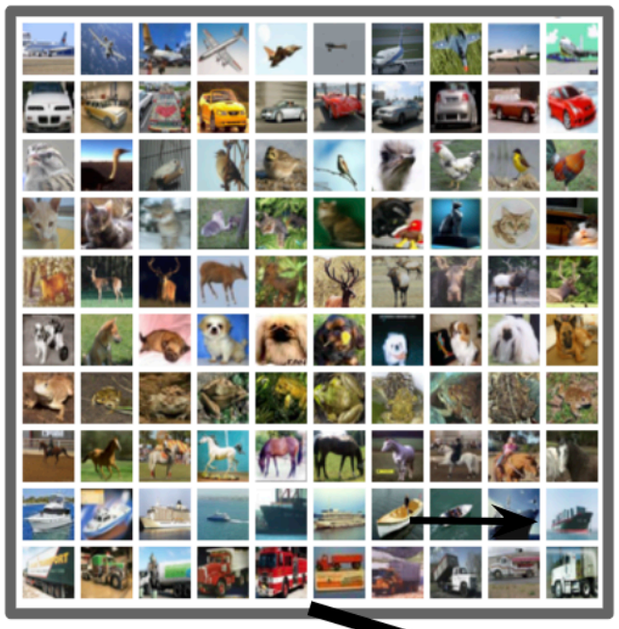


https://blog.openai.com/generative-models/

[Kingma & Dhariwal, NeurIPS 2018]

Chicken or Seven?

Training: *CIFAR-10*



Testing: *SVHN*

GENERATIVE MODEL

$$p(\mathbf{x}_{\text{CIFAR-10}}) \overset{?}{>} p(\mathbf{x}_{\text{SVHN}})$$

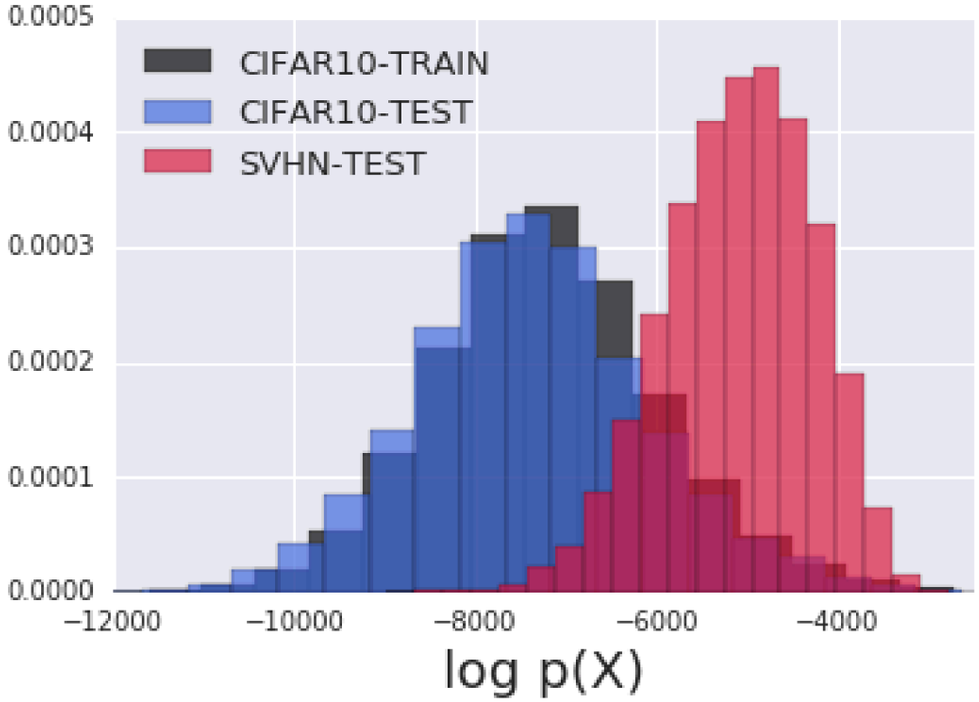
CIFAR-10 vs SVHN

First let's try a normalizing flow, Glow [Kingma & Dhariwal, 2018]:

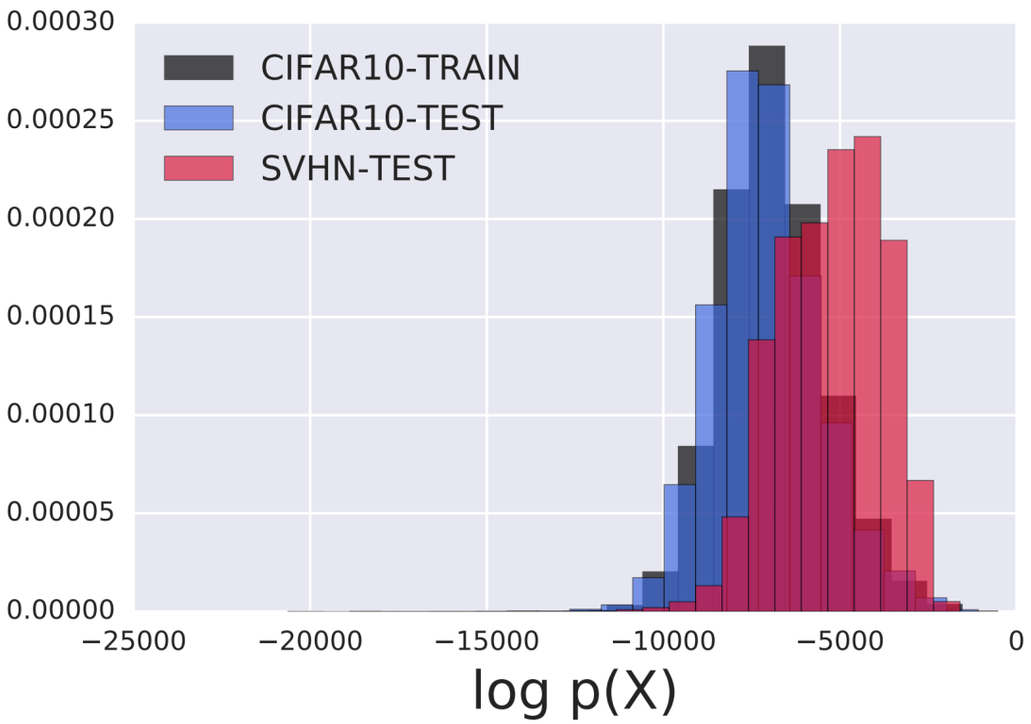
(lower is better)

Data Set	Avg. Bits Per Dimension
Glow Trained on CIFAR-10	
CIFAR10-Train	3.386
CIFAR10-Test	3.464
SVHN-Test	**2.389**
Glow Trained on SVHN	
SVHN-Test	2.057

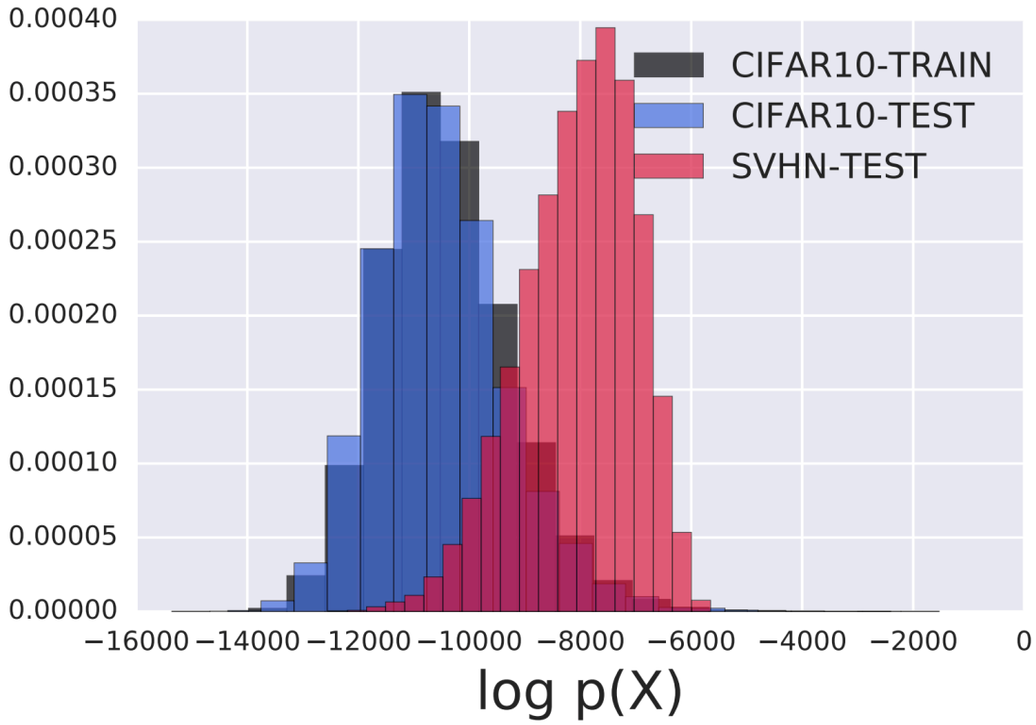
CIFAR-10 vs SVHN



Glow (Normalizing Flow)



Variational Autoencoder



PixelCNN

Shirt or Seven?

Training: *FashionMNIST*

Testing: *MNIST*

GENERATIVE MODEL

$$p(\mathbf{x}_{\text{FashionMNIST}}) \overset{?}{>} p(\mathbf{x}_{\text{MNIST}})$$

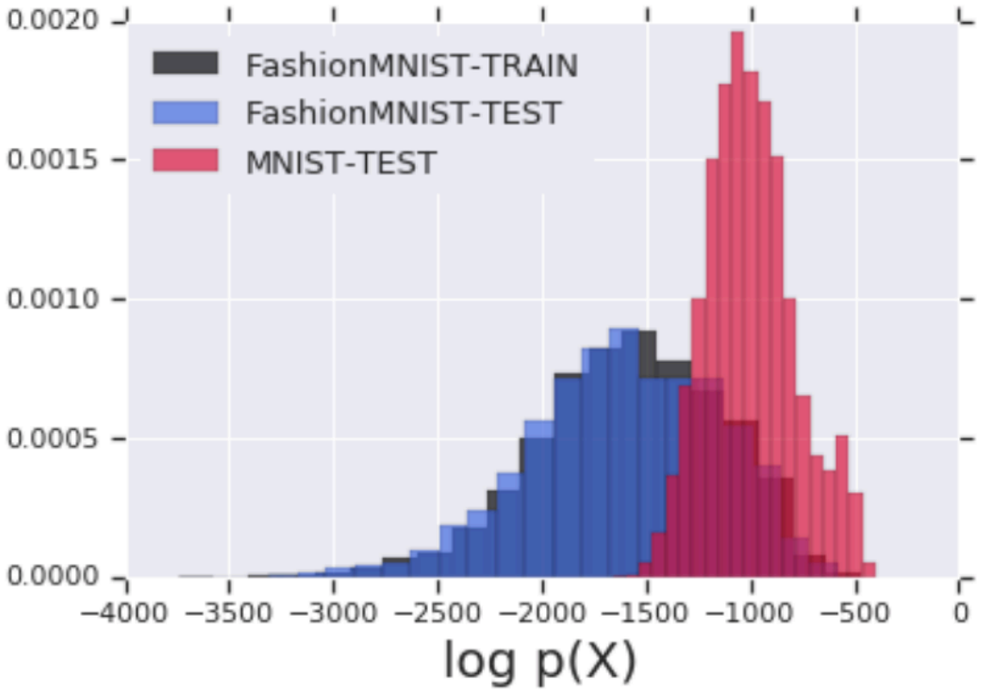
FashionMNIST vs MINST

Again, let's first try Glow [Kingma & Dhariwal, 2018]:

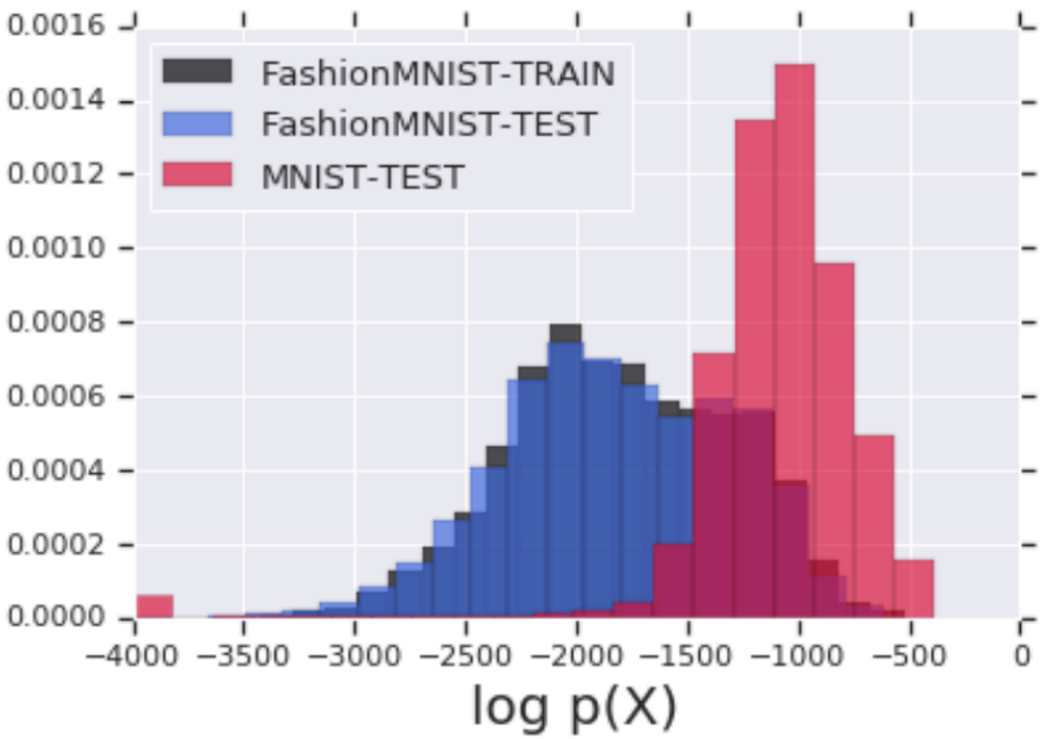
(lower is better)

Data Set	Avg. Bits Per Dimension
Glow Trained on FashionMNIST	
FashionMNIST-Train	2.902
FashionMNIST-Test	2.958
MNIST-Test	**1.833**
Glow Trained on MNIST	
MNIST-Test	1.262

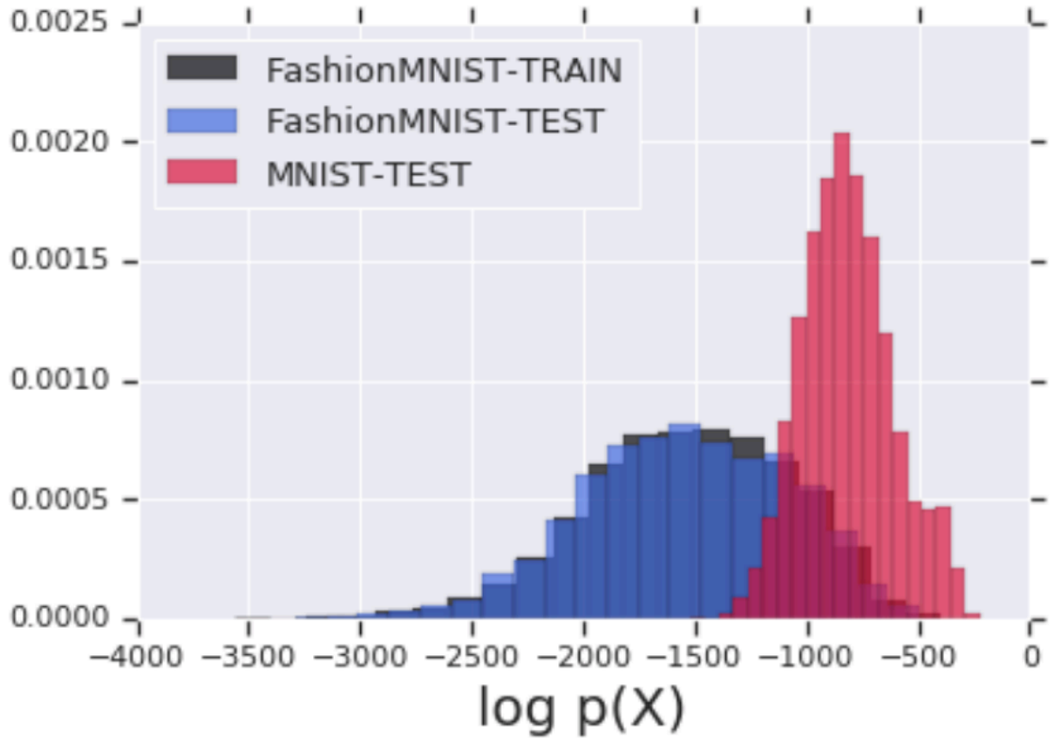
FashionMNIST vs MNIST



Glow (Normalizing Flow)



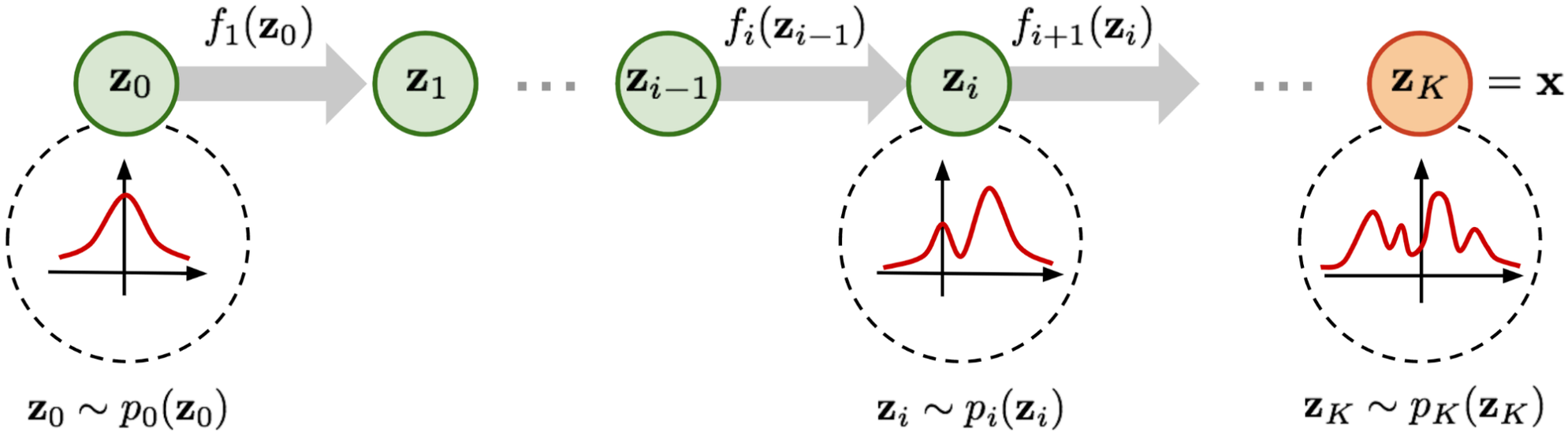
Variational Autoencoder



PixelCNN

PART #3

Digging Deeper into Flow-Based Models



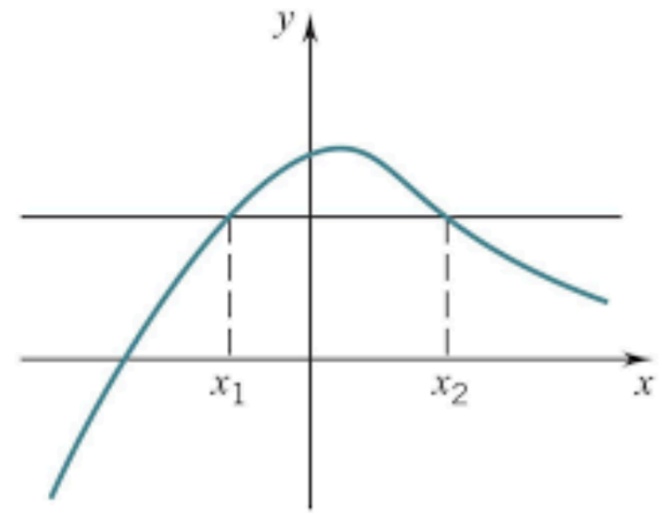
Normalizing Flows: Background

Change of Variables Formula (X → Z):

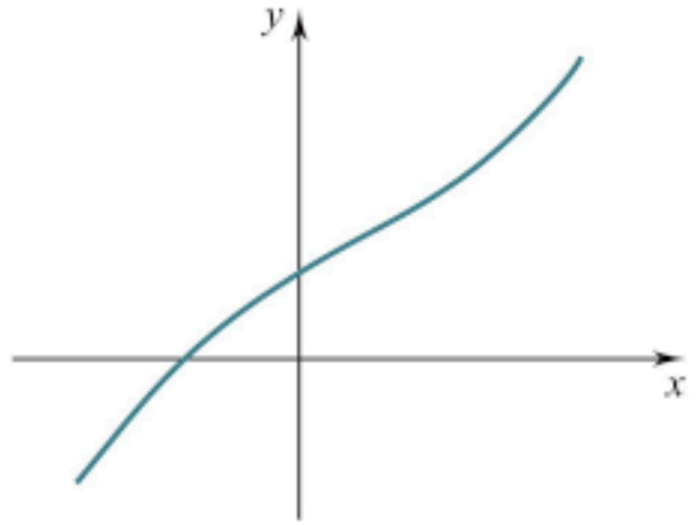
$$p_z(f(X)) \left| \frac{df(X)}{dX} \right| = p(X)$$

