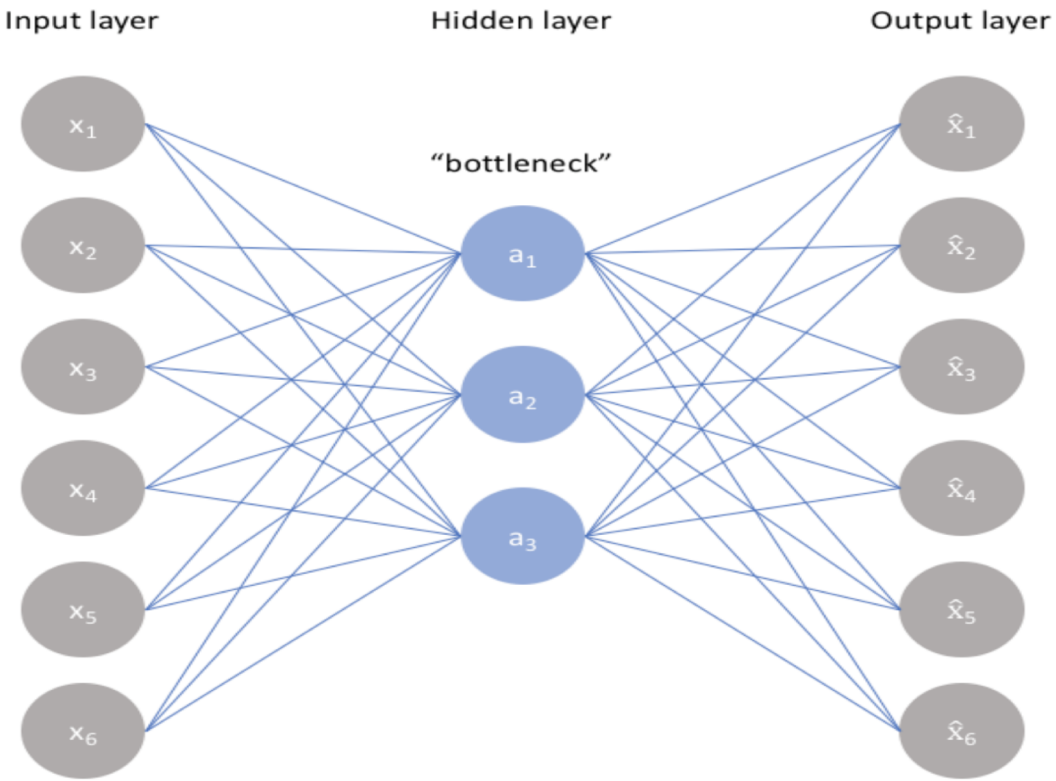
Adversarial Autoencoders

Mohit Bajaj

Autoencoder

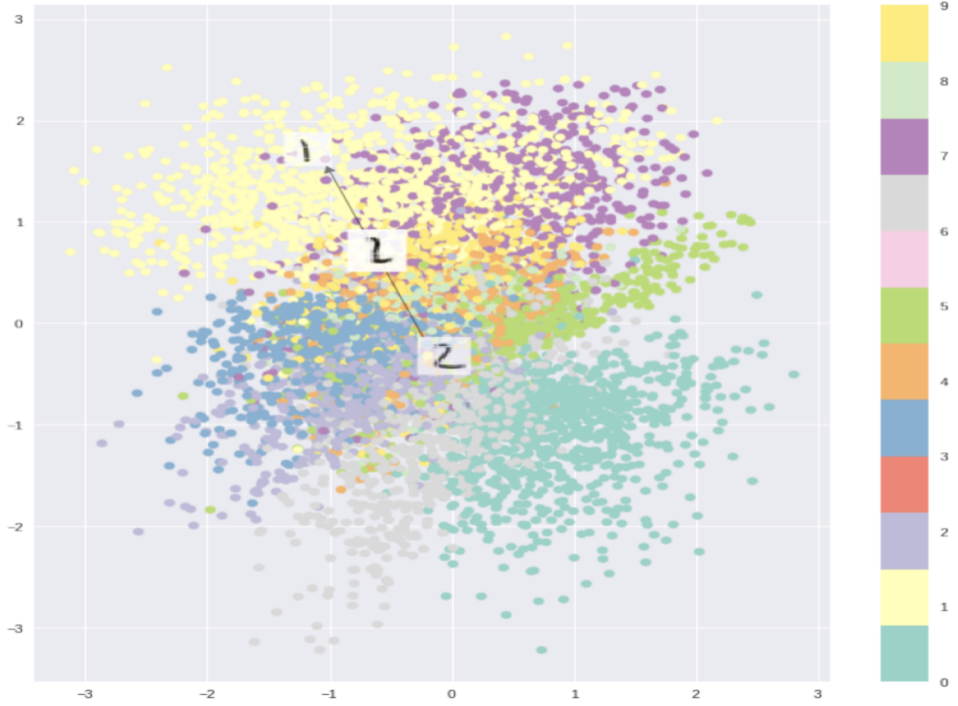


Autoencoder: Latent space

Optimizing purely for reconstruction loss

https://towardsdatascience.com/intuitively-understanding-variational-autoencoders-1bfe67eb5daf

Variational Autoencoders



Optimizing using both reconstruction loss and KL divergence loss

Regularization + Stochasticity

https://towardsdatascience.com/intuitively-understanding-variational-autoencoders-1bfe67eb5daf

Variational Autoencoders - ELBO

