Adversarial Autoencoders

Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, Ian Goodfellow, Brendan Frey

Presented by: Paul Vicol

Outline

- Adversarial Autoencoders
  - AAE with *continuous* prior distributions
  - AAE with *discrete* prior distributions
  - AAE vs VAE
- Wasserstein Autoencoders
  - Generalization of Adversarial Autoencoders
  - Theoretical Justification for AAEs

Regularizing Autoencoders

- Classical *unregularized* autoencoders minimize a reconstruction loss $||x - \hat{x}||^2$
- This yields an *unstructured latent space*
  - Examples from the data distribution are mapped to codes scattered in the space
  - No constraint that similar inputs are mapped to nearby points in the latent space
  - We cannot sample codes to generate novel examples

- VAEs are one approach to regularizing the latent distribution

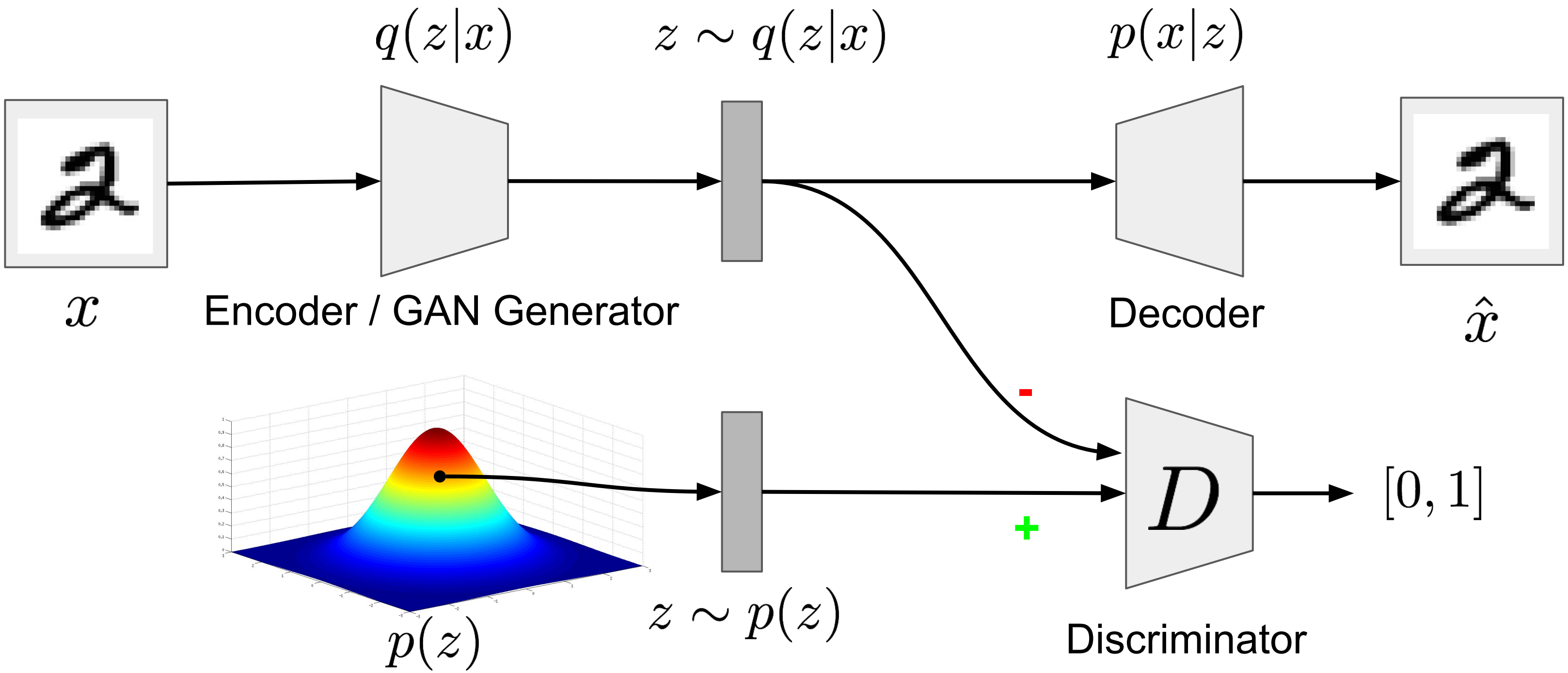
Adversarial Autoencoders - Motivation

- **<u>Goal:</u>** An approach to *impose structure* on the latent space of an autoencoder