So what's the catch?

f(**x**) must be a *bijection*

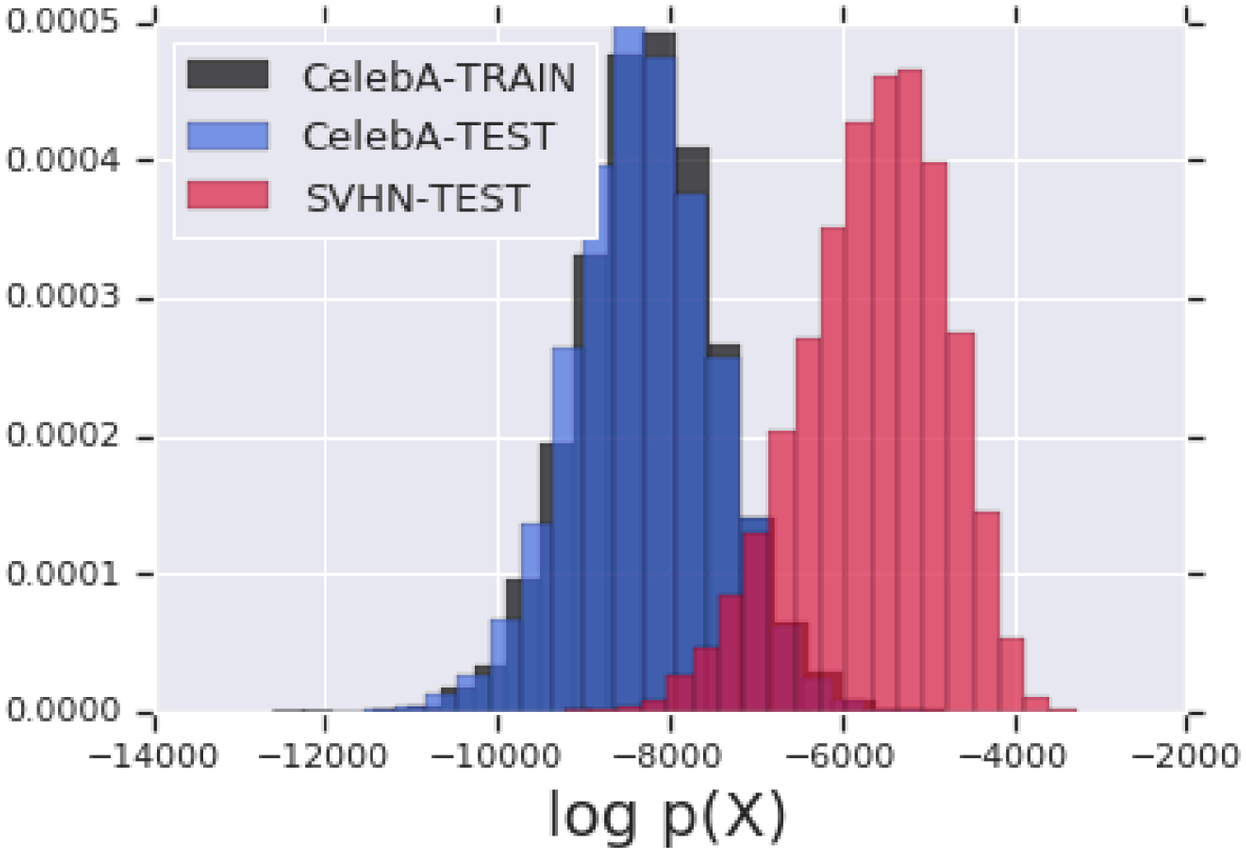


f is not one-to-one: $f(x_1) = f(x_2)$

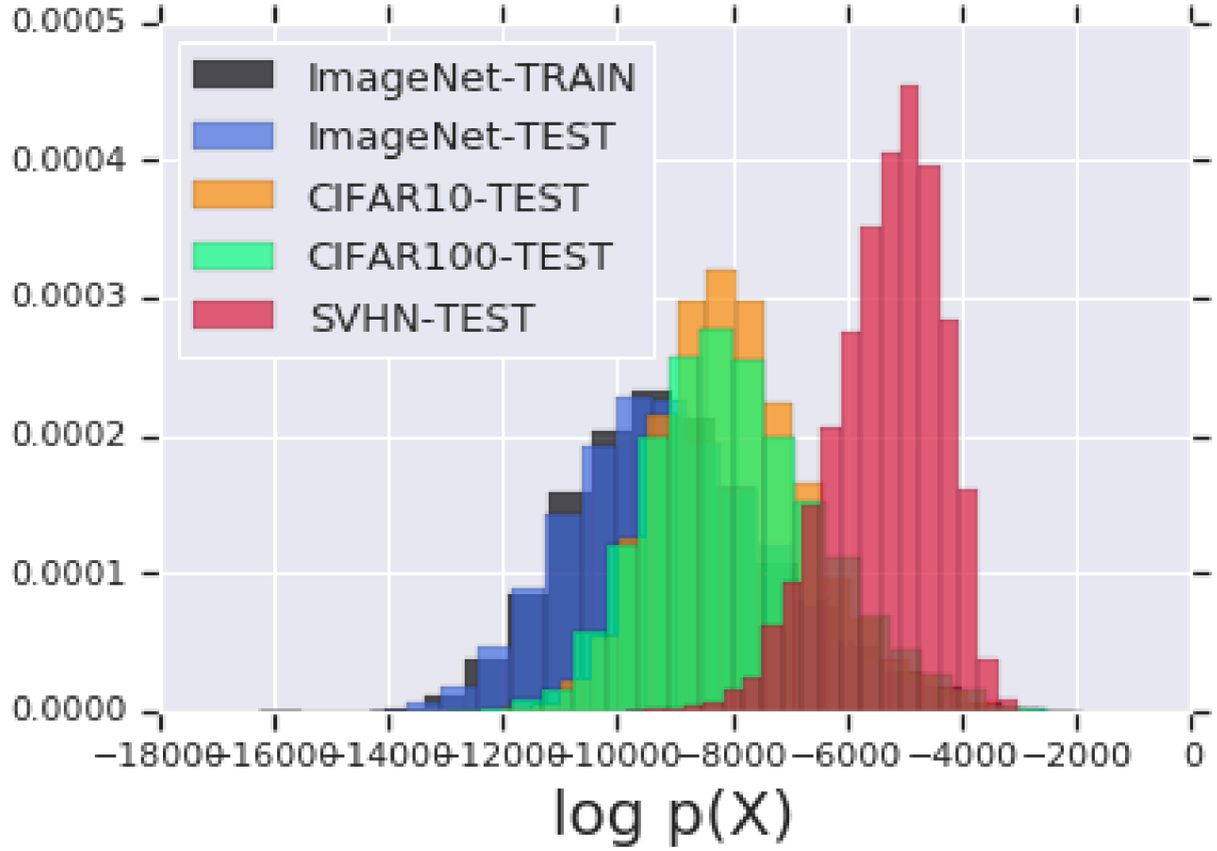


f is one-to-one:

Additional Data Sets

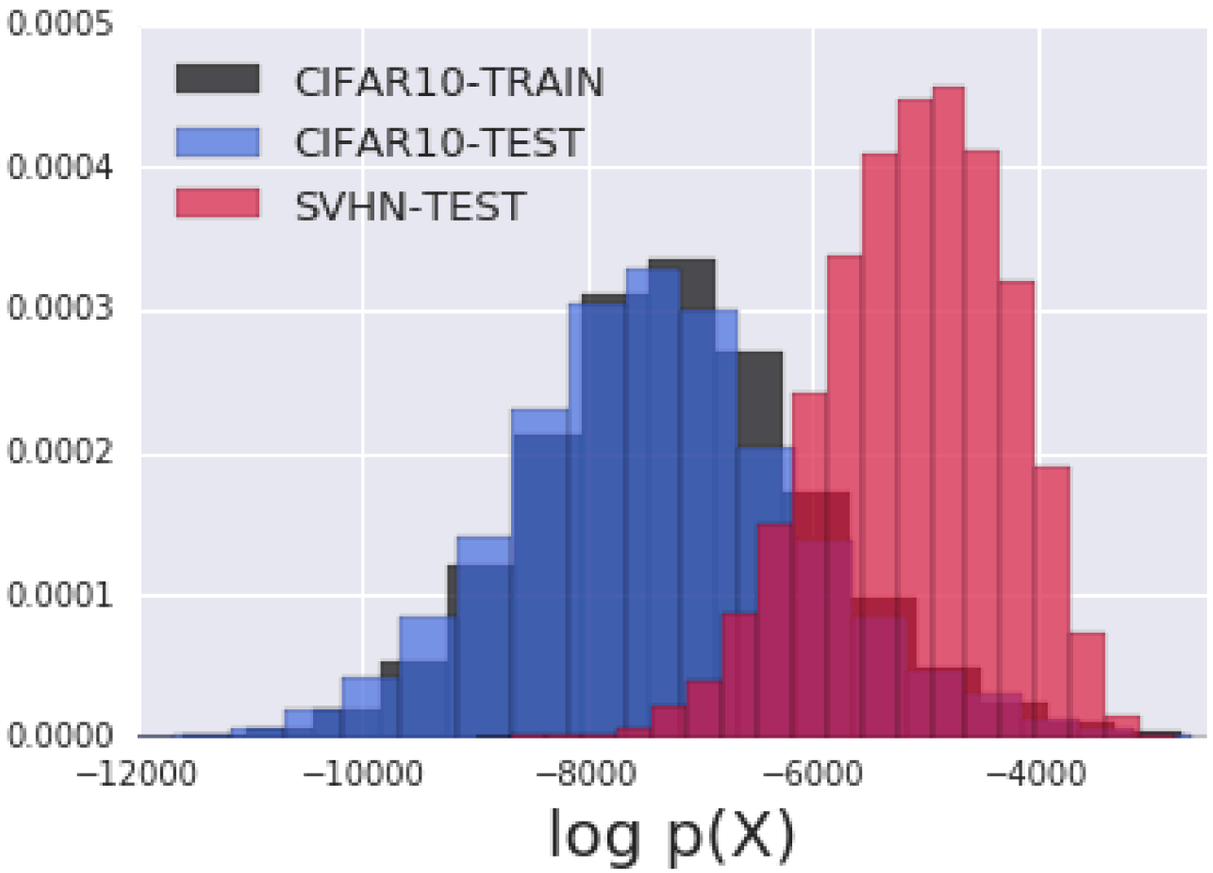


CelebA vs SVHN

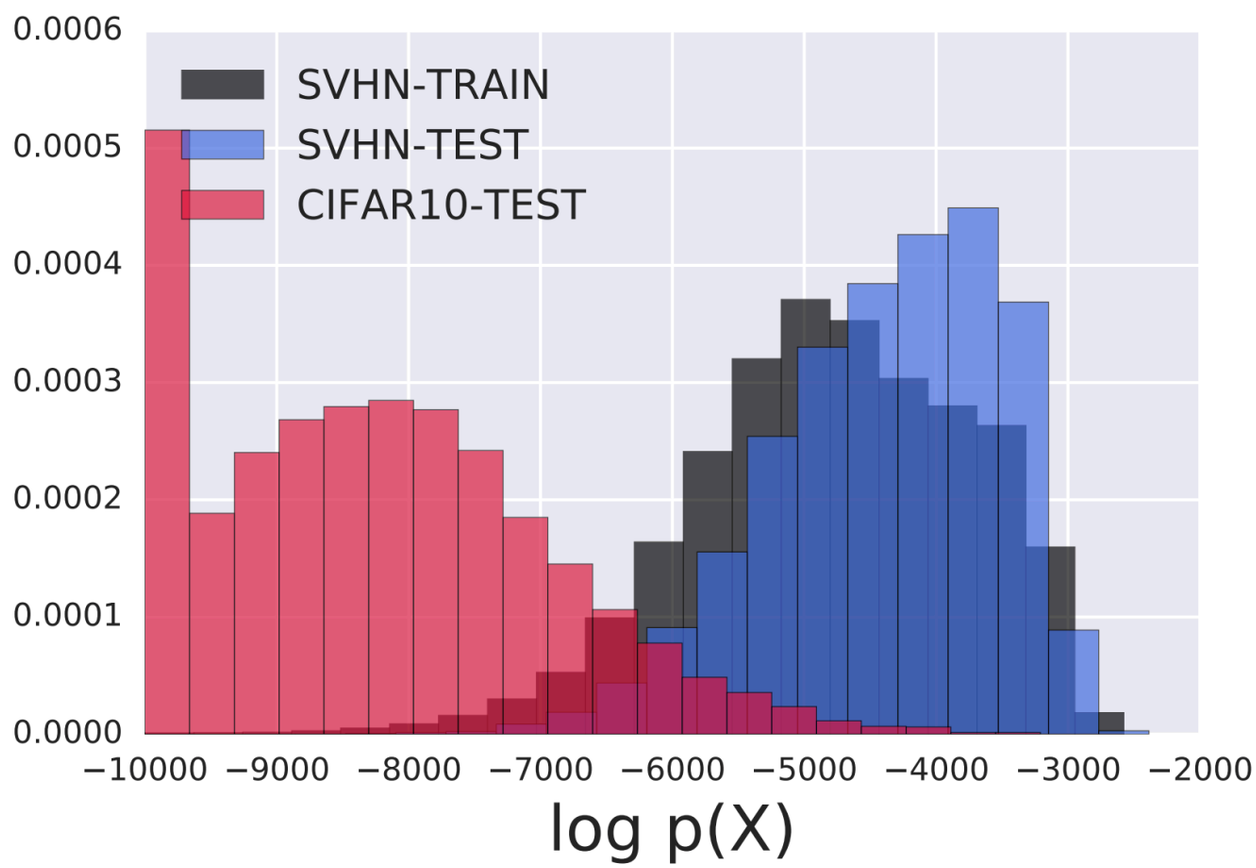


ImageNet vs CIFAR-10 vs SVHN

The Phenomenon is Not Symmetric



CIFAR-10 vs SVHN

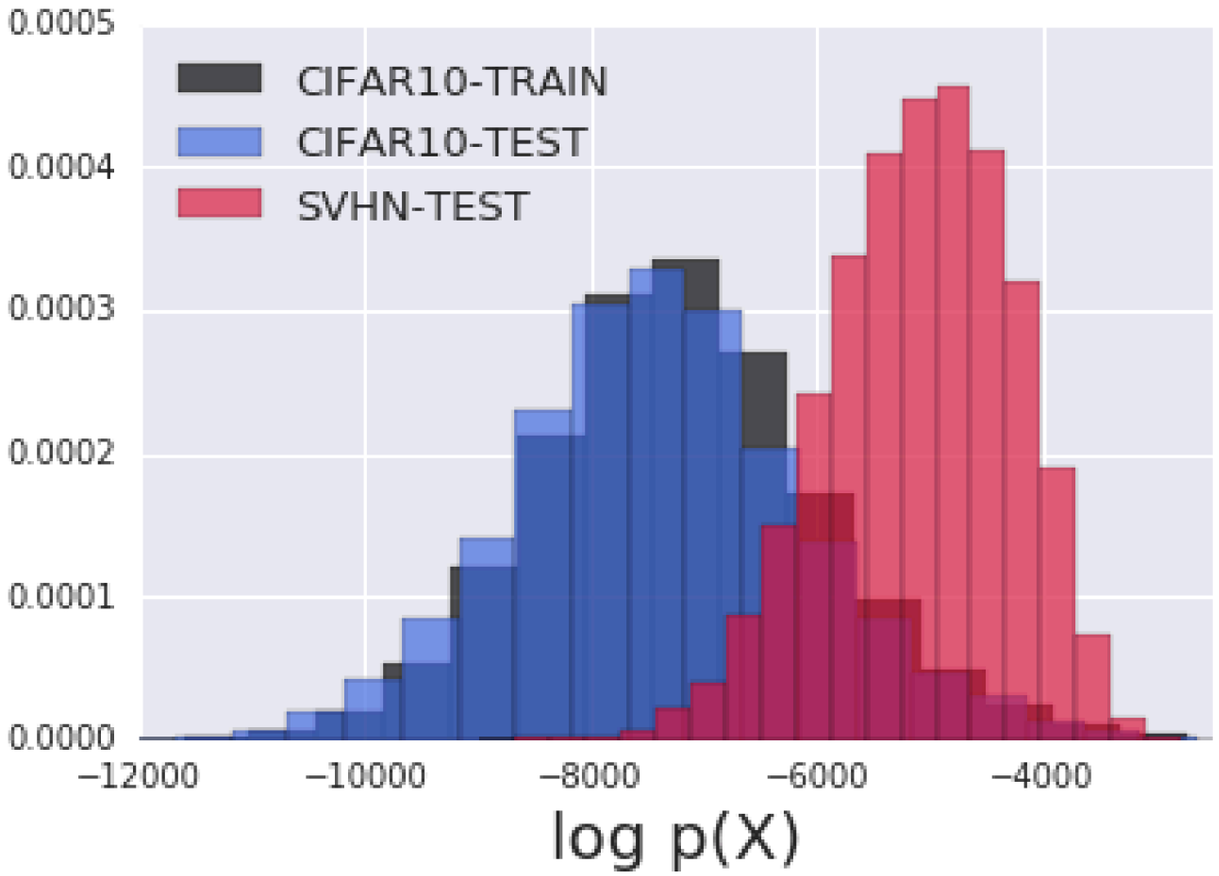


SVHN vs CIFAR-10

Uniform (Constant) Inputs

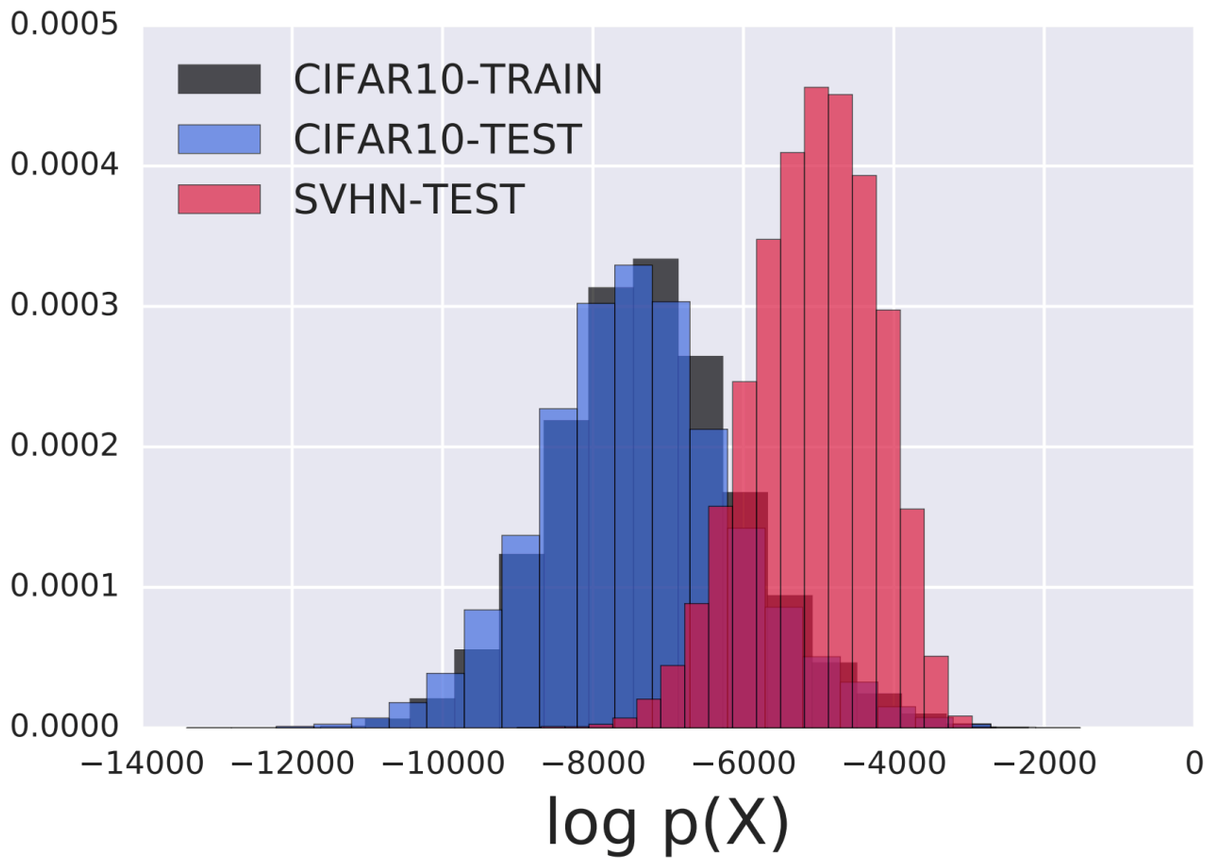
Data Set	Avg. Bits Per Dimension
Glow Trained on FashionMNIST	
Random	8.686
Constant (0)	**0.339**
Glow Trained on CIFAR-10	
Random	15.773
Constant (128)	**0.589**