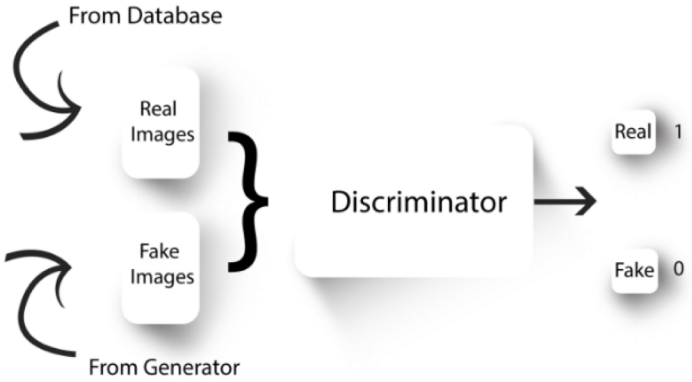
- $q_\phi(z|x)$ is a good approximation to $p(z|x)$:

$$\log p(x) - KL(q_\phi(z|x) \,||\, p_\theta(z|x)) = \mathbb{E}_{z\sim q_\phi(z|x)}[\log p(x|z)] - KL(q_\phi(z|x) \,||\, p(z))$$

$$\log p(x) \geq \underbrace{\mathbb{E}_{z\sim q_\phi(z|x)}[\log p_\theta(x|z)] - KL(q_\phi(z|x) \,||\, p(z))}_{\text{ELBO}}$$

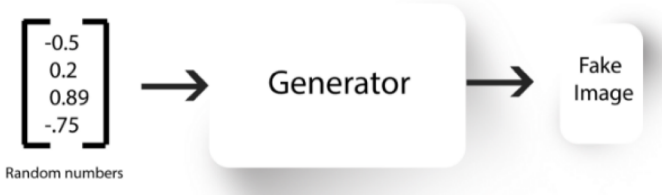
Slide borrowed from Siddhesh

GAN



Discriminator

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_r(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z)))]$$