- **<u>Idea:</u>** Train an autoencoder with an *adversarial loss* to match the distribution of the latent space to an arbitrary prior
  - Can use *any prior that we can sample from* either continuous (*Gaussian*) or discrete (*Categorical*)

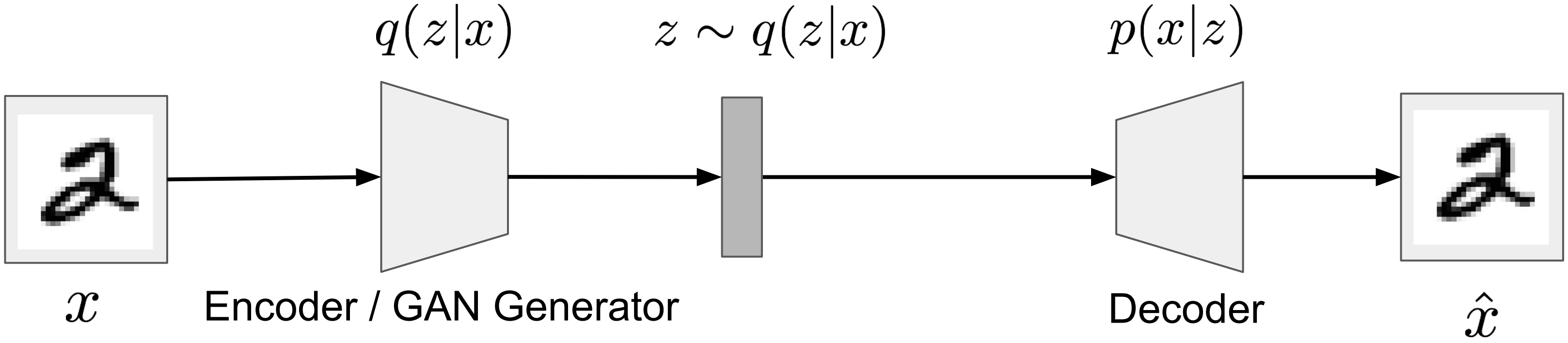
AAE Architecture

- Adversarial autoencoders are generative autoencoders that use *adversarial training* to impose an arbitrary prior on the latent code