Ensembling Makes No Difference



CIFAR-10 vs SVHN

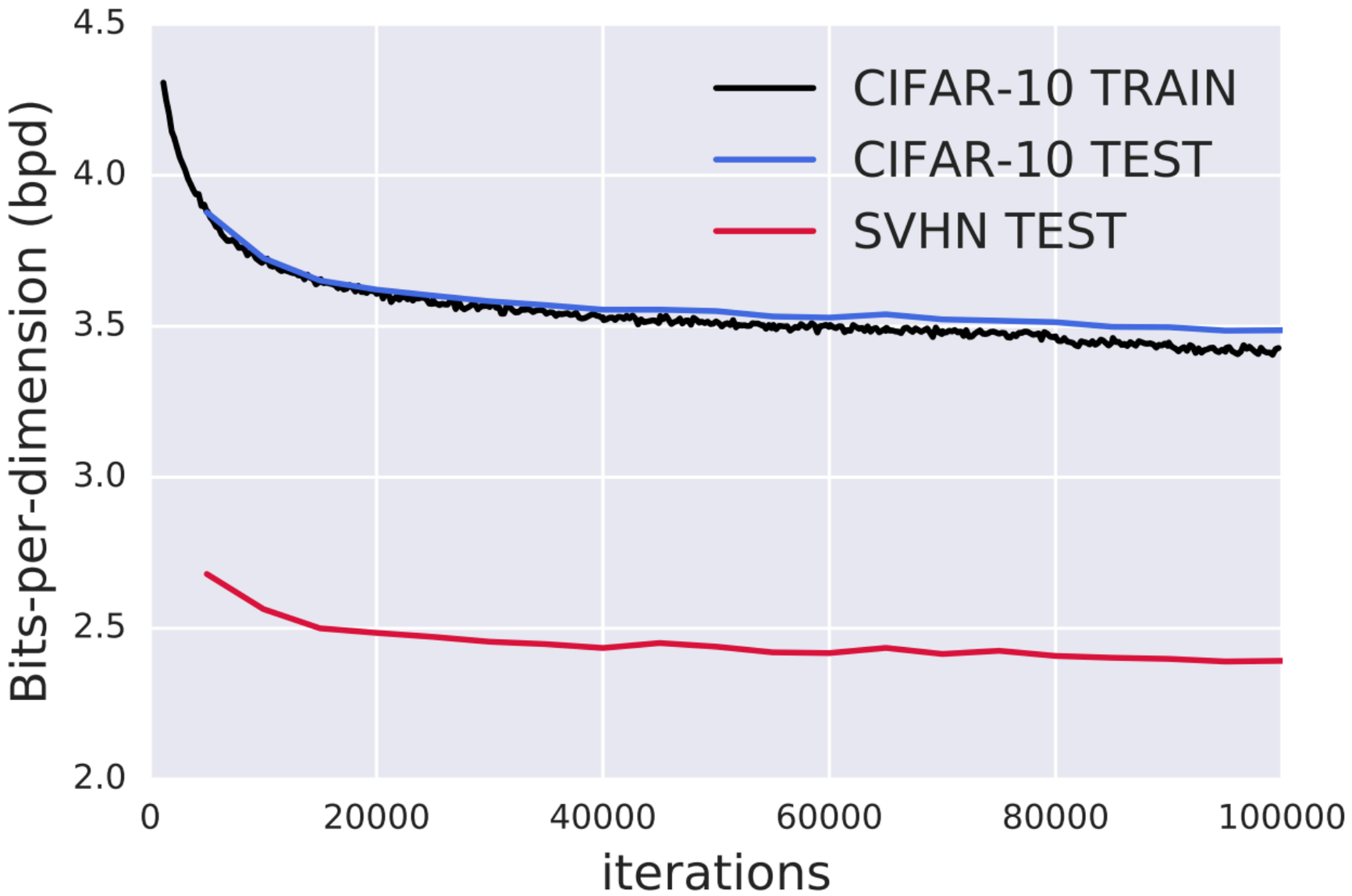
1 Glow



CIFAR-10 vs SVHN

Ensemble of 10 Glows

Early-Stopping Makes No Difference



During Optimization

Switching to an Analytical Approach

Mathematical characterization:

$$0 < \mathbb{E}_q[\log p(\boldsymbol{x};\boldsymbol{\theta})] - \mathbb{E}_{p^*}[\log p(\boldsymbol{x};\boldsymbol{\theta})]$$

Non-Training Distribution

Training Distribution

Second Moment of Training Distribution

$$\approx \frac{1}{2}\operatorname{Tr}\left\{\left[\nabla^2_{\boldsymbol{x}_0}\log p_z(f(\boldsymbol{x}_0;\boldsymbol{\phi})) + \nabla^2_{\boldsymbol{x}_0}\log\left|\frac{\partial \boldsymbol{f}_{\boldsymbol{\phi}}}{\partial \boldsymbol{x}_0}\right|\right](\boldsymbol{\Sigma}_q - \boldsymbol{\Sigma}_{p^*})\right\}$$

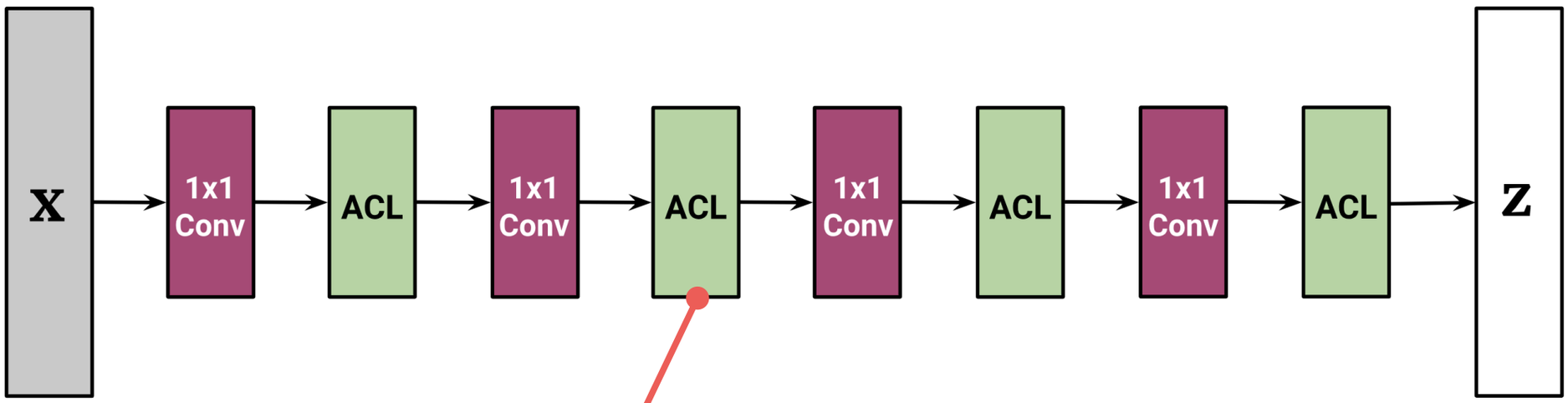
Change-of-Variable Terms

Second Moment of Non-Training Distribution

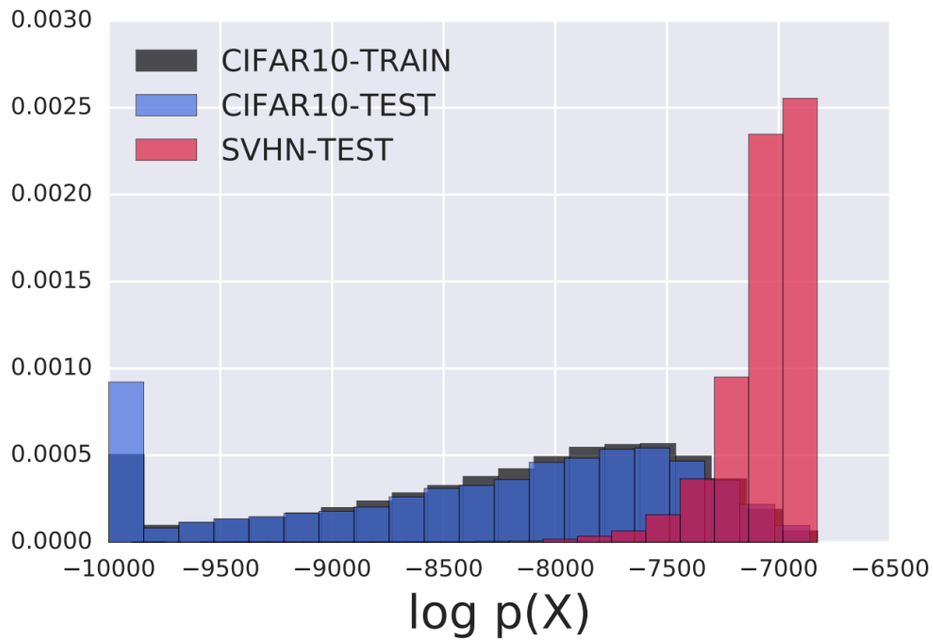
Analysis via Constant-Volume Flows

We define a sub-class we term *constant-volume* (w.r.t. input) flows.

Glow Architecture:



Use only *translation* operations.



CIFAR-10 vs SVHN

Analysis via Constant-Volume Flows

Mathematical characterization:

$$0 < \underline{\mathbb{E}_q}\left[\log p(\boldsymbol{x};\boldsymbol{\theta})\right] - \underline{\mathbb{E}_{p^*}}\left[\log p(\boldsymbol{x};\boldsymbol{\theta})\right]$$

Non-Training Distribution

Training Distribution

Second Moment of Training Distribution

$$\approx \frac{1}{2}\operatorname{Tr}\left\{\left[\nabla^2_{\boldsymbol{x}_0}\log p_z(f(\boldsymbol{x}_0;\boldsymbol{\phi})) + \nabla^2_{\boldsymbol{x}_0}\log\left|\frac{\partial f_{\boldsymbol{\phi}}}{\partial \boldsymbol{x}_0}\right|\right](\underline{\boldsymbol{\Sigma}_q} - \overline{\boldsymbol{\Sigma}_{p^*}})\right\}$$

Change-of-Variable Terms

Second Moment of Non-Training Distribution

Analysis via Constant-Volume Flows

Plugging in the CV-Glow transform:

$$\mathrm{Tr}\left\{\left[\nabla^2_{\boldsymbol{x}_0} \log p(\boldsymbol{x}_0;\boldsymbol{\theta})\right](\boldsymbol{\Sigma}_q - \boldsymbol{\Sigma}_{p^*})\right\}$$

Analysis via Constant-Volume Flows

Plugging in the CV-Glow transform:

$$\mathrm{Tr}\left\{\left[\nabla^2_{\boldsymbol{x}_0}\log p(\boldsymbol{x}_0;\boldsymbol{\theta})\right](\boldsymbol{\Sigma}_q-\boldsymbol{\Sigma}_{p^*})\right\}$$

$$=\frac{\partial^2}{\partial z^2}\log p(\boldsymbol{z};\boldsymbol{\psi})\sum_{c=1}^{C}\left(\prod_{k=1}^{K}\sum_{j=1}^{C}u_{k,c,j}\right)^2\sum_{h,w}(\sigma^2_{q,h,w,c}-\sigma^2_{p^*,h,w,c})$$

Analysis via Constant-Volume Flows

Plugging in the CV-Glow transform:

$$\mathrm{Tr}\left\{\left[\nabla^2_{\boldsymbol{x}_0} \log p(\boldsymbol{x}_0;\boldsymbol{\theta})\right](\boldsymbol{\Sigma}_q - \boldsymbol{\Sigma}_{p^*})\right\}$$

$$= \frac{\partial^2}{\partial z^2} \log p(\boldsymbol{z};\boldsymbol{\psi}) \sum_{c=1}^{C} \left(\prod_{k=1}^{K} \sum_{j=1}^{C} u_{k,c,j} \right)^2 \sum_{h,w} (\sigma^2_{q,h,w,c} - \sigma^2_{p^*,h,w,c})$$

Base Distribution

Analysis via Constant-Volume Flows

Plugging in the CV-Glow transform:

$$\mathrm{Tr}\left\{\left[\nabla^2_{\boldsymbol{x}_0} \log p(\boldsymbol{x}_0;\boldsymbol{\theta})\right](\boldsymbol{\Sigma}_q - \boldsymbol{\Sigma}_{p^*})\right\}$$

$$= \frac{\partial^2}{\partial z^2} \log p(\boldsymbol{z};\boldsymbol{\psi}) \sum_{c=1}^{C} \left(\prod_{k=1}^{K} \sum_{j=1}^{C} u_{k,c,j} \right)^2 \sum_{h,w} (\sigma^2_{q,h,w,c} - \sigma^2_{p^*,h,w,c})$$

Base Distribution

1x1 Conv. Params

Analysis via Constant-Volume Flows

Plugging in the CV-Glow transform:

$$\mathrm{Tr}\left\{\left[\nabla^2_{\boldsymbol{x}_0} \log p(\boldsymbol{x}_0;\boldsymbol{\theta})\right](\boldsymbol{\Sigma}_q - \boldsymbol{\Sigma}_{p^*})\right\}$$

$$= \frac{\partial^2}{\partial z^2} \log p(\boldsymbol{z};\boldsymbol{\psi}) \sum_{c=1}^{C} \left(\prod_{k=1}^{K} \sum_{j=1}^{C} u_{k,c,j} \right)^2 \sum_{h,w} (\sigma^2_{q,h,w,c} - \sigma^2_{p^*,h,w,c})$$

Base Distribution

1x1 Conv. Params

Second Moment of Training Distribution

Second Moment of Non-Training Distribution

Analysis via Constant-Volume Flows

Plugging in the CV-Glow transform:

$$\mathrm{Tr}\left\{\left[\nabla^2_{\boldsymbol{x}_0} \log p(\boldsymbol{x}_0;\boldsymbol{\theta})\right](\boldsymbol{\Sigma}_q - \boldsymbol{\Sigma}_{p^*})\right\}$$

$$= \underbrace{\frac{\partial^2}{\partial z^2} \log p(\boldsymbol{z};\boldsymbol{\psi})}_{\text{Base Distribution}} \sum_{c=1}^{C} \underbrace{\left(\prod_{k=1}^{K} \sum_{j=1}^{C} u_{k,c,j}\right)^2}_{\text{1x1 Conv. Params}} \sum_{h,w} (\sigma^2_{q,h,w,c} - \sigma^2_{p^*,h,w,c})$$

Second Moment of Training Distribution

Second Moment of Non-Training Distribution

Negative for all log-concave densities (e.g. Gaussian)

Non-negative due to square

Analysis via Constant-Volume Flows

$$0 < \mathbb{E}_q[\log p(\boldsymbol{x};\boldsymbol{\theta})] - \mathbb{E}_{p^*}[\log p(\boldsymbol{x};\boldsymbol{\theta})]$$

Non-Training Distribution

Training Distribution

Analysis via Constant-Volume Flows

$$0 < \underline{\mathbb{E}_q}[\log p(\boldsymbol{x};\boldsymbol{\theta})] - \underline{\mathbb{E}_{p^*}}[\log p(\boldsymbol{x};\boldsymbol{\theta})]$$

Non-Training Distribution

Training Distribution

$$\approx \frac{\partial^2}{\partial z^2}\log p(\boldsymbol{z};\boldsymbol{\psi}) \sum_{c=1}^{C}\left(\prod_{k=1}^{K}\sum_{j=1}^{C} u_{k,c,j}\right) \sum_{h,w}(\sigma^2_{q,h,w,c} - \sigma^2_{p^*,h,w,c})$$

Analysis via Constant-Volume Flows

$$0 < \mathbb{E}_q[\log p(\boldsymbol{x};\boldsymbol{\theta})] - \mathbb{E}_{p^*}[\log p(\boldsymbol{x};\boldsymbol{\theta})]$$

Non-Training Distribution

Training Distribution

Second Moment of Training Distribution

$$\approx \left(-\frac{\partial^2}{\partial z^2}\log p(\boldsymbol{z};\boldsymbol{\psi})\right) \sum_{c=1}^{C} \left(\prod_{k=1}^{K}\sum_{j=1}^{C} u_{k,c,j}\right) \sum_{h,w} (\sigma^2_{q,h,w,c} - \sigma^2_{p^*,h,w,c})$$

Second Moment of Non-Training Distribution

Analysis via Constant-Volume Flows

$$0 < \mathbb{E}_q[\log p(\boldsymbol{x};\boldsymbol{\theta})] - \mathbb{E}_{p^*}[\log p(\boldsymbol{x};\boldsymbol{\theta})]$$

Non-Training Distribution

Training Distribution

Second Moment of Training Distribution

$$\approx \left(-\frac{\partial^2}{\partial z^2}\log p(\boldsymbol{z};\boldsymbol{\psi})\right) \sum_{c=1}^{C} \left(\prod_{k=1}^{K}\sum_{j=1}^{C} u_{k,c,j}\right) \sum_{h,w} (\sigma^2_{q,h,w,c} - \sigma^2_{p^*,h,w,c})$$

Second Moment of Non-Training Distribution

- [ ] CIFAR-10 vs SVHN
- [ ] Asymmetry
- [ ] Uniform Inputs
- [ ] Ensembling
- [ ] Early Stopping

Analysis via Constant-Volume Flows

$$0 < \mathbb{E}_q[\log p(\boldsymbol{x};\boldsymbol{\theta})] - \mathbb{E}_{p^*}[\log p(\boldsymbol{x};\boldsymbol{\theta})]$$

Non-Training Distribution

Training Distribution

Second Moment of Training Distribution

$$\approx \left(-\frac{\partial^2}{\partial z^2}\log p(\boldsymbol{z};\boldsymbol{\psi})\right) \sum_{c=1}^{C} \left(+\prod_{k=1}^{K}\sum_{j=1}^{C} u_{k,c,j}\right) \sum_{h,w} \left(-(\sigma^2_{q,h,w,c} - \sigma^2_{p^*,h,w,c})\right)$$

Second Moment of Non-Training Distribution

- [ ] **CIFAR-10 vs SVHN** (plugging in empirical moments)
- [ ] Asymmetry
- [ ] Uniform Inputs
- [ ] Ensembling
- [ ] Early Stopping

Analysis via Constant-Volume Flows

$$0 < \mathbb{E}_q[\log p(\boldsymbol{x};\boldsymbol{\theta})] - \mathbb{E}_{p^*}[\log p(\boldsymbol{x};\boldsymbol{\theta})]$$

Non-Training Distribution

Training Distribution

Second Moment of Training Distribution

$$\approx \left(-\frac{\partial^2}{\partial z^2}\log p(\boldsymbol{z};\boldsymbol{\psi})\right) \sum_{c=1}^{C} \left(\prod_{k=1}^{K}\sum_{j=1}^{C} u_{k,c,j}\right) \sum_{h,w} (\sigma^2_{q,h,w,c} - \sigma^2_{p^*,h,w,c})$$

Second Moment of Non-Training Distribution

- [ ] **CIFAR-10 vs SVHN** (plugging in empirical moments)
- [ ] **Asymmetry** (due to sub. being non-commutative)
- [ ] Uniform Inputs
- [ ] Ensembling
- [ ] Early Stopping

Analysis via Constant-Volume Flows

$$0 < \mathbb{E}_q[\log p(\boldsymbol{x};\boldsymbol{\theta})] - \mathbb{E}_{p^*}[\log p(\boldsymbol{x};\boldsymbol{\theta})]$$

Non-Training Distribution

Training Distribution

Second Moment of Training Distribution

$$\approx \underbrace{\left(-\frac{\partial^2}{\partial z^2}\log p(\boldsymbol{z};\boldsymbol{\psi})\right)}_{\text{(-)}} \sum_{c=1}^{C} \underbrace{\left(\prod_{k=1}^{K}\sum_{j=1}^{C} u_{k,c,j}\right)}_{\text{(+)}} \sum_{h,w} (\sigma^2_{q,h,w,c} - \sigma^2_{p^*,h,w,c})$$

Second Moment of Non-Training Distribution

- [ ] **CIFAR-10 vs SVHN** (plugging in empirical moments)
- [ ] **Asymmetry** (due to sub. being non-commutative)
- [ ] **Uniform Inputs**
- [ ] **Ensembling**
- [ ] **Early Stopping**

Analysis via Constant-Volume Flows

$$0 < \mathbb{E}_q[\log p(\boldsymbol{x};\boldsymbol{\theta})] - \mathbb{E}_{p^*}[\log p(\boldsymbol{x};\boldsymbol{\theta})]$$