Variational Autoencoder vs GAN

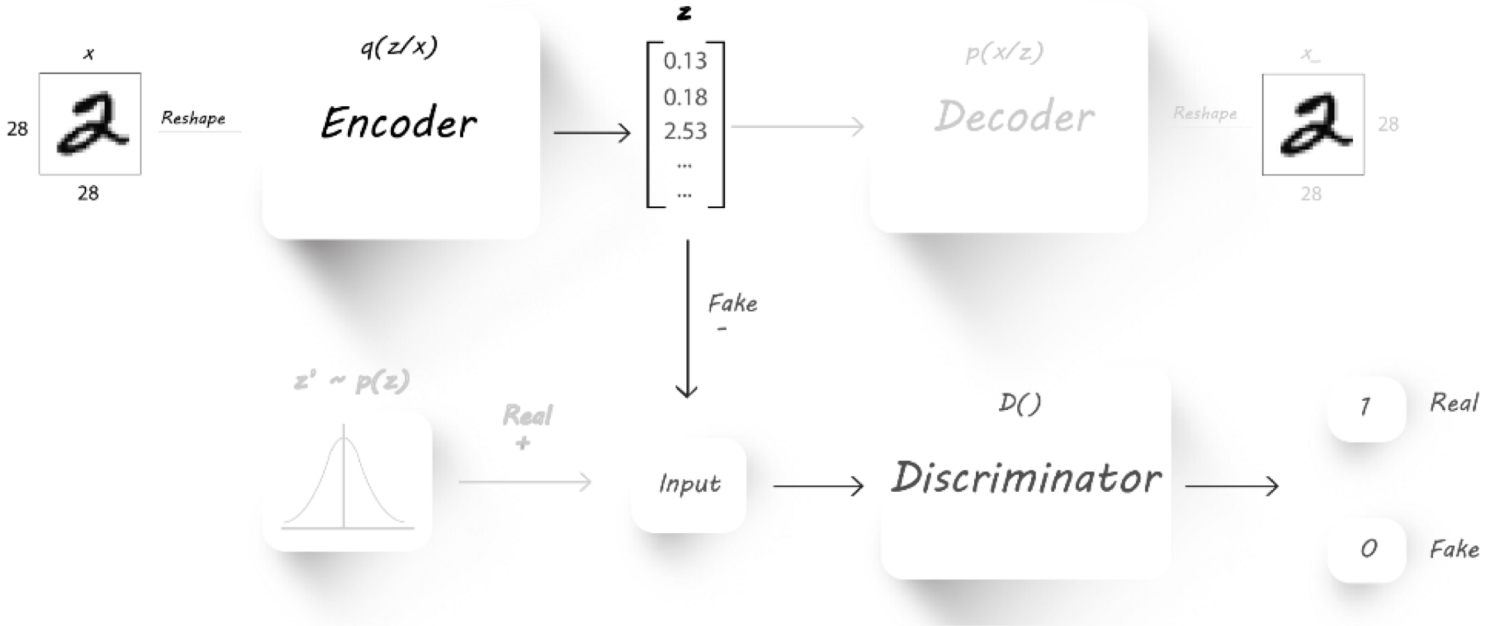
- VAEs have good interpretation because we obtain the approximate posterior unlike GANs (implicit)
  - But need to know prior in closed form (Reparameterization trick)
  - Priors can’t be complex
- GANs are better in capturing the distribution because of no explicit constraints
  - But no latent representations for the interpretation

Variational Autoencoder vs GAN

- VAEs have good interpretation because we obtain the approximate posterior unlike GANs (implicit)
  - But need to know prior in closed form (Reparameterization trick)
  - Priors can’t be complex
- GANs are better in capturing the distribution because of no explicit constraints
  - But no latent representations for the interpretation

Can we get best of both the worlds?

AAE


x
28
28
Reshape
q(z/x)
Encoder
z
0.13
0.18
2.53
...
...
p(x/z)
Decoder
Reshape
x_
28
28
Fake
-
z' ~ p(z)
Real
+
Input
D()
Discriminator
1
Real
0
Fake

AAE

- Autoencoder framework
  - Provides interpretable latent representation
- Doesn’t use KL divergence (regularization) used in VAEs
  - Instead uses adversarial training for regularization
- Need not know prior in the closed form
  - Only should be able to sample from it
- No need for the reparameterization trick
  - No need to sample from $q(z|x)$: Deterministic

AAE vs VAE

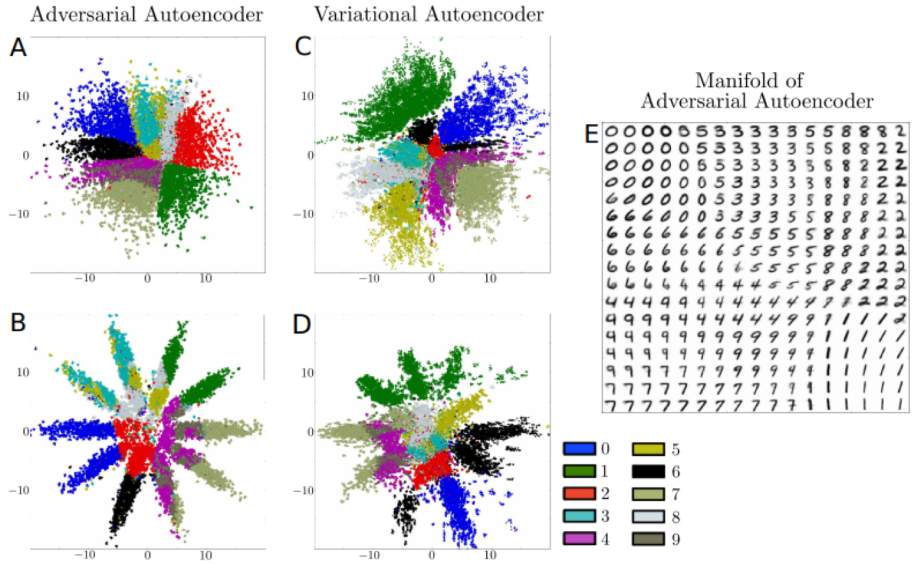


Figure 2: Comparison of adversarial and variational autoencoder on MNIST. The hidden code **z** of the *hold-out* images for an adversarial autoencoder fit to (a) a 2-D Gaussian and (b) a mixture of 10 2-D Gaussians. Each color represents the associated label. Same for variational autoencoder with (c) a 2-D gaussian and (d) a mixture of 10 2-D Gaussians. (e) Images generated by uniformly sampling the Gaussian percentiles along each hidden code dimension **z** in the 2-D Gaussian adversarial autoencoder.