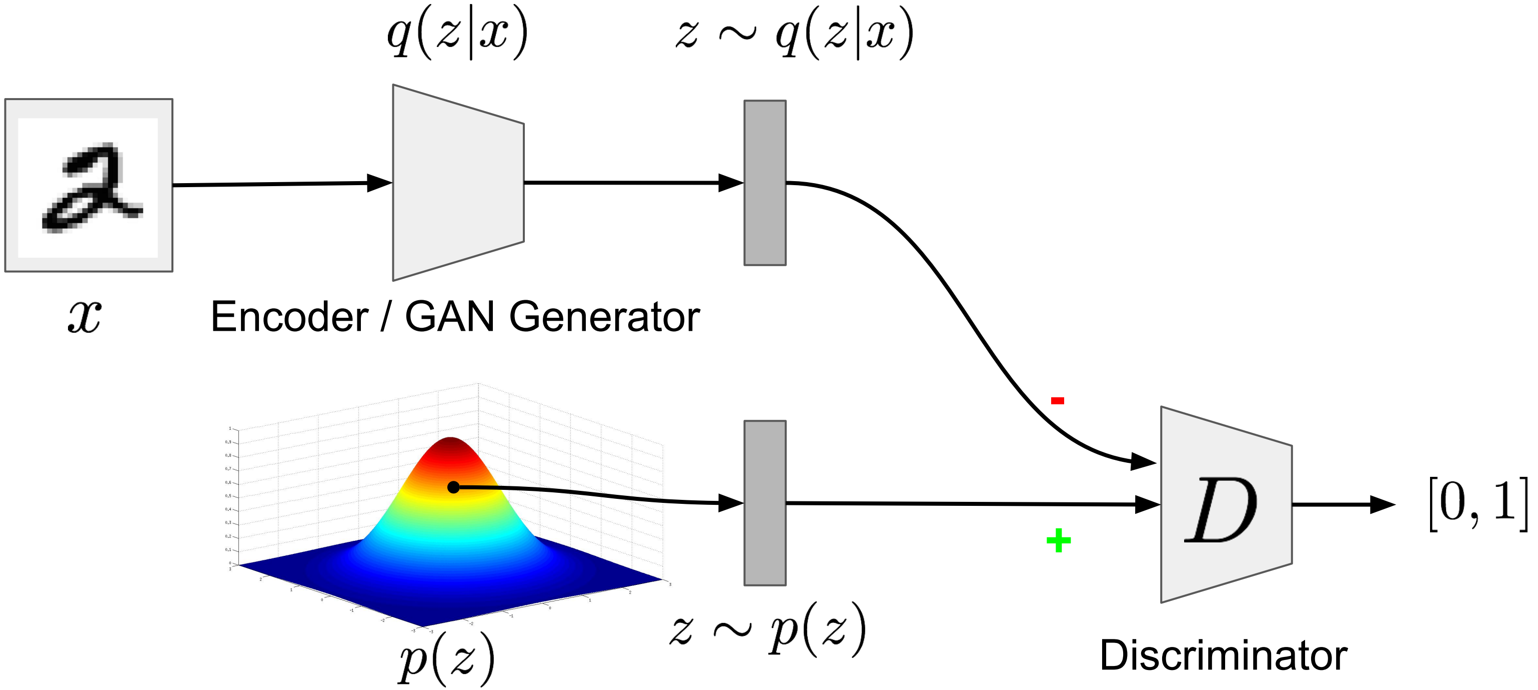
Training an AAE - Phase 1

1. The **<u>reconstruction phase</u>**: Update the encoder and decoder to minimize reconstruction error



Training an AAE - Phase 2

2. **<u>Regularization phase</u>**: Update discriminator to distinguish true prior samples from generated samples; update generator to fool the discriminator



AAE vs VAE

- VAEs use a KL divergence term to impose a prior on the latent space
- AAEs use adversarial training to match the latent distribution with the prior

$$\mathcal{L} = \mathbb{E}_x \underbrace{\left[\mathbb{E}_{q(z|x)}[-\log p(x|z)]\right]}_{\text{Reconstruction Error}} + \mathbb{E}_x \underbrace{\left[\mathrm{KL}(q(z|x)||p(z))\right]}_{\text{KL Regularizer}}$$