Non-Training Distribution

Training Distribution

Second Moment of Training Distribution

$$\approx \left(-\frac{\partial^2}{\partial z^2}\log p(\boldsymbol{z};\boldsymbol{\psi})\right) \sum_{c=1}^{C} \left(\prod_{k=1}^{K}\sum_{j=1}^{C} u_{k,c,j}\right) \sum_{h,w} \left(-(\sigma^2_{q,h,w,c} - \sigma^2_{p^*,h,w,c})\right)$$

Second Moment of Non-Training Distribution

- [ ] **CIFAR-10 vs SVHN** (plugging in empirical moments)
- [ ] **Asymmetry** (due to sub. being non-commutative)
- [ ] **Uniform Inputs** (non-training 2nd moment is zero)
- [ ] **Ensembling**
- [ ] **Early Stopping**

Analysis via Constant-Volume Flows

$$0 < \mathbb{E}_q[\log p(\boldsymbol{x};\boldsymbol{\theta})] - \mathbb{E}_{p^*}[\log p(\boldsymbol{x};\boldsymbol{\theta})]$$

Non-Training Distribution

Training Distribution

Second Moment of Training Distribution

$$\approx \left(-\frac{\partial^2}{\partial z^2}\log p(\boldsymbol{z};\boldsymbol{\psi})\right) \sum_{c=1}^{C} \left(\prod_{k=1}^{K}\sum_{j=1}^{C} u_{k,c,j}\right) \sum_{h,w} (\sigma^2_{q,h,w,c} - \sigma^2_{p^*,h,w,c})$$

Second Moment of Non-Training Distribution

- ☐ **CIFAR-10 vs SVHN** (plugging in empirical moments)
- ☐ **Asymmetry** (due to sub. being non-commutative)
- ☐ **Uniform Inputs** (non-training 2nd moment is zero)
- ☐ **Ensembling**
- ☐ **Early Stopping**

} (sign doesn't depend on model param. values)

Analysis via Constant-Volume Flows

$$0 < \mathbb{E}_q[\log p(\boldsymbol{x};\boldsymbol{\theta})] - \mathbb{E}_{p^*}[\log p(\boldsymbol{x};\boldsymbol{\theta})]$$

Non-Training Distribution

Training Distribution

Second Moment of Training Distribution

$$\approx \left(-\frac{\partial^2}{\partial z^2}\log p(\boldsymbol{z};\boldsymbol{\psi})\right) \sum_{c=1}^{C} \left(\prod_{k=1}^{K}\sum_{j=1}^{C} u_{k,c,j}\right) \sum_{h,w} (\sigma^2_{q,h,w,c} - \sigma^2_{p^*,h,w,c})$$

Second Moment of Non-Training Distribution

Hypothesis: If the second-order statistics do indeed dominate, we should be able to control the likelihoods by **graying** the images…

Analysis via Constant-Volume Flows

$$0 < \mathbb{E}_q[\log p(\boldsymbol{x};\boldsymbol{\theta})] - \mathbb{E}_{p^*}[\log p(\boldsymbol{x};\boldsymbol{\theta})]$$

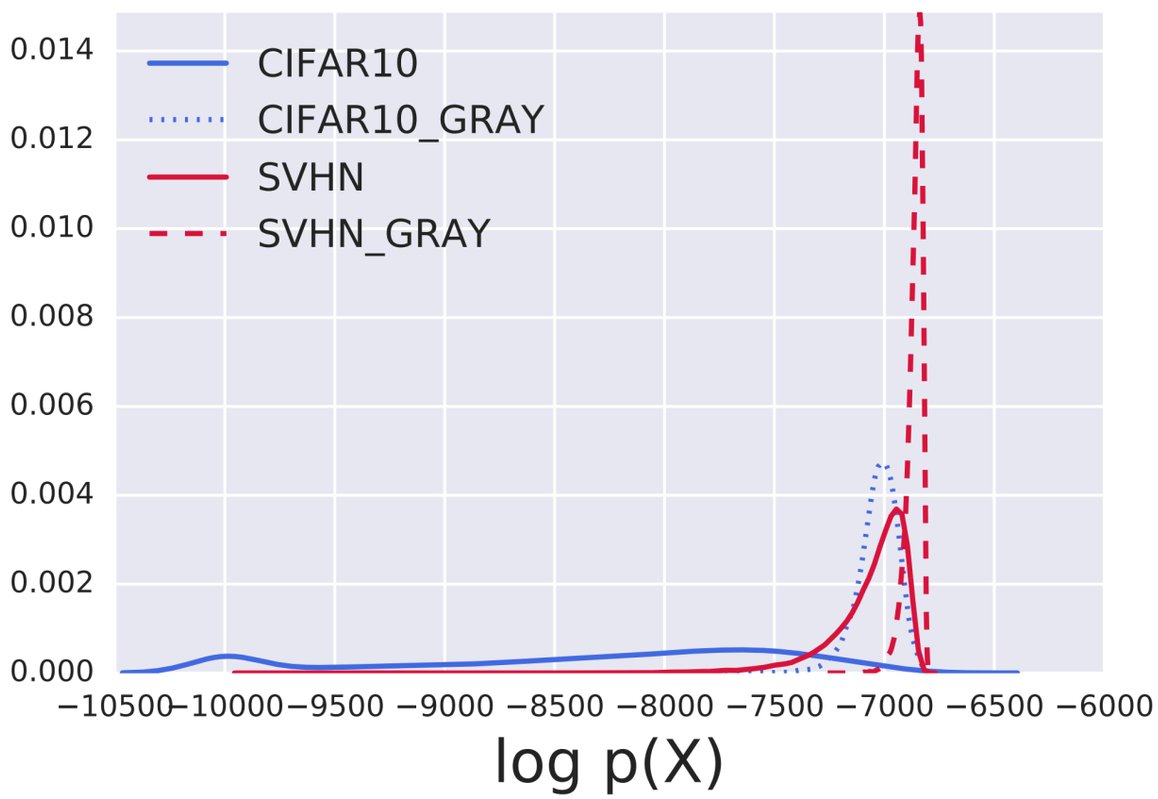
Non-Training Distribution

Training Distribution

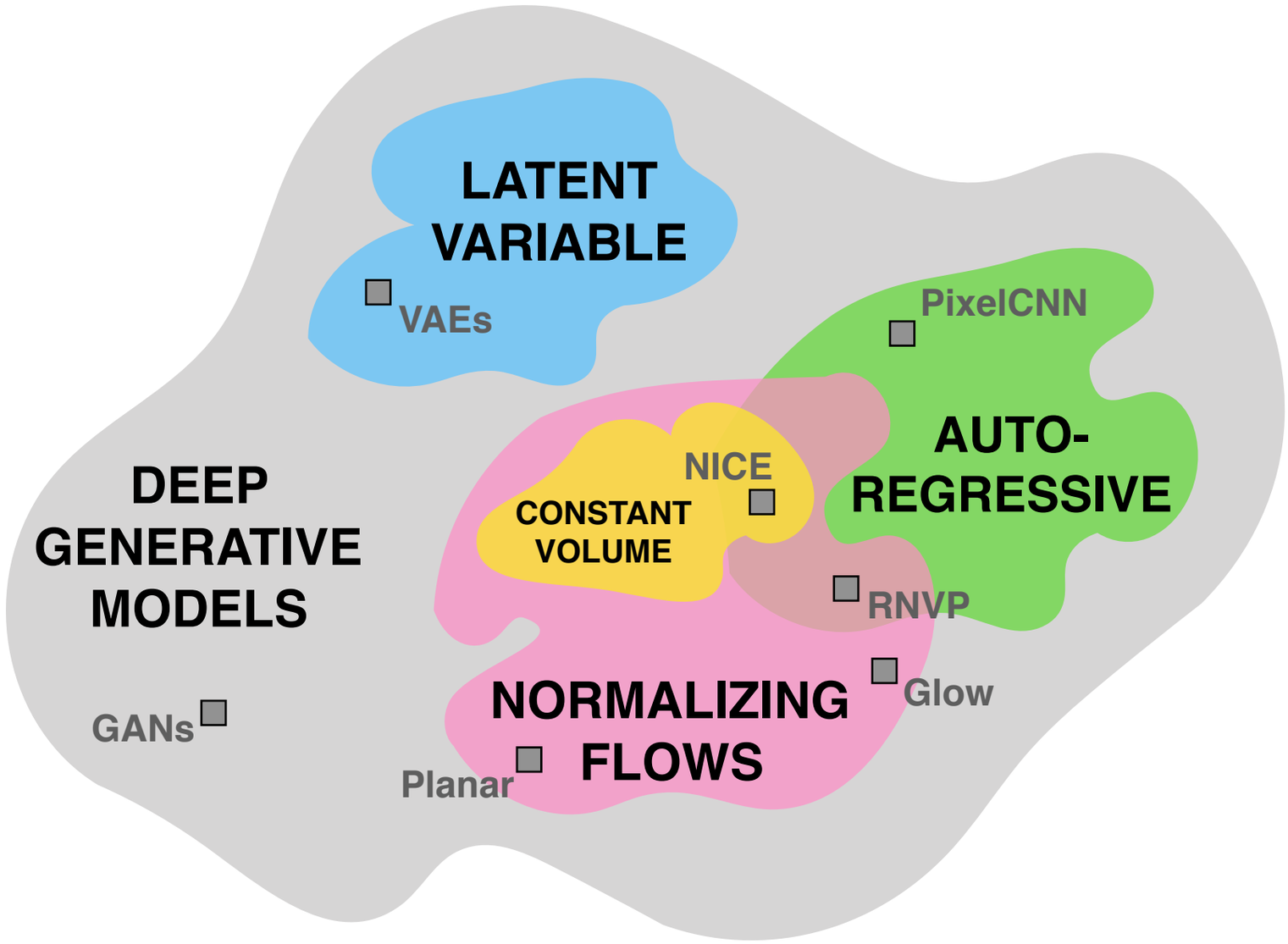
Second Moment of Training Distribution

$$\approx \left(-\frac{\partial^2}{\partial z^2}\log p(\boldsymbol{z};\boldsymbol{\psi})\right) \sum_{c=1}^{C} \left(\prod_{k=1}^{K}\sum_{j=1}^{C} u_{k,c,j}\right) \sum_{h,w} (\sigma^2_{q,h,w,c} - \sigma^2_{p^*,h,w,c})$$