Labels for regularization

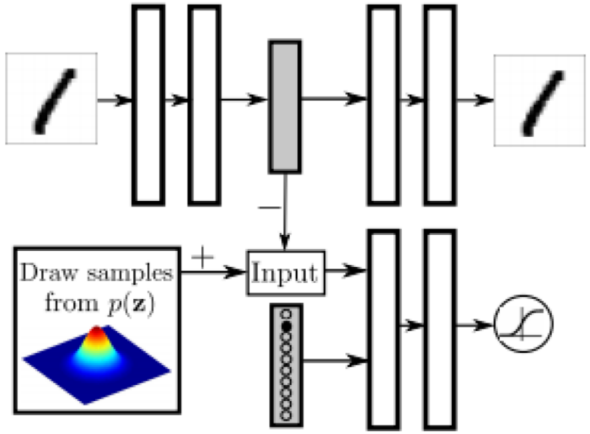
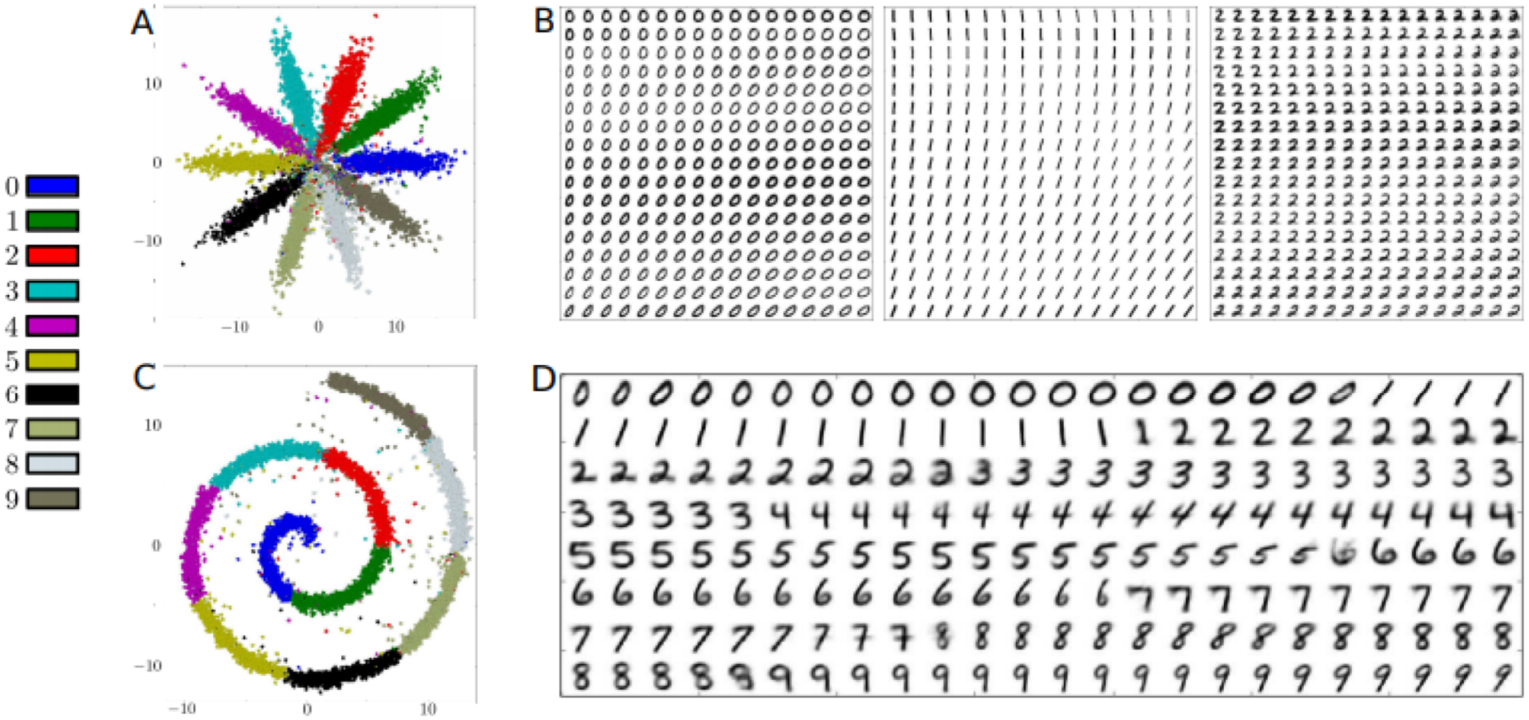