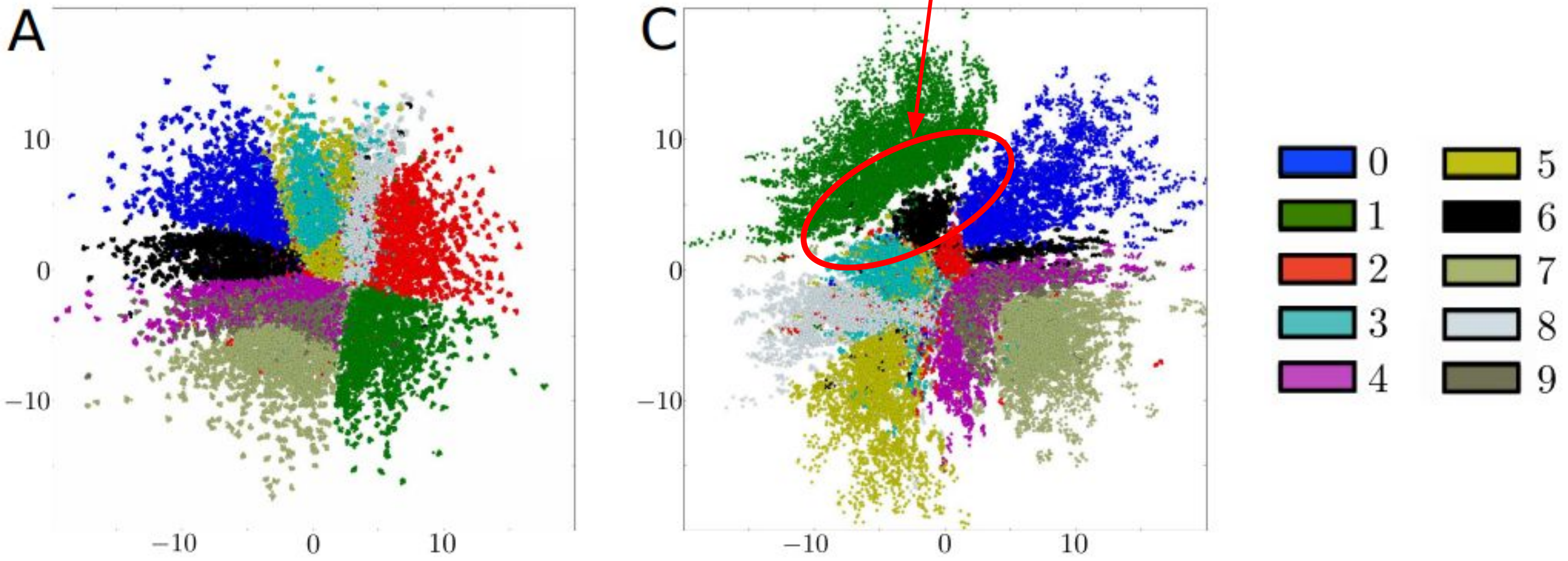
Replaced by adversarial loss in AAE

- Why would we use an AAE instead of a VAE?
  - To backprop through the KL divergence we must have access to the functional form of the prior distribution p(z)
  - In an AAE, we just need to be able to *sample* from the prior to induce the latent distribution to match the prior