Second Moment of Non-Training Distribution

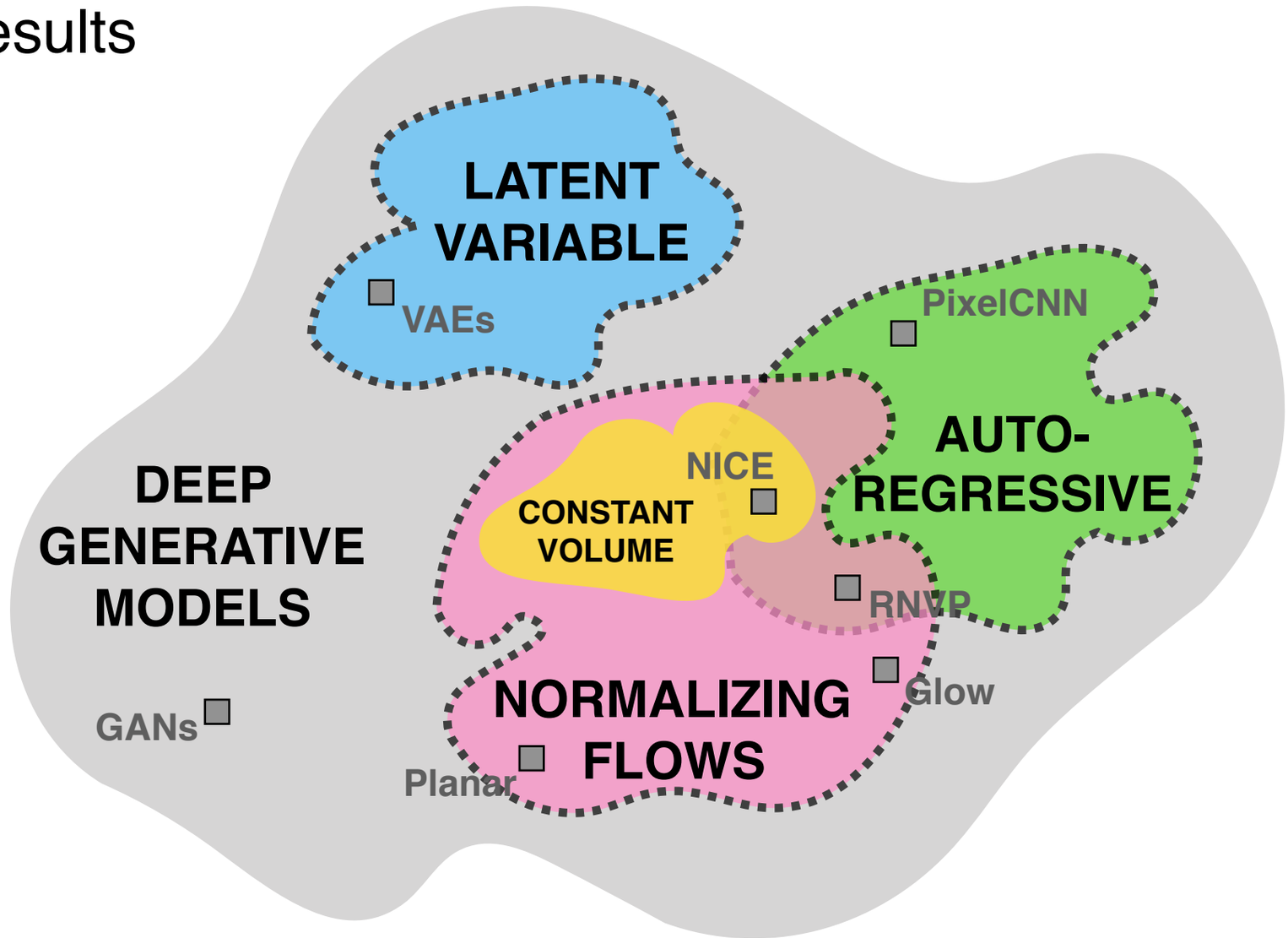


Summary of Results



Summary of Results

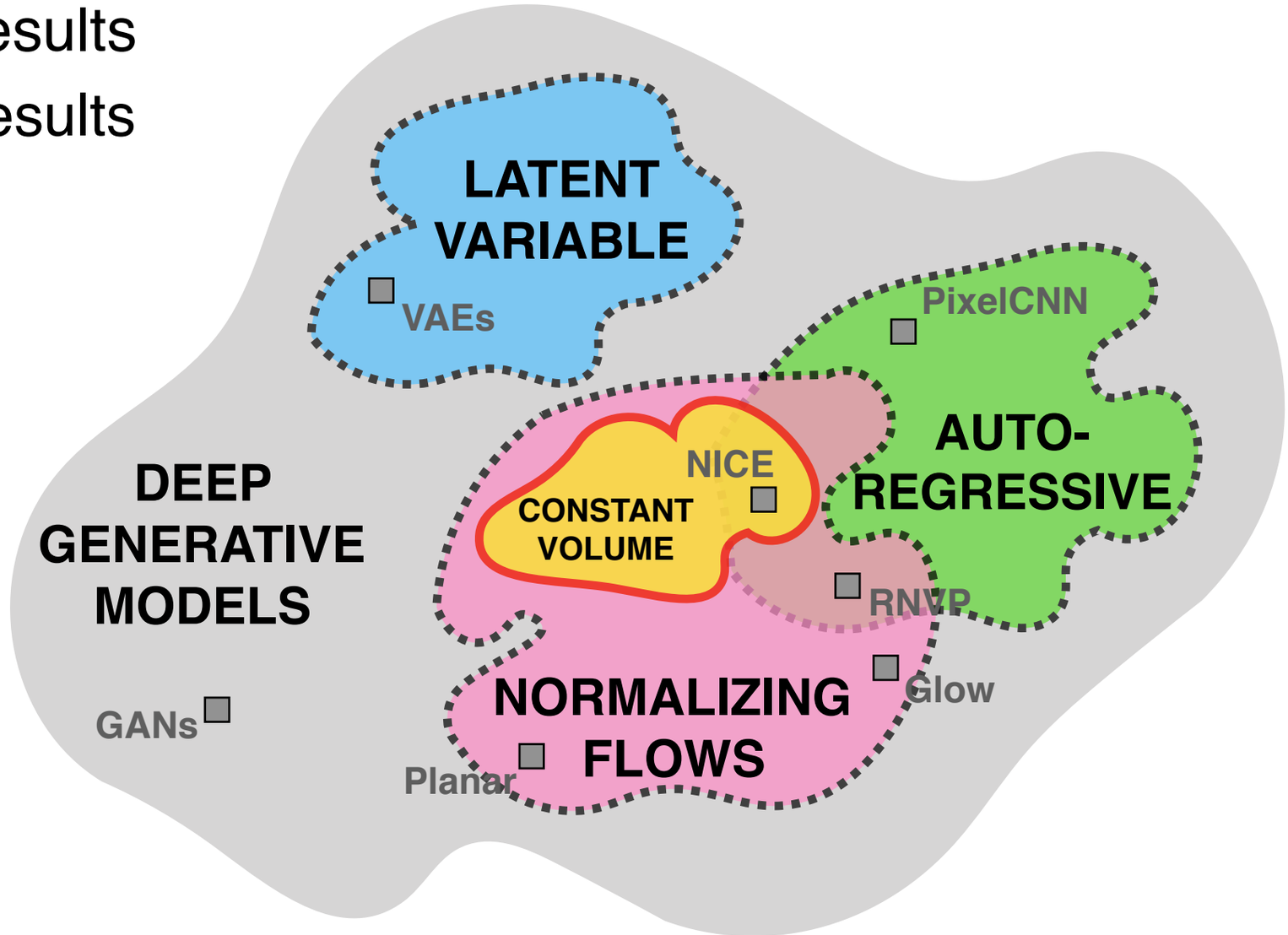
- Empirical Results



Summary of Results

Empirical Results

Analytical Results



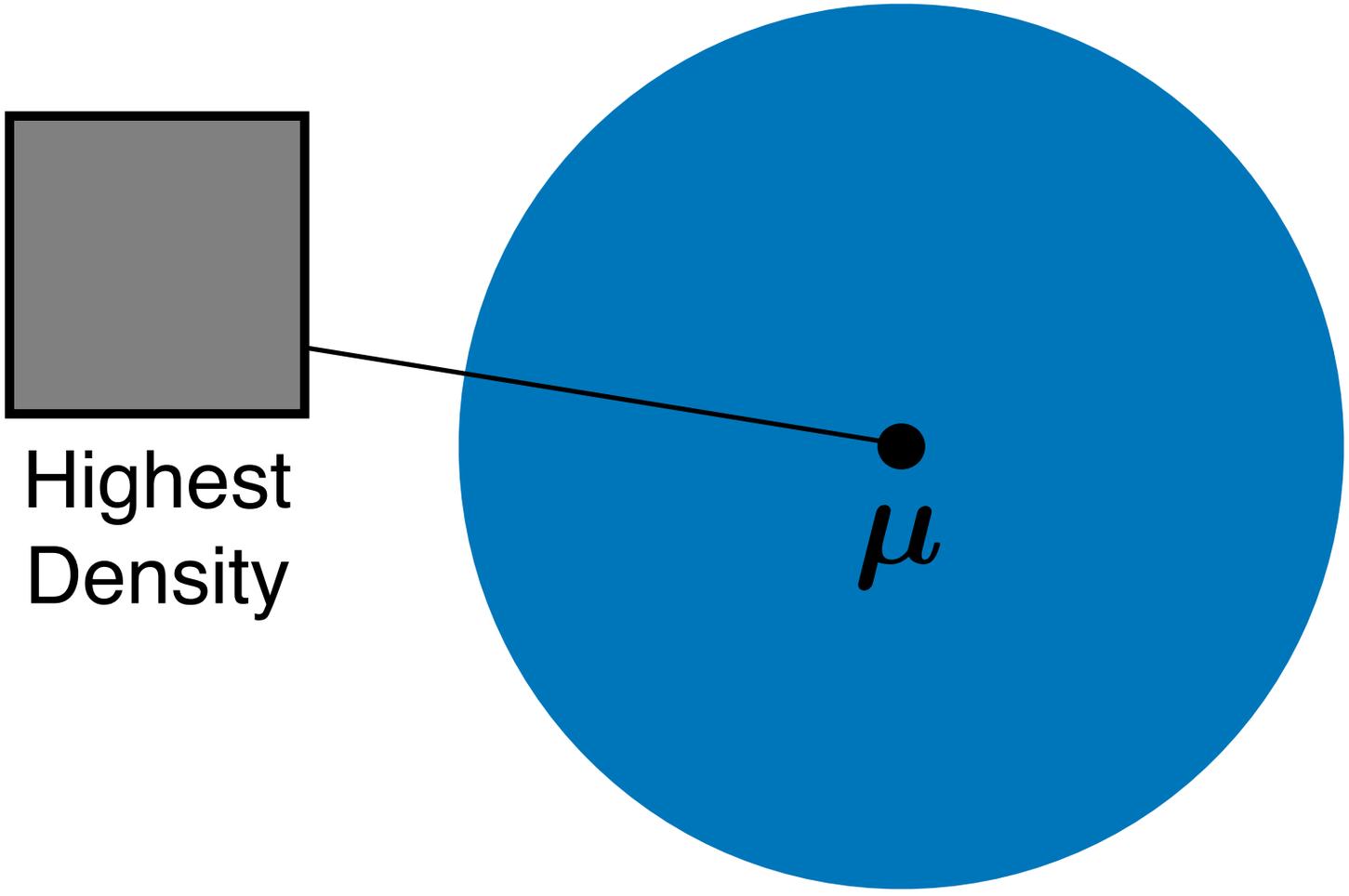
PART #4

Concentration and Typicality

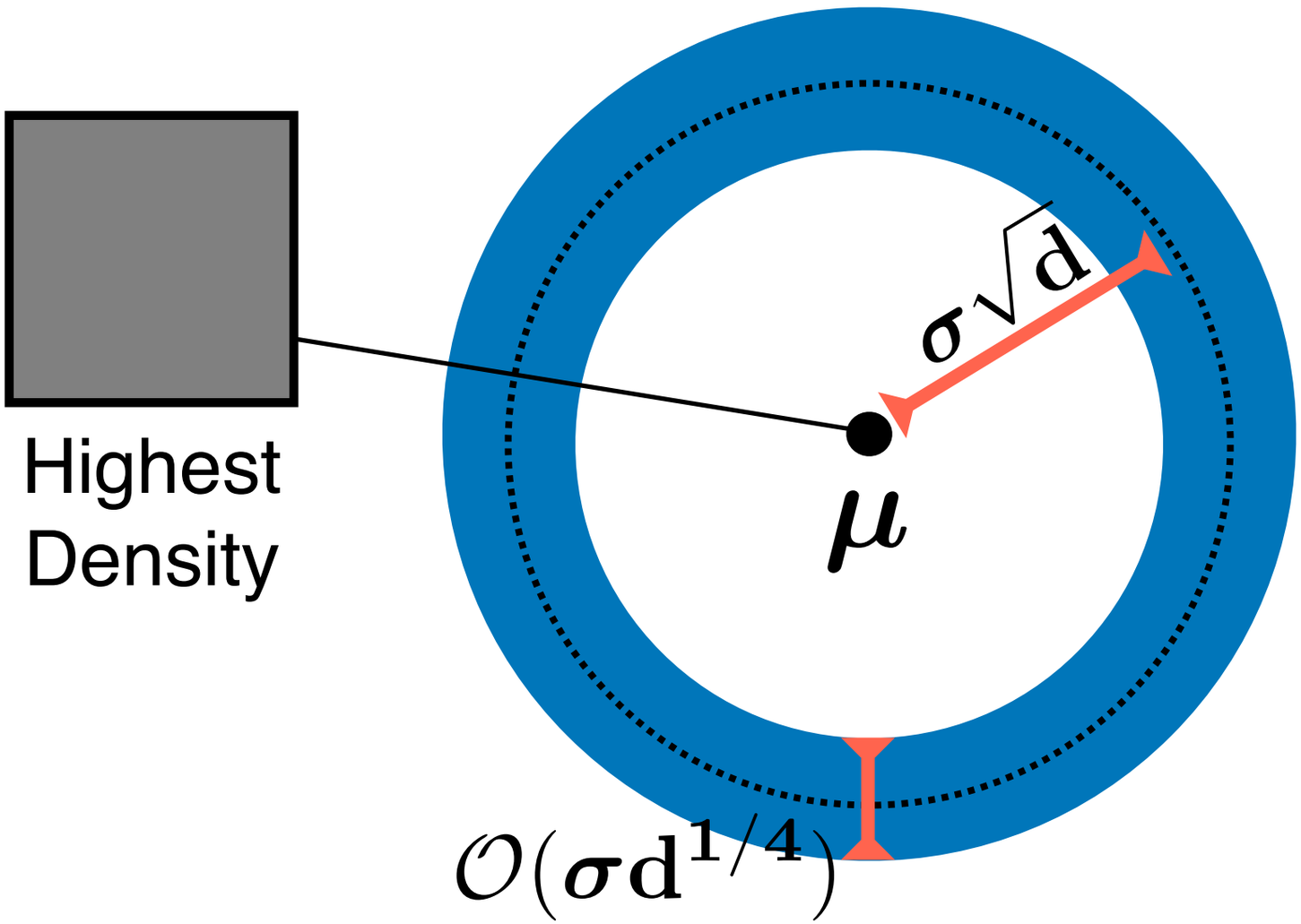
Question: In the CIFAR vs SVHN case, why don't we ever see samples from SVHN?

Samples from Glow when trained on CIFAR-10

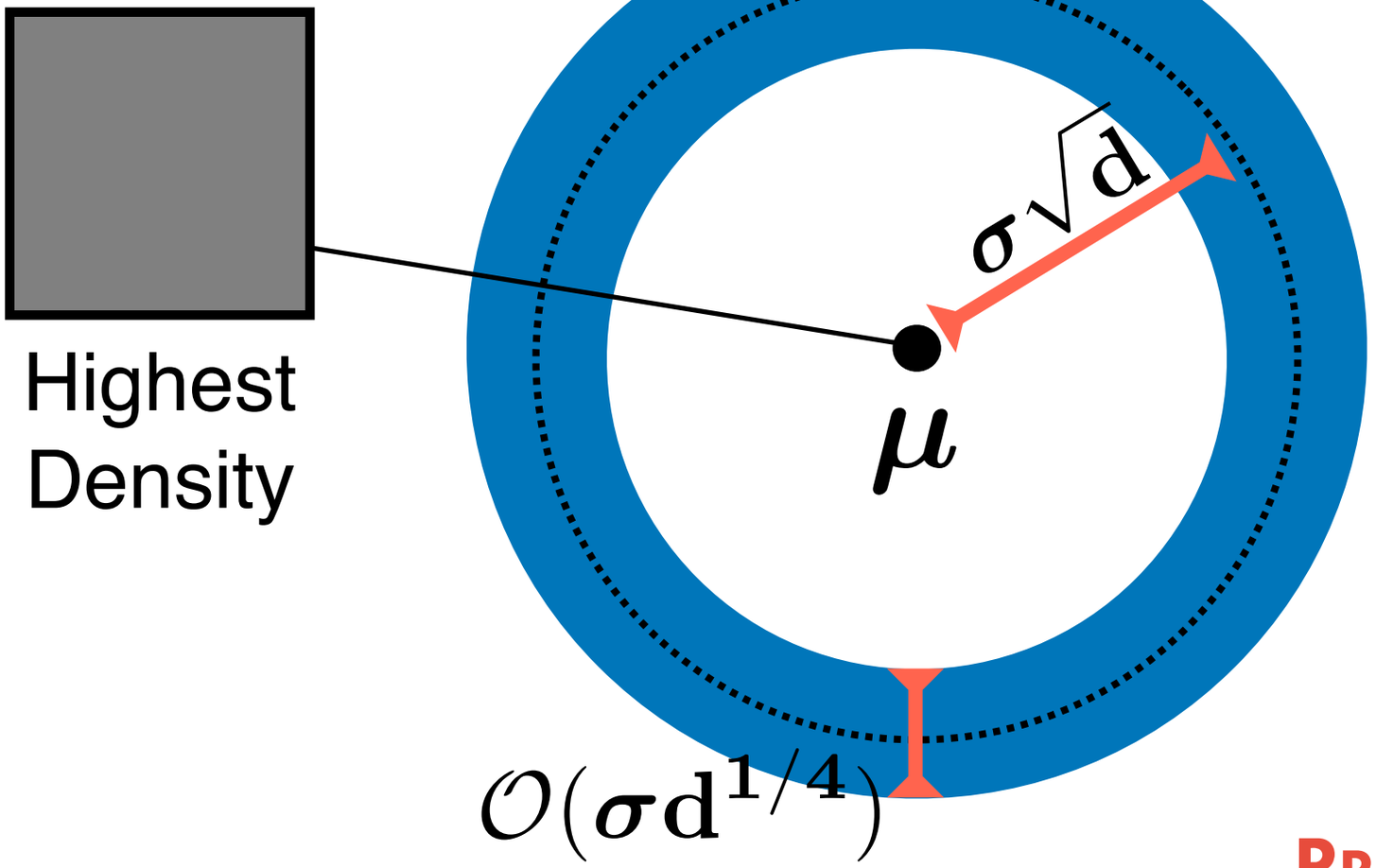
Back to basics: consider a high-dimensional Gaussian centered on the all-gray image…

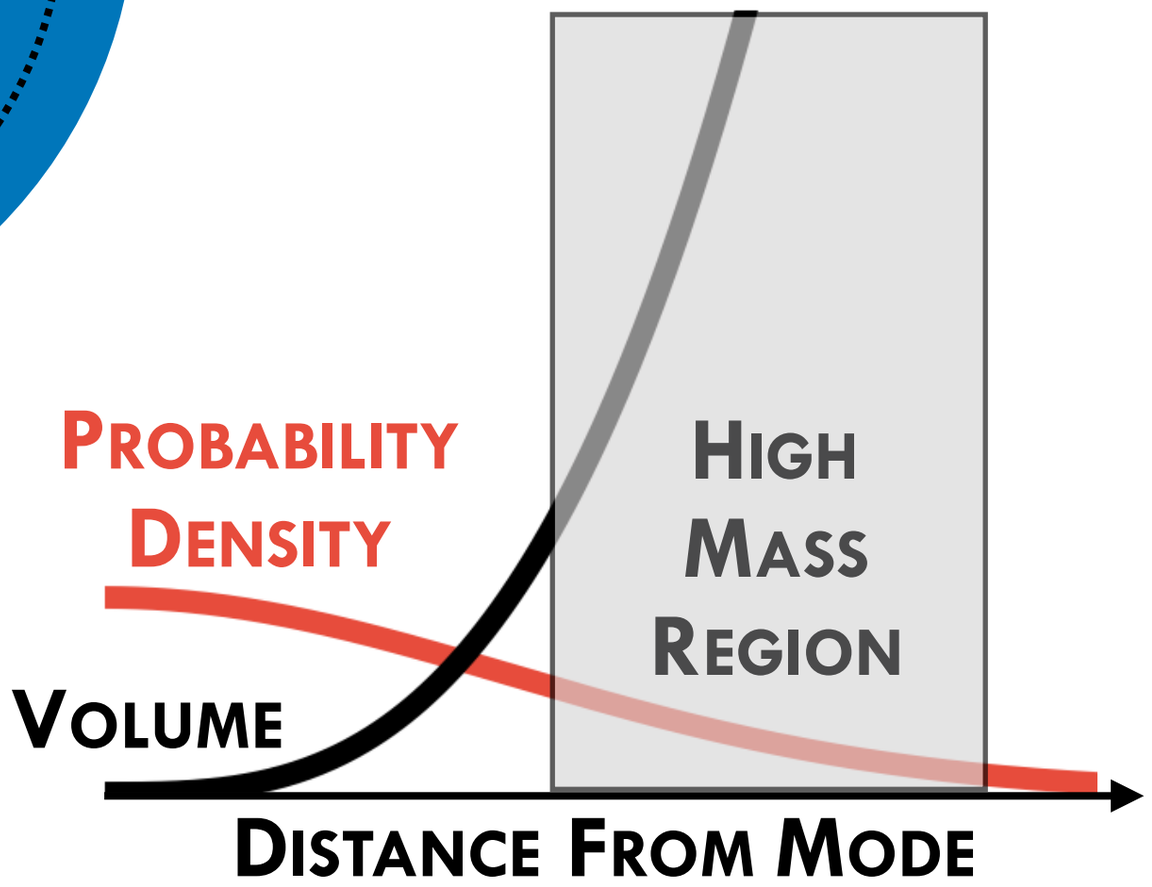


Back to basics: consider a high-dimensional Gaussian centered on the all-gray image…



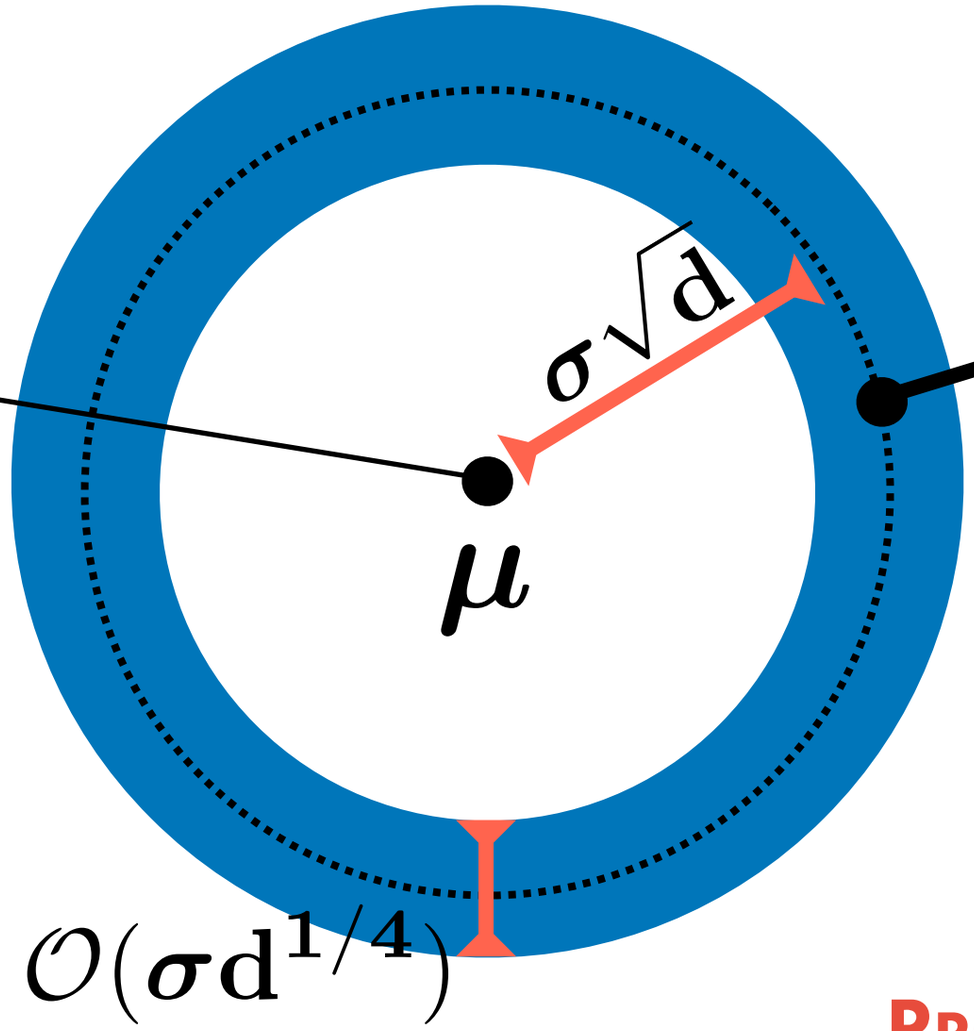
Back to basics: consider a high-dimensional Gaussian centered on the all-gray image…



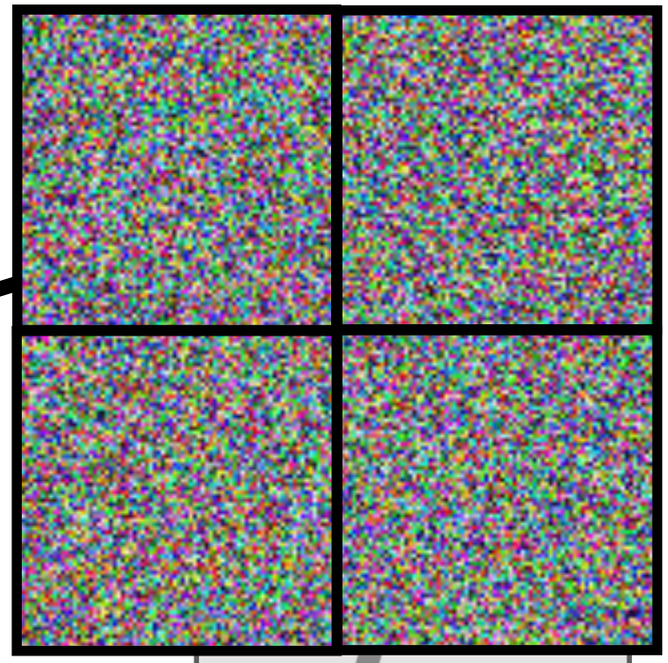


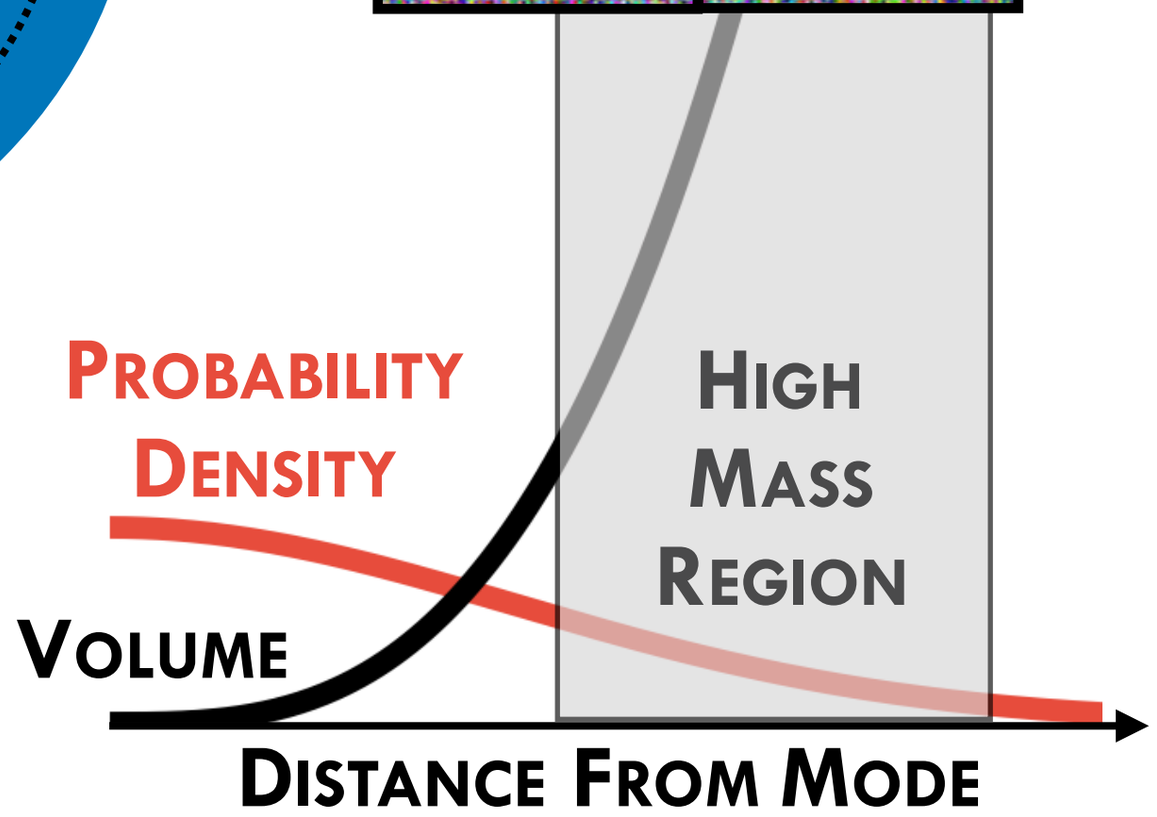
Back to basics: consider a high-dimensional Gaussian centered on the all-gray image…

Highest Density



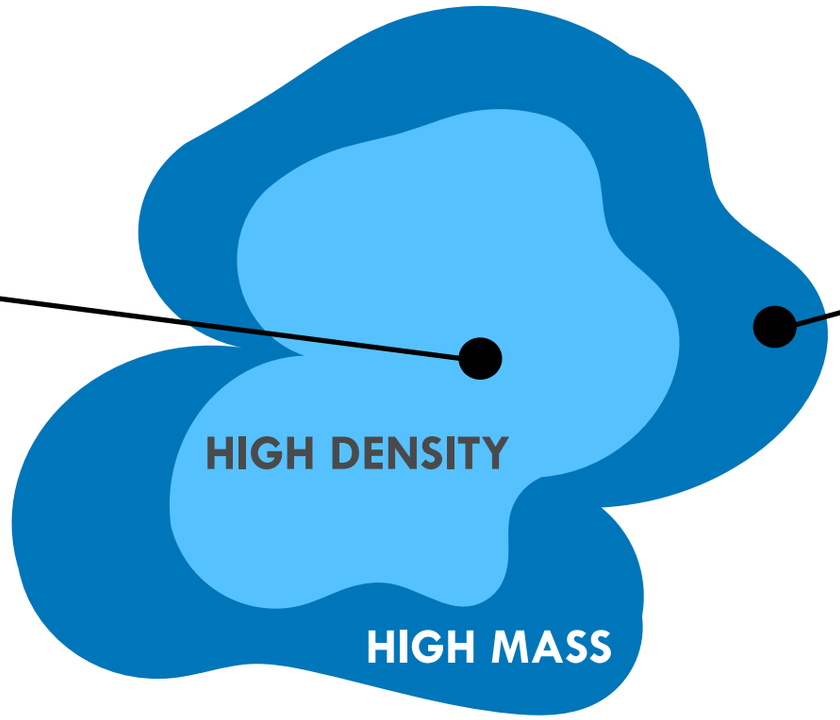
Samples





Conjecture: We are probably just seeing the same concentration phenomenon in deep generative models…

High Density



High Probability
(Samples)

FashionMNIST train vs **MNIST** test

Thus we should be checking for out-of-distribution inputs **not via density but via mass**…

Thus we should be checking for out-of-distribution inputs **not via density but via mass**…

But how do we do this?…

Thus we should be checking for out-of-distribution inputs **not via density but via mass**…

But how do we do this?…

Idea #1: Integrate the density function.

No surprise that the canonical goodness-of-fit tests use the model CDF (e.g. Kolmogorov–Smirnov, Anderson–Darling, etc.).

Thus we should be checking for out-of-distribution inputs **not via density but via mass**…

But how do we do this?…

~~Idea #1: Integrate the density function.~~

No surprise that the canonical goodness-of-fit tests use the model CDF (e.g. Kolmogorov–Smirnov, Anderson–Darling, etc.).

Thus we should be checking for out-of-distribution inputs **not via density but via mass**…

But how do we do this?…

~~Idea #1: Integrate the density function.~~

~~*No surprise that the canonical goodness-of-fit tests use the model CDF (e.g. Kolmogorov–Smirnov, Anderson–Darling, etc.).*~~

Idea #2: Check for *typicality.*

Typical Sets

Definition 2.1. ϵ-Typical Set [11] *For a distribution $p(\mathbf{x})$ with support $\mathbf{x} \in \mathcal{X}$, the ϵ-typical set $\mathcal{A}_\epsilon^N[p(\mathbf{x})] \in \mathcal{X}^N$ is comprised of all N-length sequences that satisfy*

$$\mathbb{H}[p(\mathbf{x})] - \epsilon \leq \frac{-1}{N} \sum_{n=1}^{N} \log p(\boldsymbol{x}_n) \leq \mathbb{H}[p(\mathbf{x})] + \epsilon$$

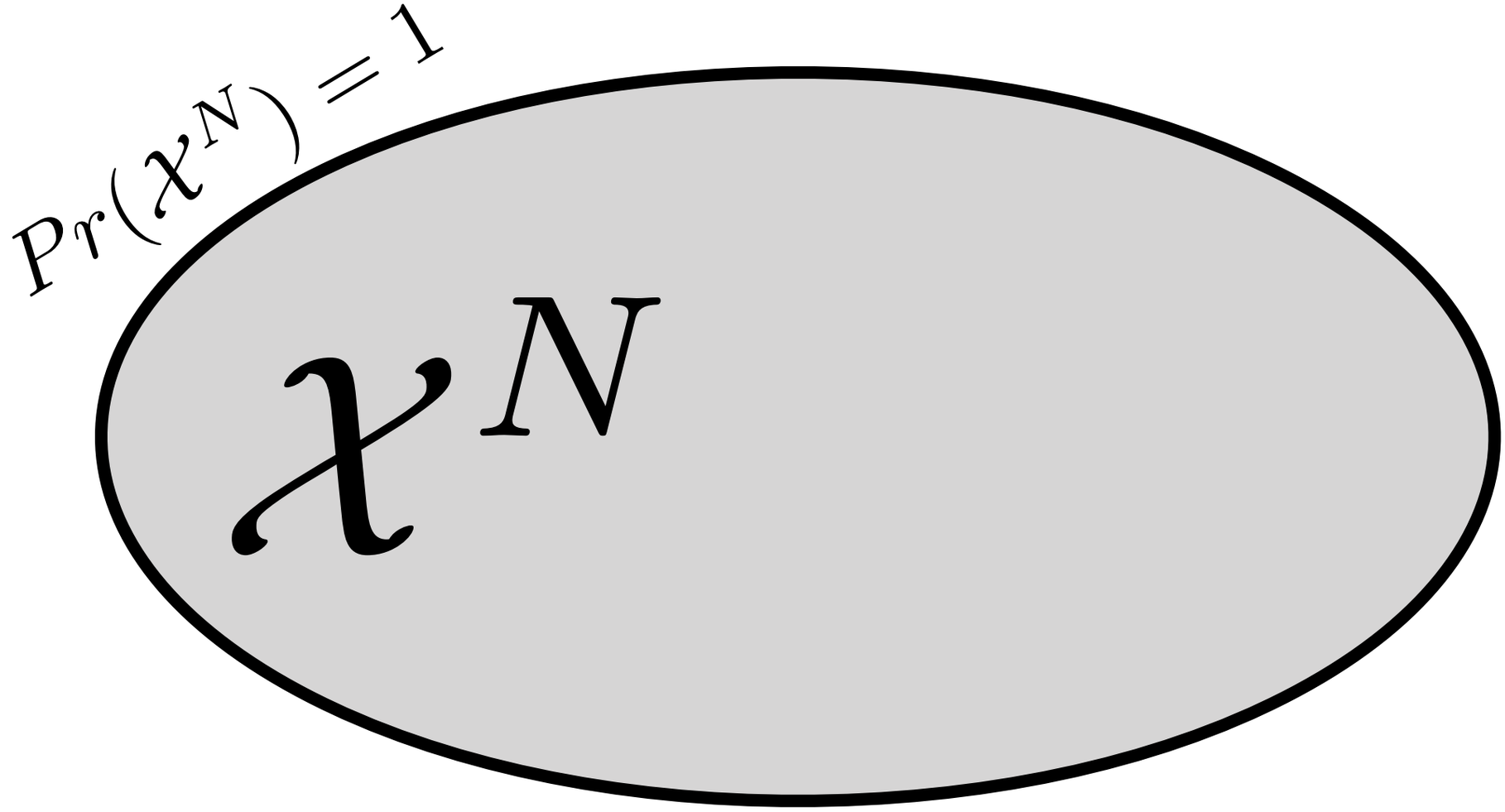
where $\mathbb{H}[p(\mathbf{x})] = \int_{\mathcal{X}} p(\mathbf{x})[-\log p(\mathbf{x})]d\mathbf{x}$ and $\epsilon \in \mathbb{R}^+$ is a small constant.

Typical Sets

Definition 2.1. ϵ-Typical Set [11] *For a distribution $p(\mathbf{x})$ with support $\mathbf{x} \in \mathcal{X}$, the ϵ-typical set $\mathcal{A}_\epsilon^N[p(\mathbf{x})] \in \mathcal{X}^N$ is comprised of all N-length sequences that satisfy*

$$\mathbb{H}[p(\mathbf{x})] - \epsilon \leq \frac{-1}{N} \sum_{n=1}^{N} \log p(\boldsymbol{x}_n) \leq \mathbb{H}[p(\mathbf{x})] + \epsilon$$

where $\mathbb{H}[p(\mathbf{x})] = \int_{\mathcal{X}} p(\mathbf{x})[-\log p(\mathbf{x})]d\mathbf{x}$ and $\epsilon \in \mathbb{R}^+$ is a small constant.

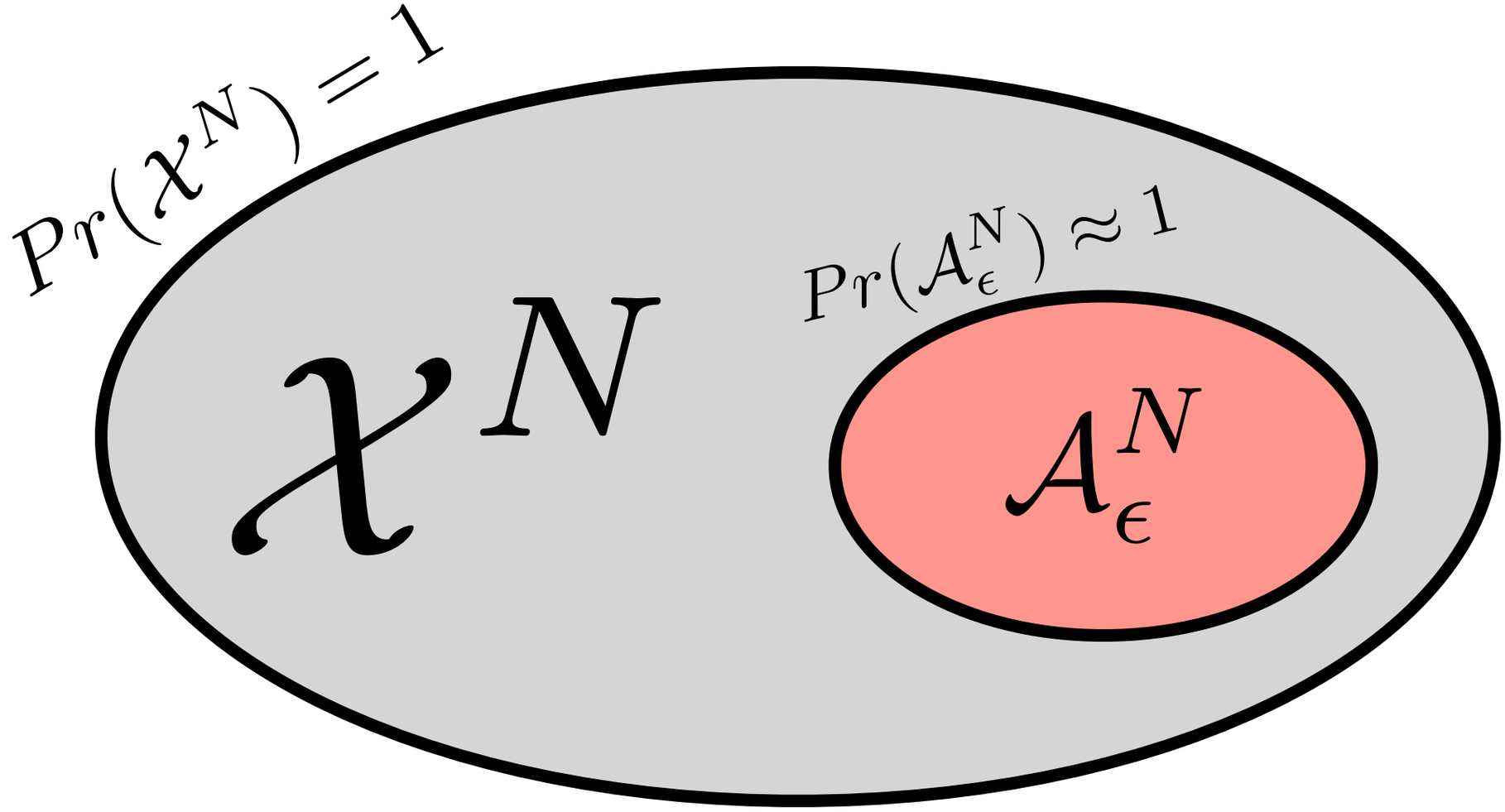


Typical Sets

Definition 2.1. ϵ-Typical Set [11] *For a distribution $p(\mathbf{x})$ with support $\mathbf{x} \in \mathcal{X}$, the ϵ-typical set $\mathcal{A}_\epsilon^N[p(\mathbf{x})] \in \mathcal{X}^N$ is comprised of all N-length sequences that satisfy*

$$\mathbb{H}[p(\mathbf{x})] - \epsilon \leq \frac{-1}{N} \sum_{n=1}^{N} \log p(\boldsymbol{x}_n) \leq \mathbb{H}[p(\mathbf{x})] + \epsilon$$

where $\mathbb{H}[p(\mathbf{x})] = \int_{\mathcal{X}} p(\mathbf{x})[-\log p(\mathbf{x})]d\mathbf{x}$ and $\epsilon \in \mathbb{R}^+$ is a small constant.



Using Typicality to Assess Model Fit

For an *M*-sized test batch

$$\widetilde{\boldsymbol{X}} = \{\tilde{\boldsymbol{x}}_1, \ldots, \tilde{\boldsymbol{x}}_M\}$$

$$\texttt{if } \widetilde{\boldsymbol{X}} \in \mathcal{A}_\epsilon^M[p(\mathbf{x};\boldsymbol{\theta})] \texttt{ then } \widetilde{\boldsymbol{X}} \sim p(\mathbf{x};\boldsymbol{\theta}),$$

$$\texttt{otherwise } \widetilde{\boldsymbol{X}} \not\sim p(\mathbf{x};\boldsymbol{\theta})$$

Using Typicality to Assess Model Fit

For an *M*-sized test batch

$$\widetilde{\boldsymbol{X}} = \{\tilde{\boldsymbol{x}}_1, \ldots, \tilde{\boldsymbol{x}}_M\}$$

$$\texttt{if } \widetilde{\boldsymbol{X}} \in \mathcal{A}_\epsilon^M[p(\mathbf{x};\boldsymbol{\theta})] \texttt{ then } \widetilde{\boldsymbol{X}} \sim p(\mathbf{x};\boldsymbol{\theta}),$$

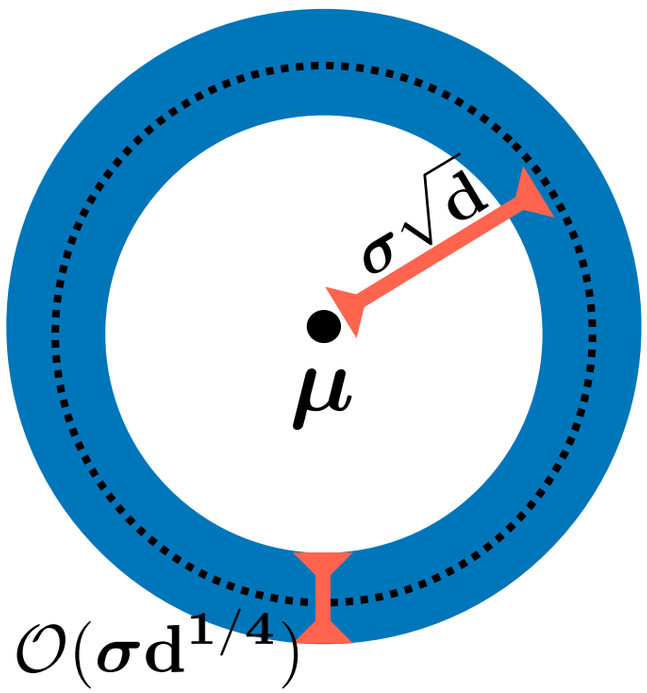
$$\texttt{otherwise } \widetilde{\boldsymbol{X}} \not\sim p(\mathbf{x};\boldsymbol{\theta})$$

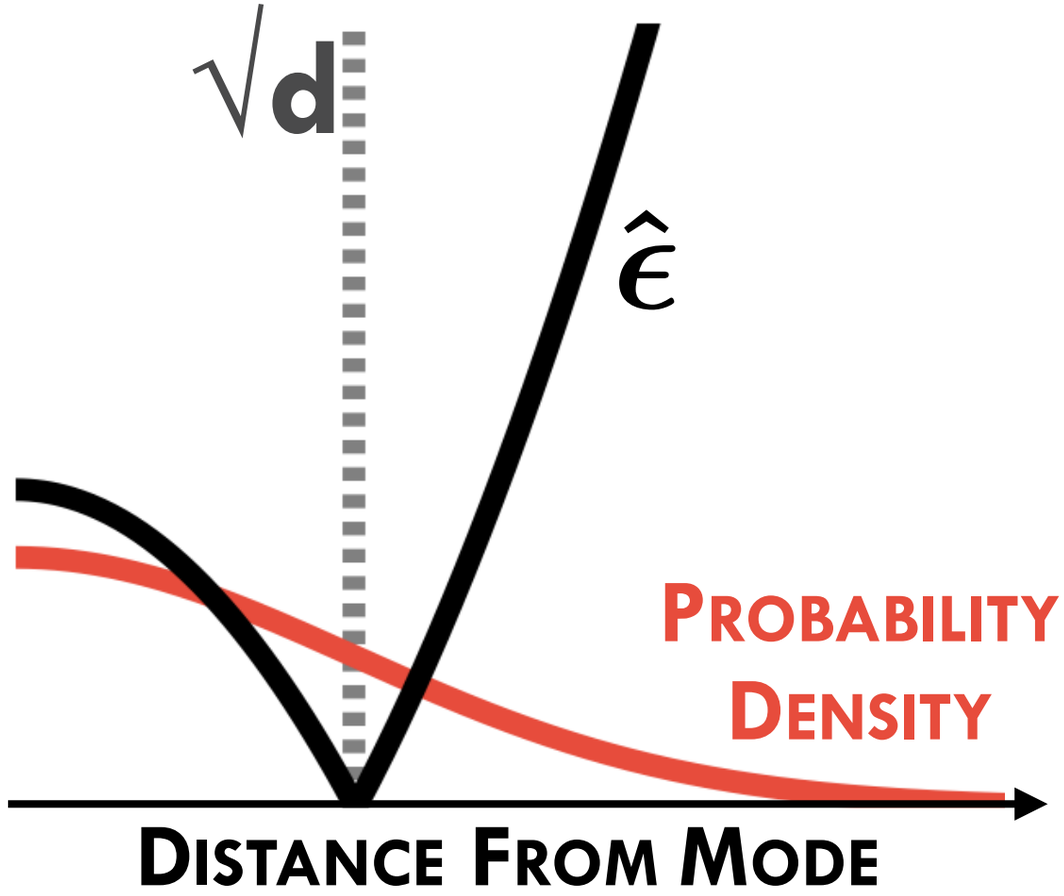
Plugging in the entropy bound…

$$\texttt{if } \left| \frac{-1}{M} \sum_{m=1}^{M} \log p(\tilde{\boldsymbol{x}}_m;\boldsymbol{\theta}) - \mathbb{H}[p(\mathbf{x};\boldsymbol{\theta})] \right| \leq \epsilon \texttt{ then } \widetilde{\boldsymbol{X}} \in \mathcal{A}_\epsilon^M[p(\mathbf{x};\boldsymbol{\theta})]$$

Gaussian Simulation

$$\left| \frac{-1}{M} \sum_{m=1}^{M} \log p(\tilde{\boldsymbol{x}}_m; \boldsymbol{\theta}) - \mathbb{H}[p(\mathbf{x}; \boldsymbol{\theta})] \right| \leq \epsilon$$





Implementation via Bootstrap CI

How should we set **ε** in practice?

Implementation via Bootstrap CI

How should we set **ε** in practice?

We construct the test statistic via a bootstrap confidence interval and validation data…

Implementation via Bootstrap CI

How should we set **ε** in practice?

We construct the test statistic via a bootstrap confidence interval and validation data…

$$H_0 : \boldsymbol{X} \sim \mathcal{A}_{\epsilon}^{M}[p(\mathbf{x};\boldsymbol{\theta})]$$

Implementation via Bootstrap CI

How should we set **ε** in practice?

We construct the test statistic via a bootstrap confidence interval and validation data…

$$H_0 : \boldsymbol{X} \in \mathcal{A}_\epsilon^M[p(\mathbf{x};\boldsymbol{\theta})]$$

$$\left| \frac{-1}{M} \sum_{m=1}^{M} \log p(\boldsymbol{x}'_{k,m};\boldsymbol{\theta}) - \frac{-1}{N} \sum_{n=1}^{N} \log p(\boldsymbol{x}_n;\boldsymbol{\theta}) \right| = \hat{\epsilon}_k$$

Implementation via Bootstrap CI

How should we set **ε** in practice?

We construct the test statistic via a bootstrap confidence interval and validation data…

$$H_0 : \boldsymbol{X} \in \mathcal{A}_\epsilon^M[p(\mathbf{x};\boldsymbol{\theta})]$$

$$\left| \frac{-1}{M} \sum_{m=1}^{M} \log p(\boldsymbol{x}'_{k,m};\boldsymbol{\theta}) - \frac{-1}{N} \sum_{n=1}^{N} \log p(\boldsymbol{x}_n;\boldsymbol{\theta}) \right| = \hat{\epsilon}_k$$

$$F(\epsilon) = \frac{1}{K} \sum_{k=1}^{K} \delta[\hat{\epsilon}_k]$$

Implementation via Bootstrap CI

Algorithm 1 A Bootstrap Test for Typicality

Input: Training data $\boldsymbol{X}$, validation data $\boldsymbol{X}'$, trained model $p(\mathbf{x};\boldsymbol{\theta})$, number of bootstrap samples K, significance level α, M-sized batch of possibly OOD inputs $\widetilde{\boldsymbol{X}}$.

Offline prior to deployment

1. Compute $\hat{\mathbb{H}}^N[p(\mathbf{x};\boldsymbol{\theta})] = \frac{-1}{N}\sum_{n=1}^{N}\log p(\boldsymbol{x}_n;\boldsymbol{\theta})$.