AAE vs VAE: Latent Space

- Imposing a *Spherical 2D Gaussian prior* on the latent space

Gaps in the latent space; not well-packed

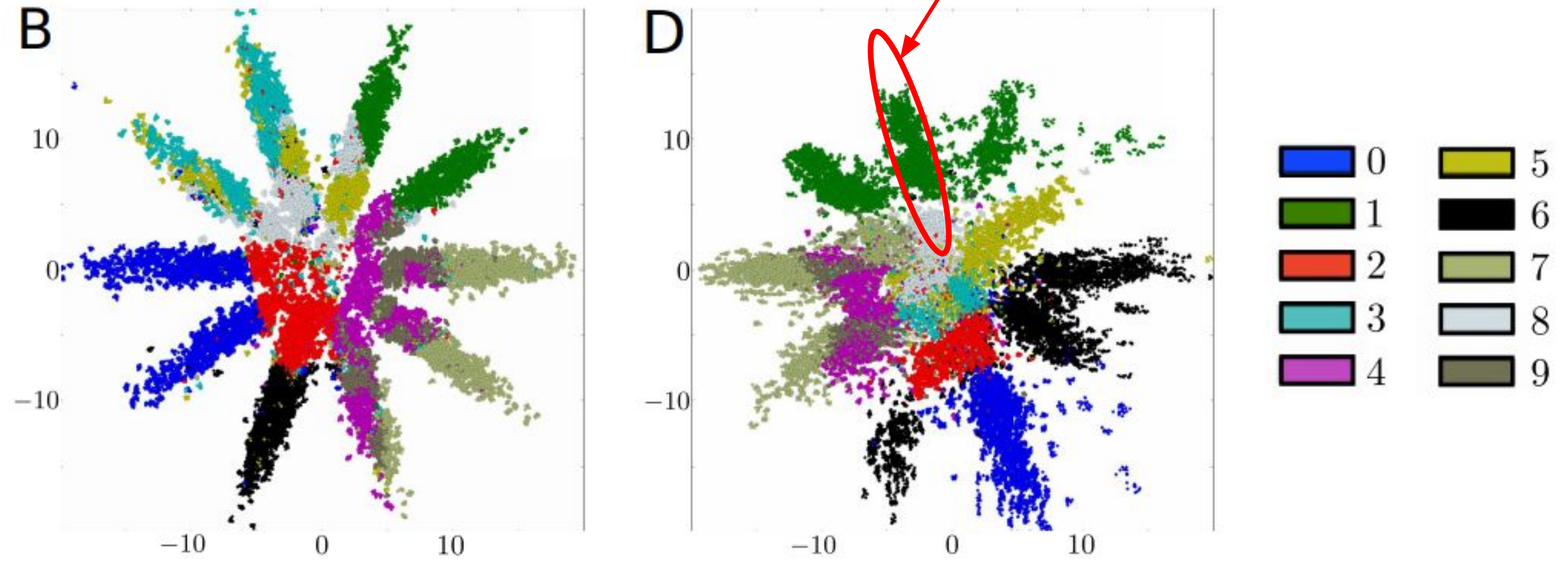


AAE **VAE**

AAE vs VAE: Latent Space

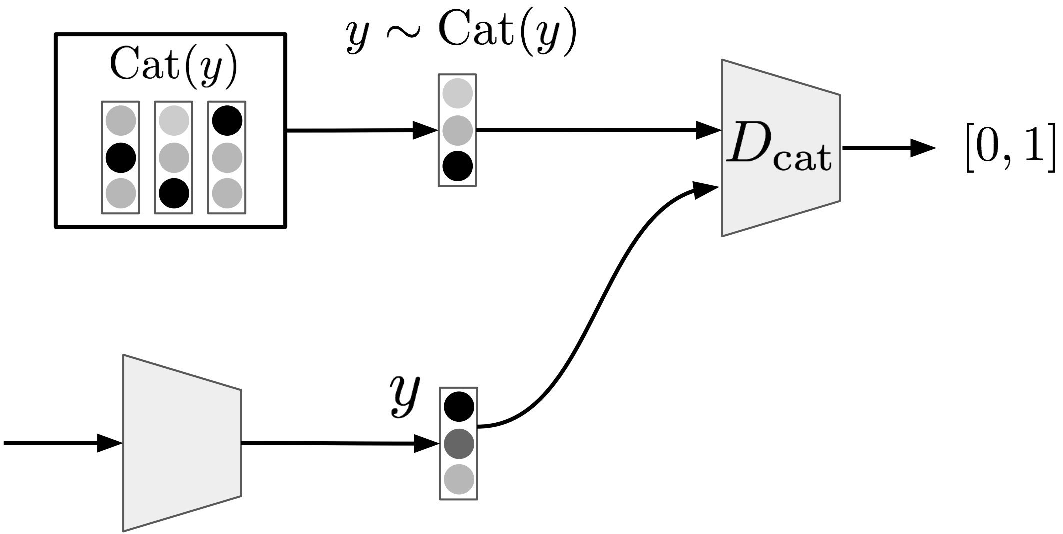
- Imposing a *mixture of 10 2D Gaussians* prior on the latent space

VAE emphasizes the modes of the distribution; has systematic differences from the prior



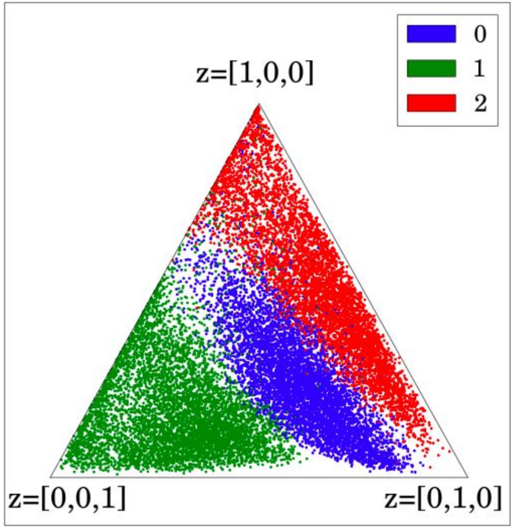
AAE **VAE**

GAN for Discrete Latent Structure

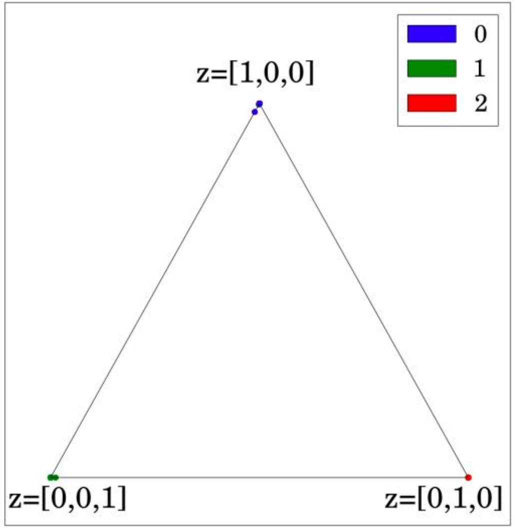


- **<u>Core idea:</u>** Use a discriminator to check that a latent variable is discrete

GAN for Discrete Latent Structure



Without GAN Regularization



With GAN Regularization

- D_{cat} induces the softmax output y to be *highly peaked* at one value
- Similar to continuous relaxation with temperature annealing, but does not require setting a temperature or annealing schedule