2. Sample K M-sized data sets from $\mathbf{X}'$ using bootstrap resampling.

3. For all $k \in [1, K]$:

Compute $\hat{\epsilon}_k = \left|\frac{-1}{M}\sum_{m=1}^{M}\log p(\boldsymbol{x}'_{k,m};\boldsymbol{\theta}) - \hat{\mathbb{H}}^N[p(\mathbf{x};\boldsymbol{\theta})]\right|$ *(Equation 7)*

4. Set $\epsilon_\alpha^M = \texttt{quantile}(F(\epsilon), \alpha)$ *(e.g.* $\alpha = .99$*)*

Online during deployment

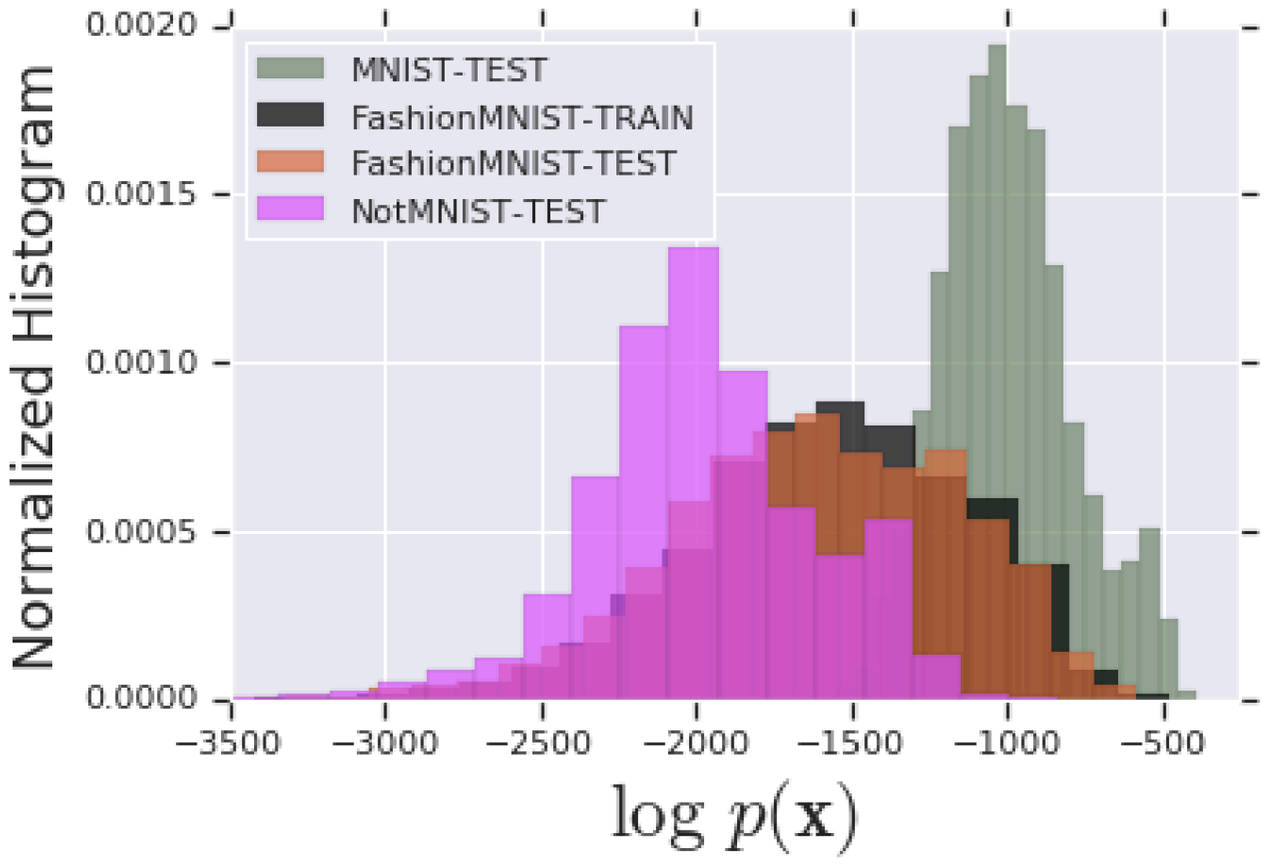
If $\left|\frac{-1}{M}\sum_{m=1}^{M}\log p(\tilde{\boldsymbol{x}}_m) - \hat{\mathbb{H}}^N[p(\mathbf{x};\boldsymbol{\theta})]\right| > \epsilon_\alpha^M$:

Return $\widetilde{\mathbf{X}}$ `is out-of-distribution`

Else:

Return $\widetilde{\mathbf{X}}$ `is in-distribution`

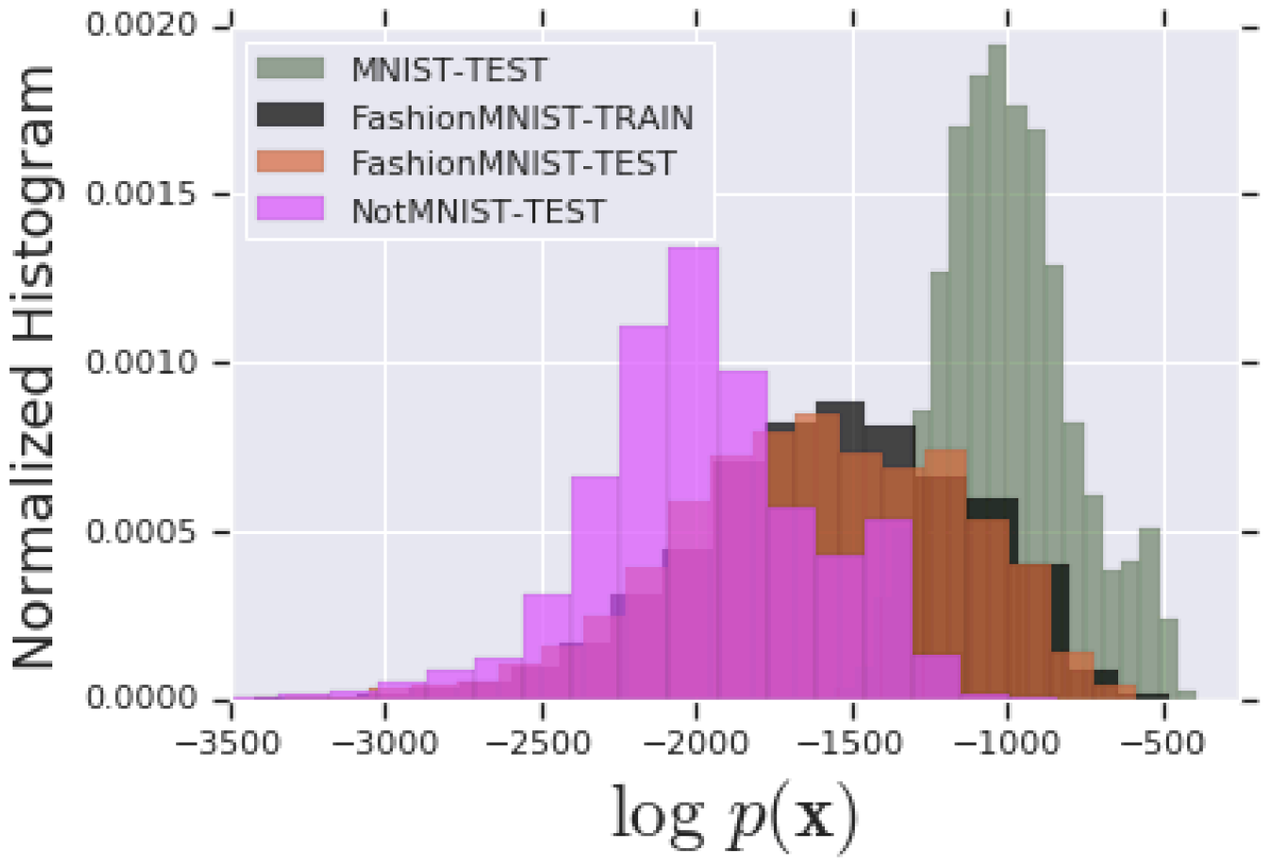
Results: Glow Trained on FashionMNIST



	M = 2			M = 10			M = 25		
METHOD	IN-DIST.	MNIST	NOTMNIST	IN-DIST.	MNIST	NOTMNIST	IN-DIST.	MNIST	NOTMNIST
Glow Trained on FashionMNIST									
Typicality Test									
KS-Test									
Max Mean Dis.									
Kern. Stein Dis.									

Fraction of M-sized batches classified as OOD (α = 99%)

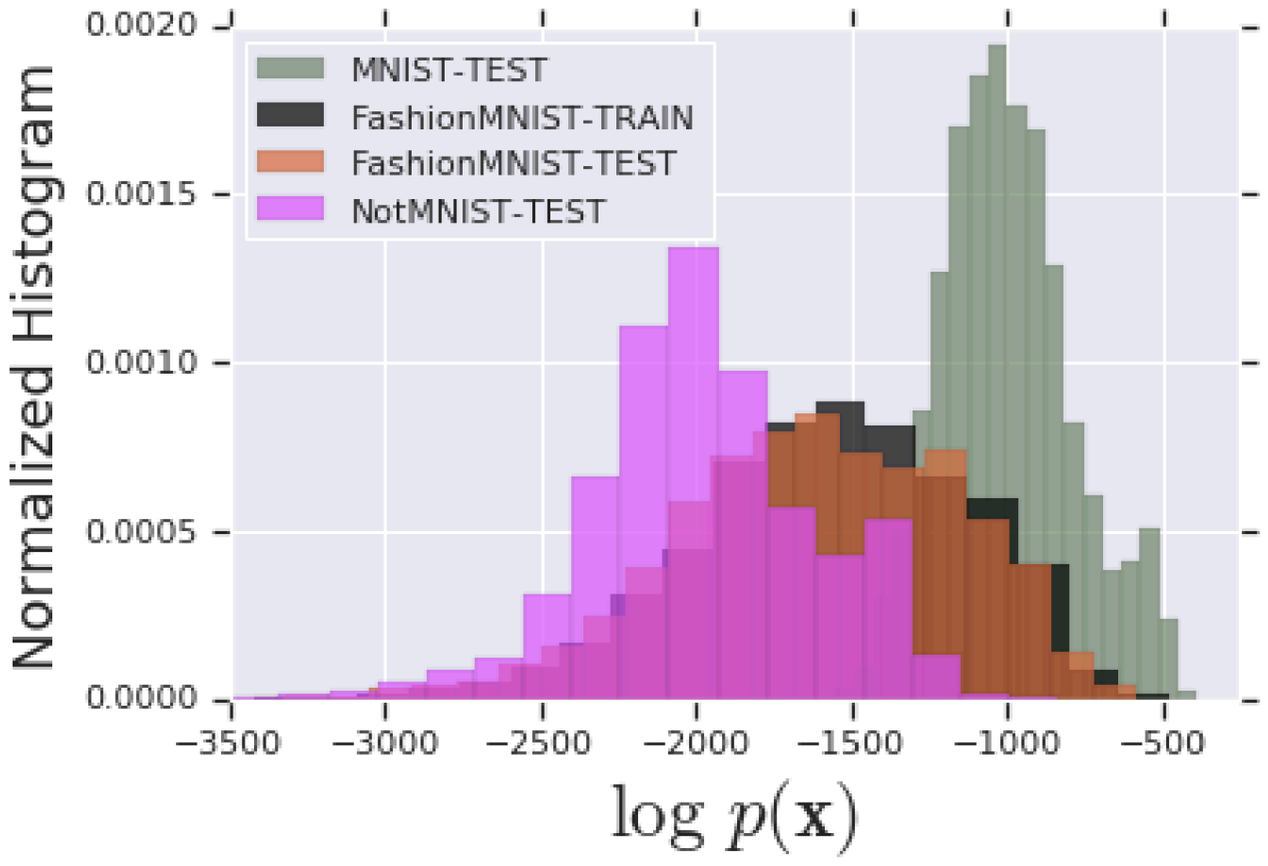
Results: Glow Trained on FashionMNIST



	M = 2			M = 10			M = 25		
METHOD	IN-DIST.	MNIST	NOTMNIST	IN-DIST.	MNIST	NOTMNIST	IN-DIST.	MNIST	NOTMNIST
Glow Trained on FashionMNIST									
Typicality Test							0.01±.00	**1.00**±.00	**1.00**±.00
KS-Test							**0.00**±.00	**1.00**±.00	0.98±.01
Max Mean Dis.							0.04±.04	**1.00**±.00	**1.00**±.00
Kern. Stein Dis.							0.02±.03	0.76±.21	0.00±.00

Fraction of M-sized batches classified as OOD (α = 99%)

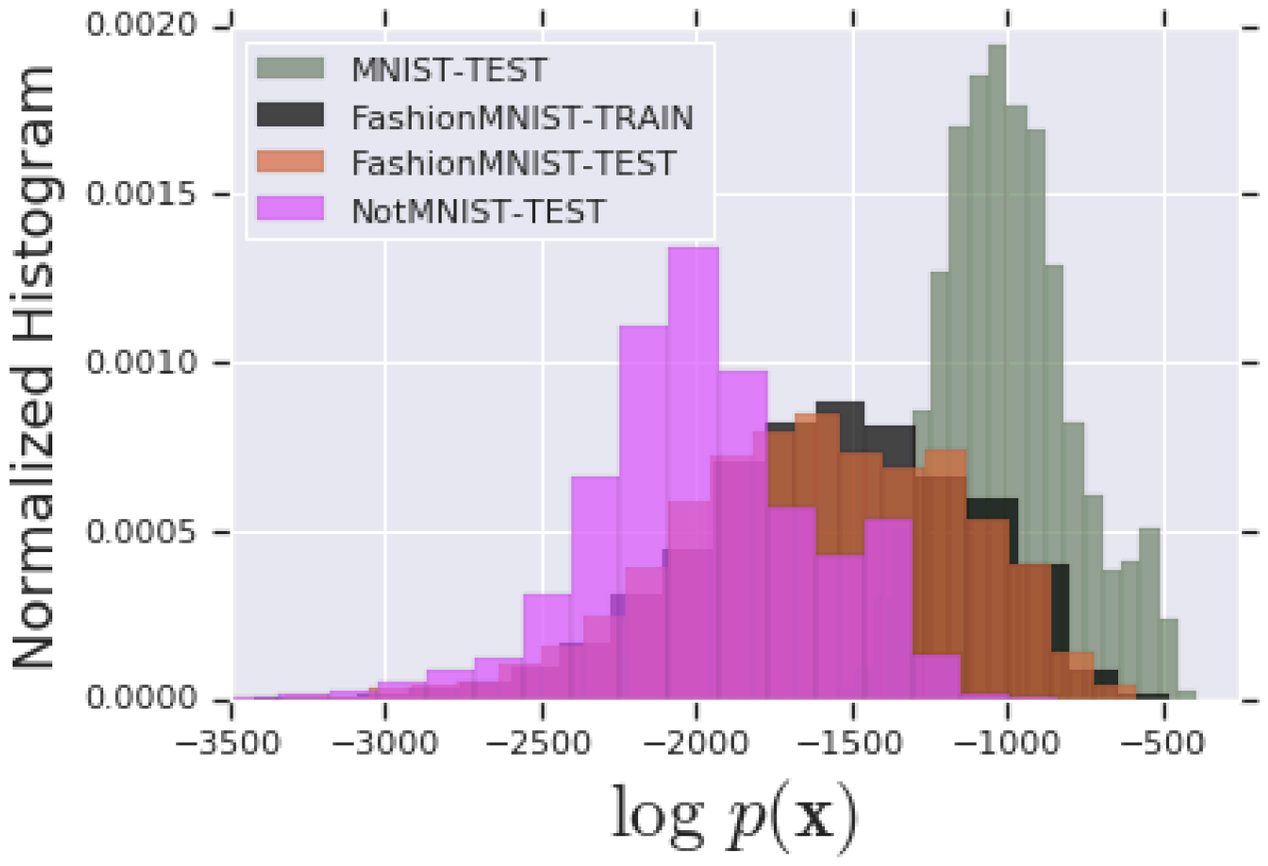
Results: Glow Trained on FashionMNIST



	M = 2			M = 10			M = 25		
METHOD	IN-DIST.	MNIST	NOTMNIST	IN-DIST.	MNIST	NOTMNIST	IN-DIST.	MNIST	NOTMNIST
Glow Trained on FashionMNIST									
Typicality Test				0.02±.02	**1.00**±.00	**0.69**±.11	0.01±.00	**1.00**±.00	**1.00**±.00
KS-Test				**0.01**±.00	**1.00**±.00	0.61±.01	**0.00**±.00	**1.00**±.00	0.98±.01
Max Mean Dis.				0.02±.02	0.63±.12	0.37±.24	0.04±.04	**1.00**±.00	**1.00**±.00
Kern. Stein Dis.				**0.01**±.01	0.21±.11	0.01±.00	0.02±.03	0.76±.21	0.00±.00

Fraction of M-sized batches classified as OOD (α = 99%)

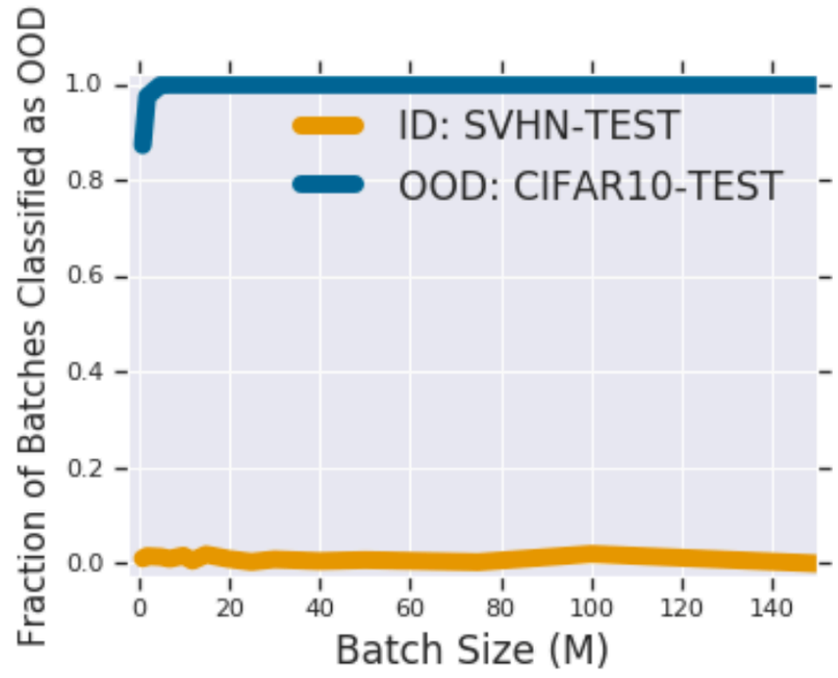
Results: Glow Trained on FashionMNIST



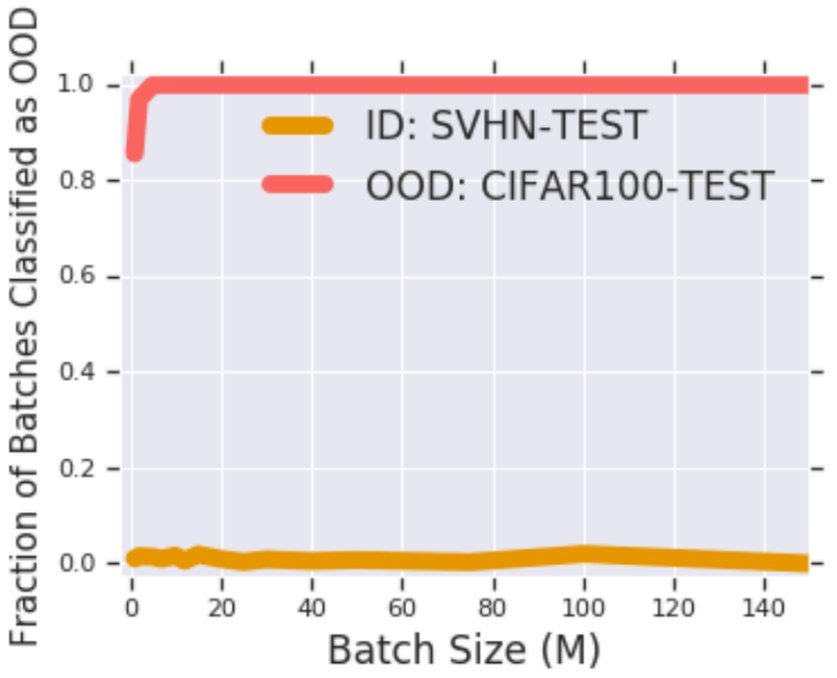
	M = 2			M = 10			M = 25		
METHOD	IN-DIST.	MNIST	NOTMNIST	IN-DIST.	MNIST	NOTMNIST	IN-DIST.	MNIST	NOTMNIST
Glow Trained on FashionMNIST									
Typicality Test	0.02±.01	0.14±.10	**0.08**±.04	0.02±.02	**1.00**±.00	**0.69**±.11	0.01±.00	**1.00**±.00	**1.00**±.00
KS-Test	**0.00**±.00	0.00±.00	0.00±.00	**0.01**±.00	**1.00**±.00	0.61±.01	**0.00**±.00	**1.00**±.00	0.98±.01
Max Mean Dis.	0.05±.02	**0.17**±.06	0.04±.03	0.02±.02	0.63±.12	0.37±.24	0.04±.04	**1.00**±.00	**1.00**±.00
Kern. Stein Dis.	0.05±.05	0.16±.14	0.01±.01	**0.01**±.01	0.21±.11	0.01±.00	0.02±.03	0.76±.21	0.00±.00