Semi-Supervised Adversarial Autoencoders

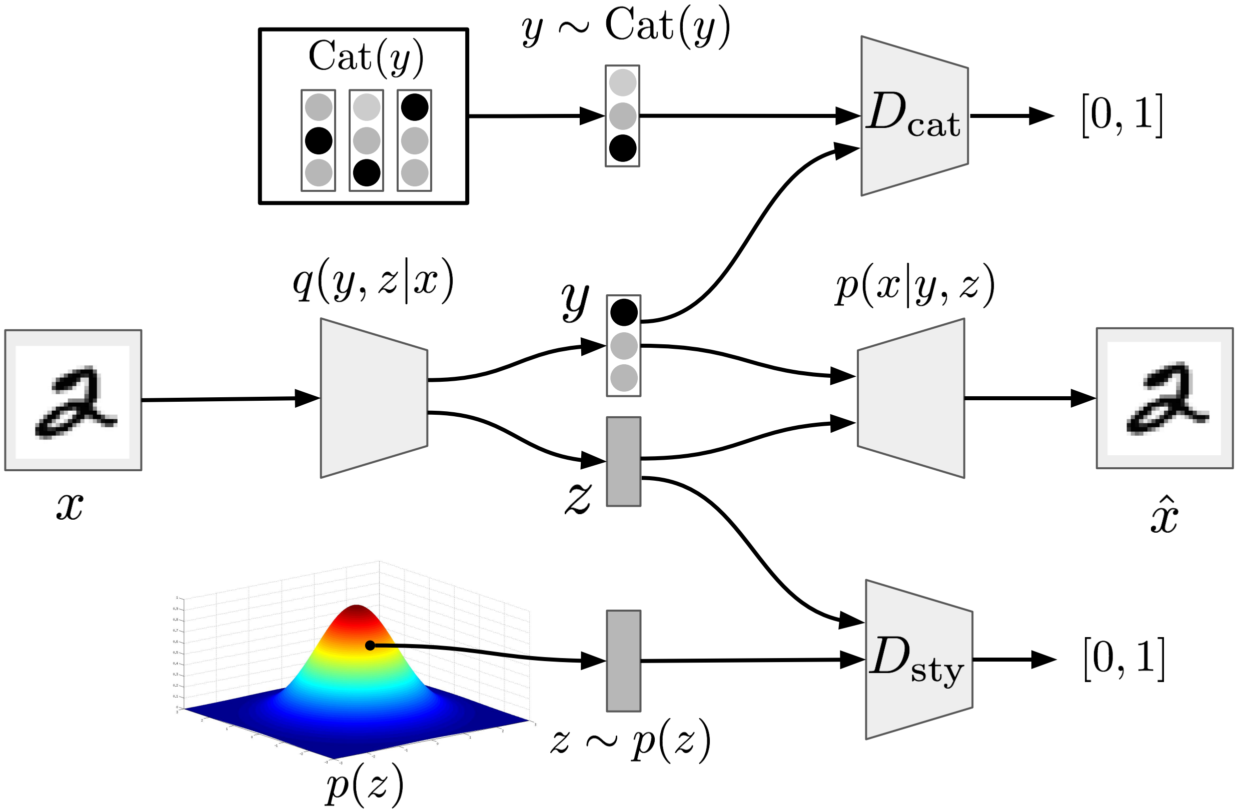
- Model for semi-supervised learning that exploits the generative description of the unlabeled data to improve classification performance
- Assume the data is generated as follows:

$$p(y) = \mathrm{Cat}(y)$$

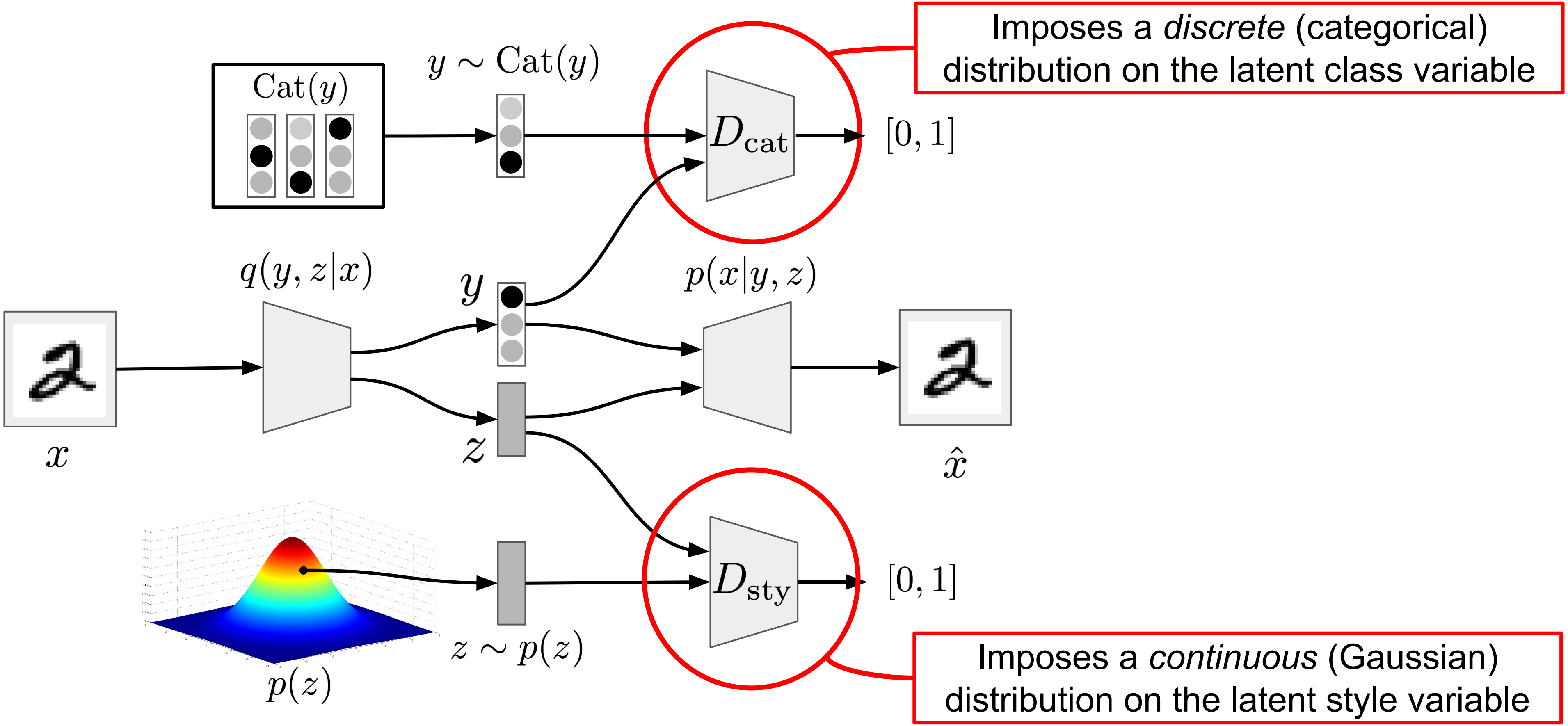
$$p(z) = \mathcal{N}(z|0, I)$$

- Now the encoder predicts both the discrete class y (content) and the continuous code z (style)
- The decoder conditions on both the class label and style vector

Semi-Supervised Adversarial Autoencoders



Semi-Supervised Adversarial Autoencoders



Semi-Supervised Classification Results

- AAEs outperform VAEs

	MNIST (100)	MNIST (1000)	MNIST (All)	SVHN (1000)
NN Baseline	25.80	8.73	1.25	47.50
VAE (M1) + TSVM	11.82 ($\pm$0.25)	4.24 ($\pm$0.07)	-	55.33 ($\pm$0.11)
VAE (M2)	11.97 ($\pm$1.71)	3.60 ($\pm$0.56)	-	-
VAE (M1 + M2)	3.33 ($\pm$0.14)	2.40 ($\pm$0.02)	0.96	36.02 ($\pm$0.10)
VAT	2.33	1.36	0.64 ($\pm$0.04)	24.63
CatGAN	1.91 ($\pm$0.1)	1.73 ($\pm$0.18)	0.91	-
Ladder Networks	1.06 ($\pm$0.37)	0.84 ($\pm$0.08)	0.57 ($\pm$0.02)	-
ADGM	0.96 ($\pm$0.02)	-	-	16.61 ($\pm$0.24)
Adversarial Autoencoders	1.90 ($\pm$0.10)	1.60 ($\pm$0.08)	0.85 ($\pm$0.02)	17.70 ($\pm$0.30)

Table 2: Semi-supervised classification performance (error-rate) on MNIST and SVHN.

Unsupervised Clustering with AAEs

- An AAE can disentangle *discrete class variables* from continuous latent style variables without supervision
- The inference network $q(y|x)$ predicts one-hot vector with K = *num clusters*

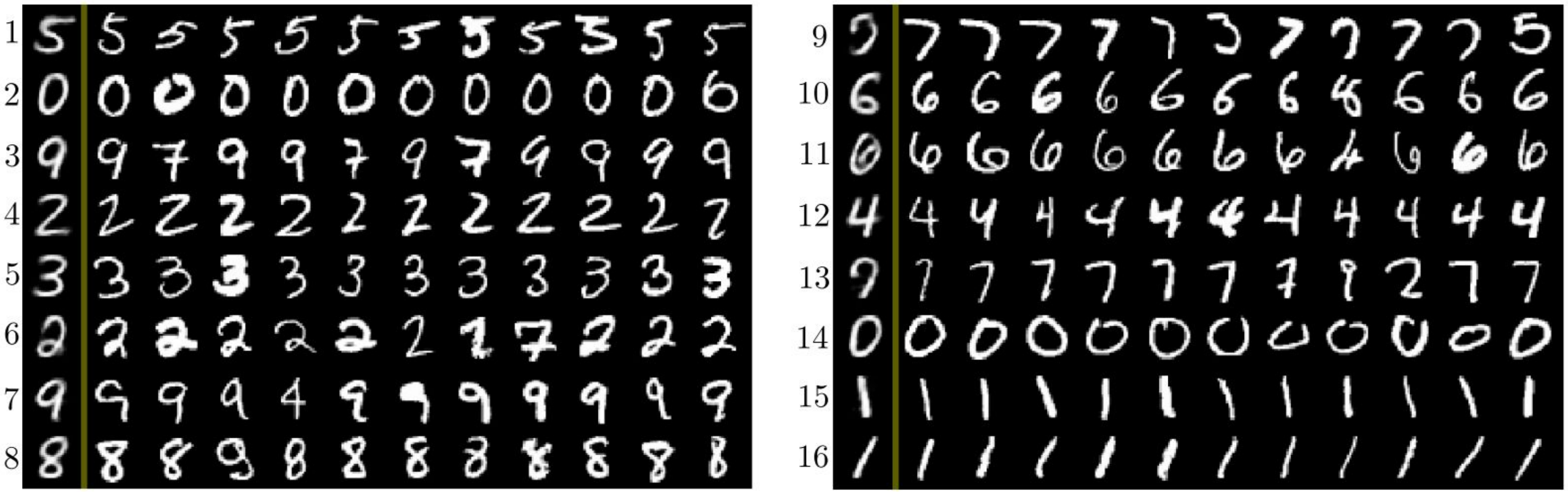


Figure 9: Unsupervised clustering of MNIST using the AAE with 16 clusters. Each row corresponds to one cluster with the first image being the cluster head. (see text)

Adversarial Autoencoder Summary

Pros

- Flexible approach to impose arbitrary distributions over the latent space
- Works with any distribution you can sample from, continuous and discrete
- Does not require temperature/annealing hyperparameters

Cons

- May be challenging to train due to the GAN objective
- Not scalable to many latent variables → need a discriminator for each

Wasserstein Auto-Encoders (Oral, ICLR 2018)

- Generative models (VAEs & GANs) try to minimize discrepancy measures between the data distribution P_X and the model distribution P_G
- WAE minimizes a penalized form of the Wasserstein distance between the model distribution and the target distribution:

$$D_{\mathrm{WAE}}(P_X, P_G) := \inf_{Q(Z|X)\in\mathcal{Q}} \underbrace{\mathbb{E}_{P_X}\mathbb{E}_{Q(Z|X)}\big[c\big(X, G(Z)\big)\big]}_{\textbf{Reconstruction}\text{ cost}} + \lambda \cdot \underbrace{\mathcal{D}_Z(Q_Z, P_Z)}_{\textbf{Regularizer}\text{ encourages the encoded distribution to match the prior}}$$

WAE - Justification for AAEs

- **<u>Theoretical justification for AAEs:</u>**
- When $c(x, y) = ||x - y||_2^2$ WAE = AAE
- AAEs minimize the 2-Wasserstein distance between P_X and P_G

- *WAE generalizes AAE* in two ways:
  1. Can use any cost function $c(x, y)$ in the input space $\mathcal{X}$
  2. Can use any discrepancy measure D_Z in the latent space $\mathcal{Z}$
     - Not just an adversarial one

Thank you!