Fraction of M-sized batches classified as OOD (α = 99%)

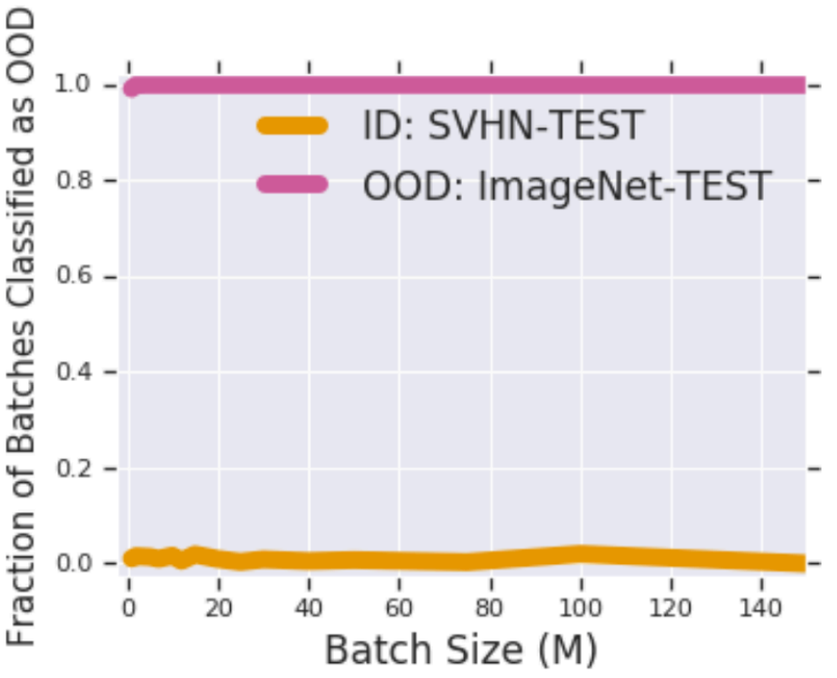
Results: Glow Trained on SVHN



(a) SVHN Train, CIFAR10 Test

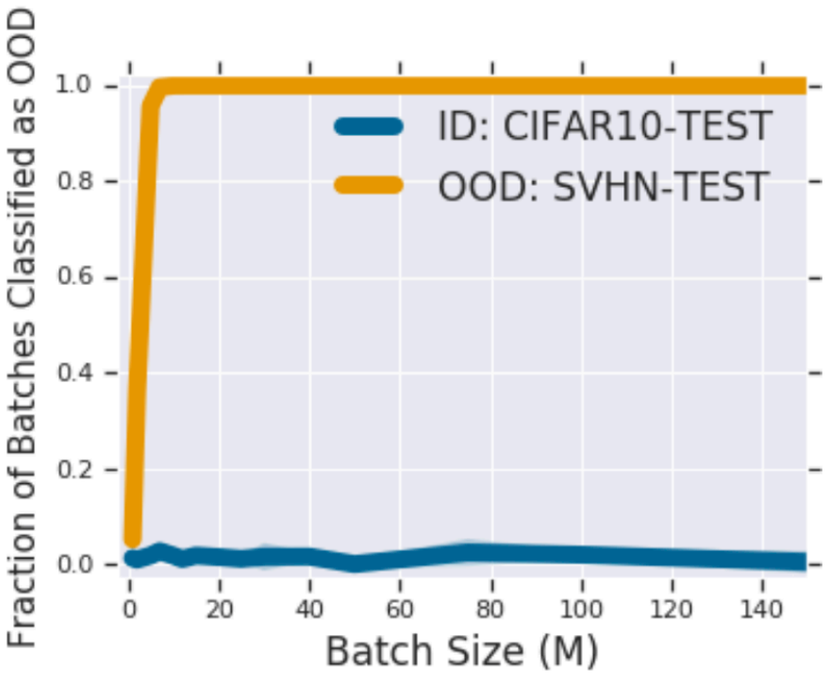


(b) SVHN Train, CIFAR100 Test

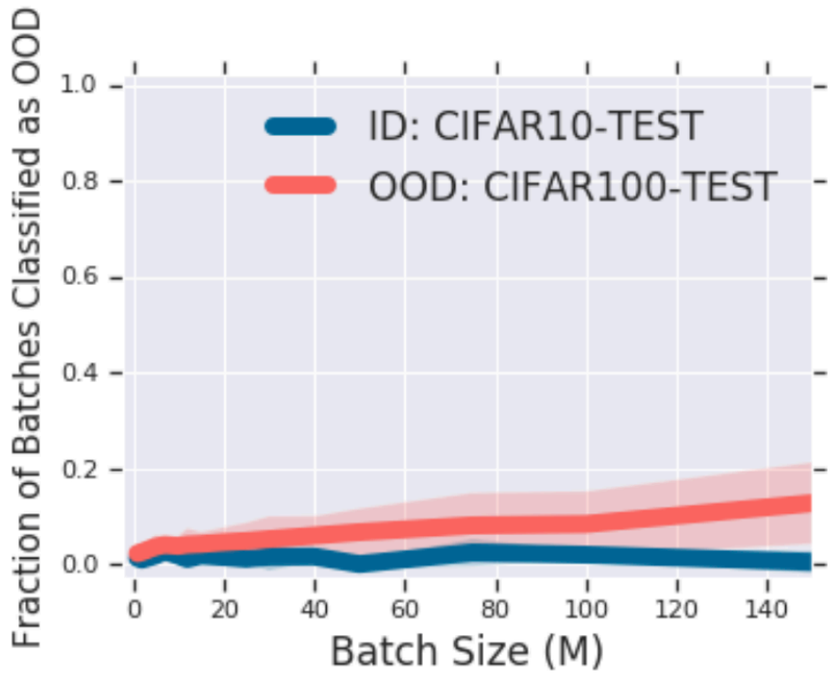


(c) SVHN Train, ImageNet Test

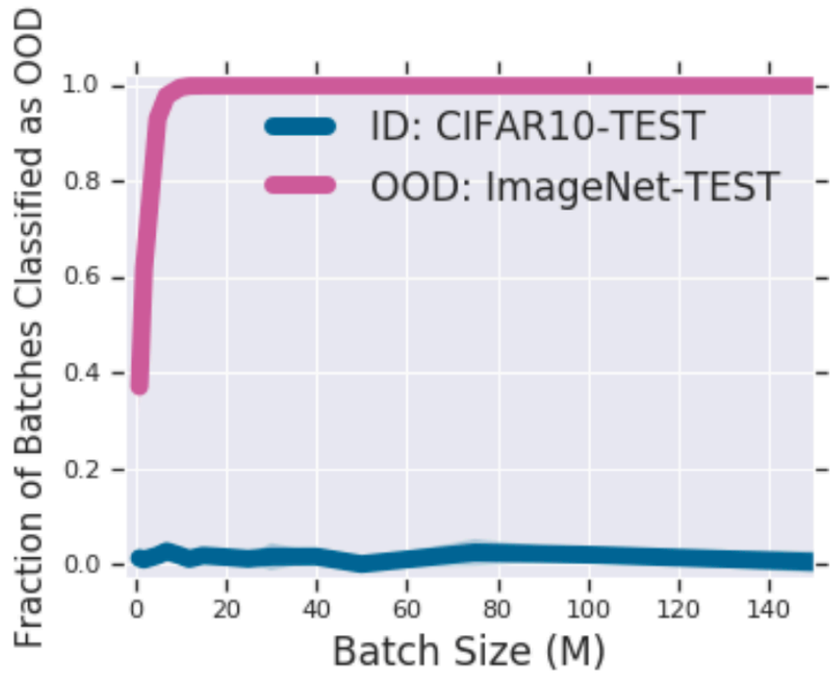
Results: Glow Trained on CIFAR-10



(d) CIFAR10 Train, SVHN Test

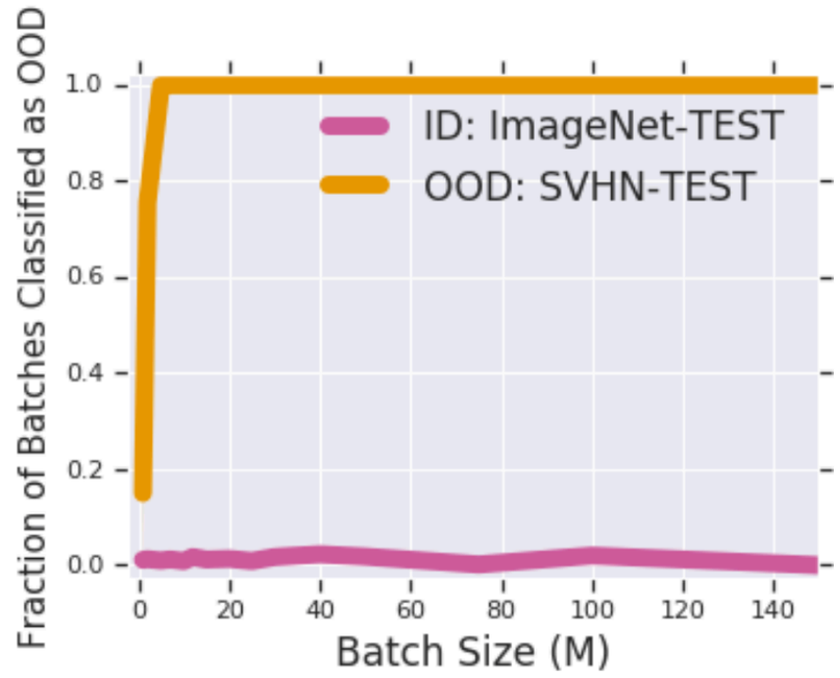


(e) CIFAR10 Train, CIFAR100 Test

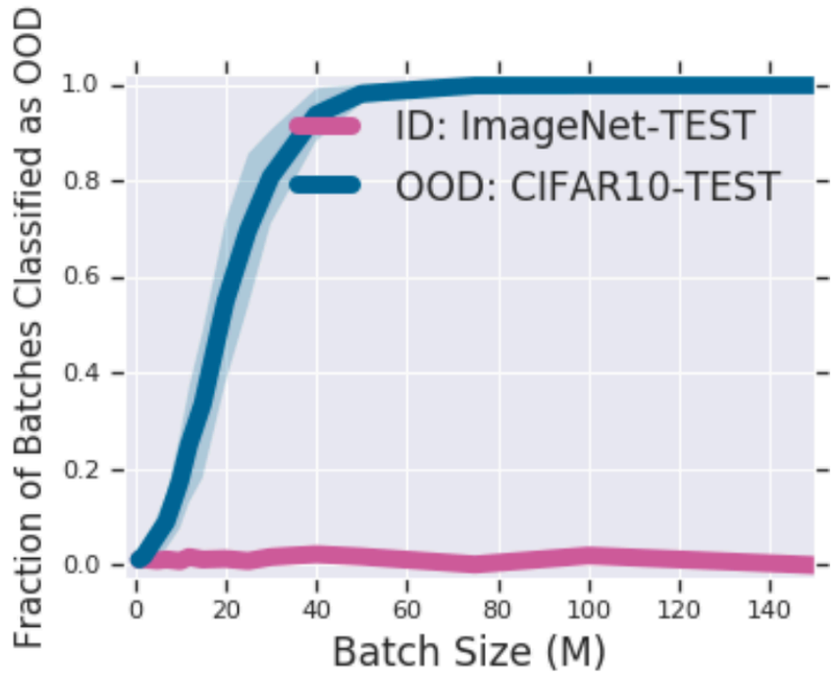


(f) CIFAR10 Train, ImageNet Test

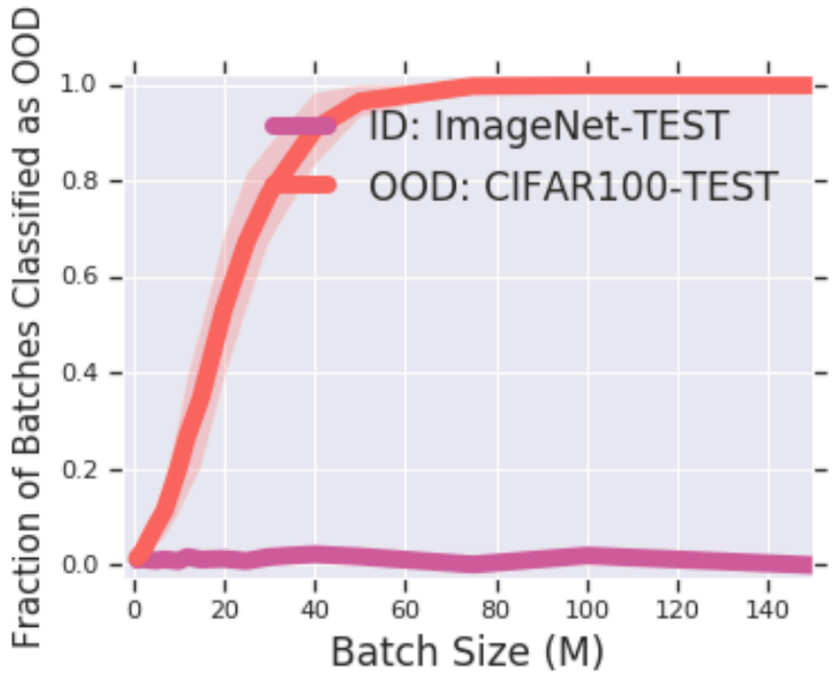
Results: Glow Trained on ImageNet



(g) ImageNet Train, SVHN Test

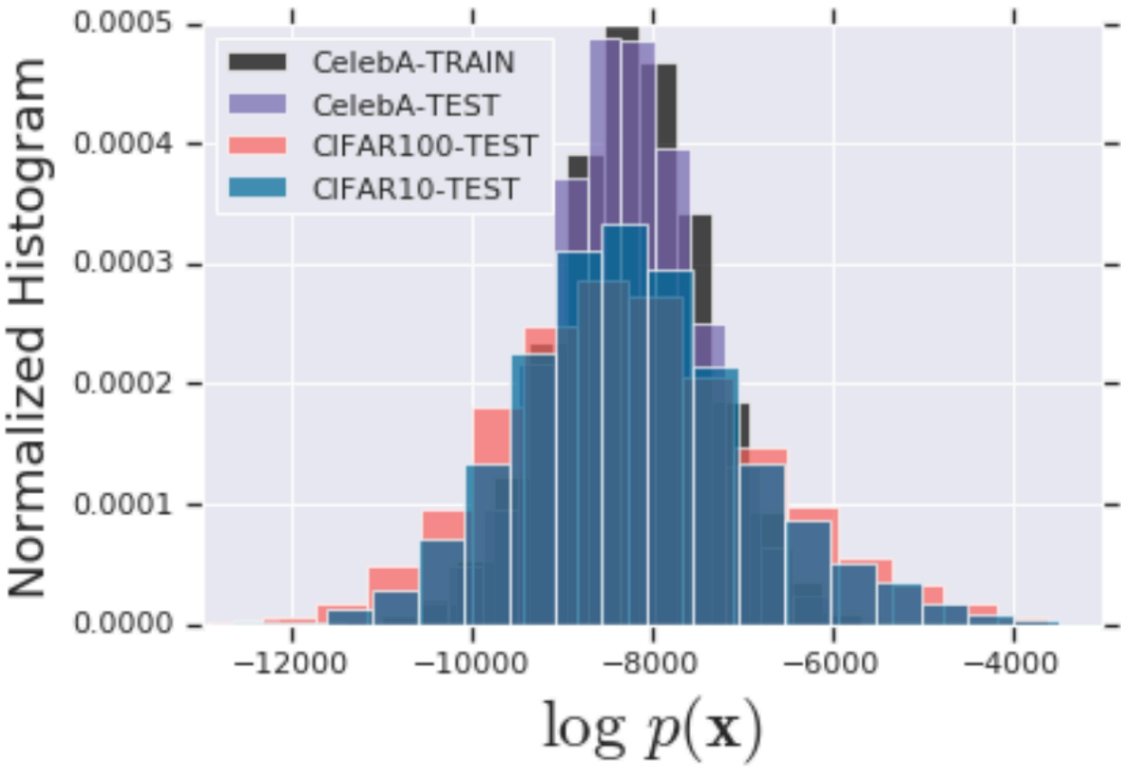


(h) ImageNet Train, CIFAR10 Test

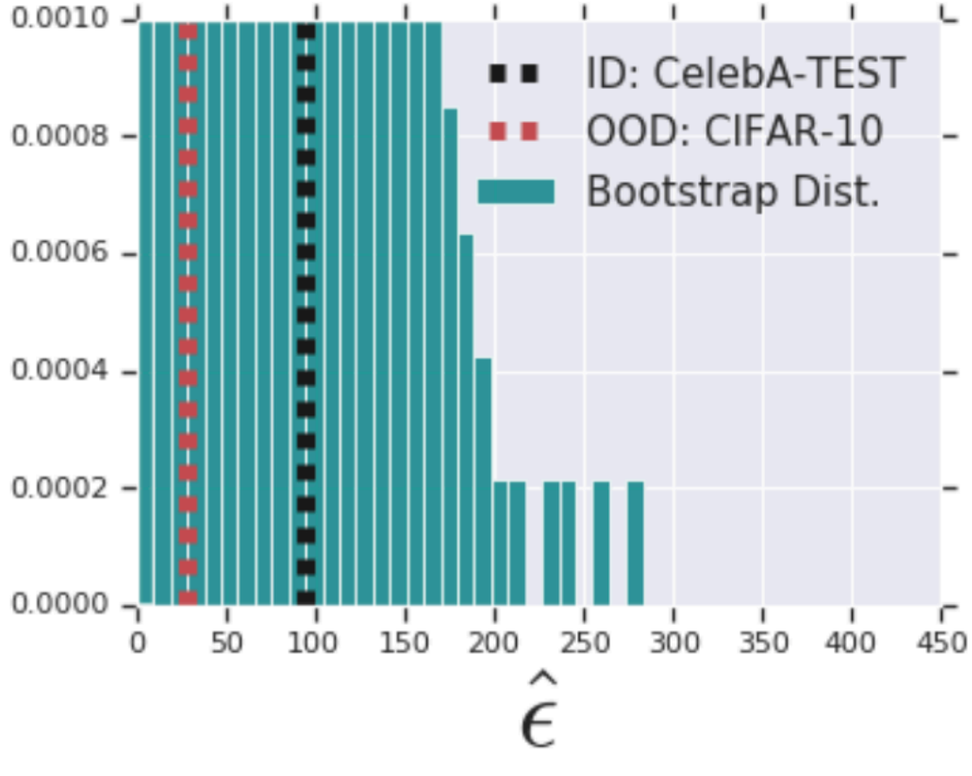


(i) ImageNet Train, CIFAR100 Test

Limitations: Only as Good as Likelihood Dist.



(b) Glow: CelebA vs CIFARs



(c) Bootstrap Dist. ($M = 200$)

PART #5

Conclusions

Conclusions

1. **Failure of Likelihood-Based Anomaly Detection**: Using a likelihood / density threshold is a bad idea for OOD / anomaly detection, especially in high dimensions.

2. **Connecting Likelihoods to Typical Sets**: Using the empirical dist. of likelihoods can still work surprisingly well, if used properly (via connection to typicality).

3. **Moving Beyond Generation:** If we really want to leverage these new generative models for principled probabilistic inference, we need to devise goodness-of-fit tests and other critiques that can appropriately scale.

Published as a conference paper at ICLR 2019

DO DEEP GENERATIVE MODELS KNOW WHAT THEY DON'T KNOW?

Eric Nalisnick[*†], **Akihiro Matsukawa, Yee Whye Teh, Dilan Gorur, Balaji Lakshminarayanan**[*]
DeepMind

https://openreview.net/forum?id=H1xwNhCcYm

Detecting Out-of-Distribution Inputs to Deep Generative Models Using a Test for Typicality

Eric Nalisnick*
DeepMind
enalisnick@google.com

Akihiro Matsukawa
DeepMind
amatsukawa@google.com

Yee Whye Teh
DeepMind
ywteh@google.com

Balaji Lakshminarayanan*
DeepMind
balajiln@google.com

Coming soon to ArXiv...

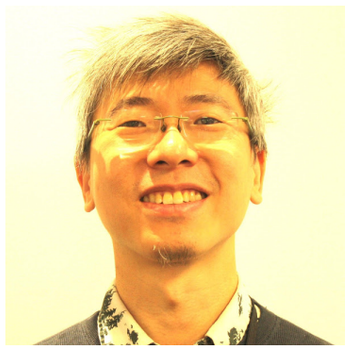
Thank you. Questions?

In collaboration with…

Aki Matsukawa

Balaji Lakshminarayanan

Dilan Gorur



Yee Whye Teh