Figure 3: Regularizing the hidden code by providing a one-hot vector to the discriminative network. The one-hot vector has an extra label for training points with unknown classes.

Latent space



Supervised AAE

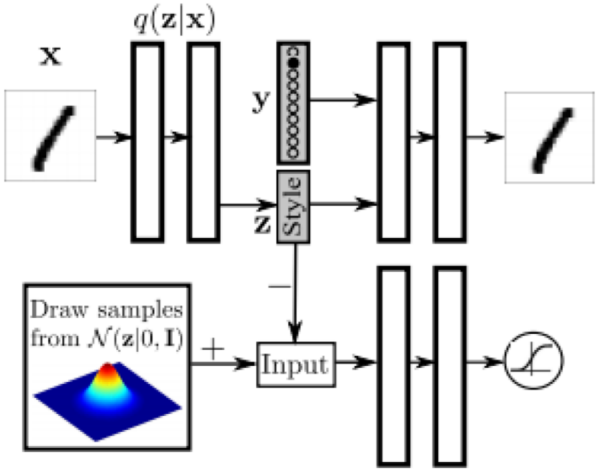
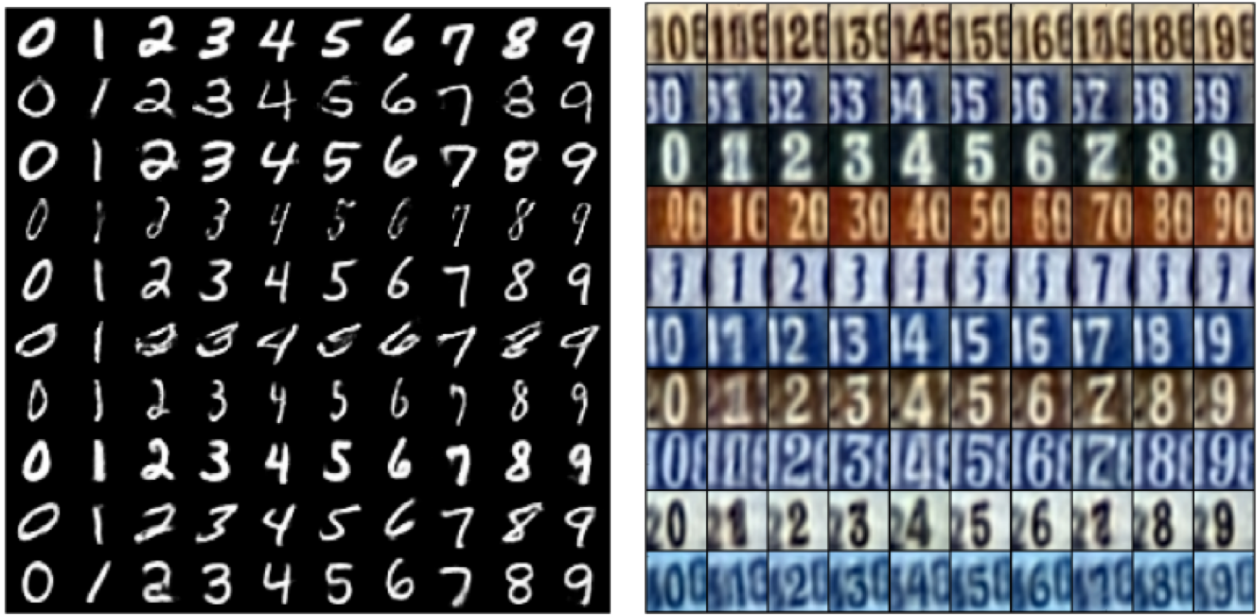


Figure 6: Disentangling the label information from the hidden code by providing the one-hot vector to the generative model. The hidden code in this case learns to represent the style of the image.

Disentanglement



(a) MNIST

(b) SVHN

Figure 7: Disentangling content and style (15-D Gaussian) on MNIST and SVHN datasets.

Semi Supervised Setting

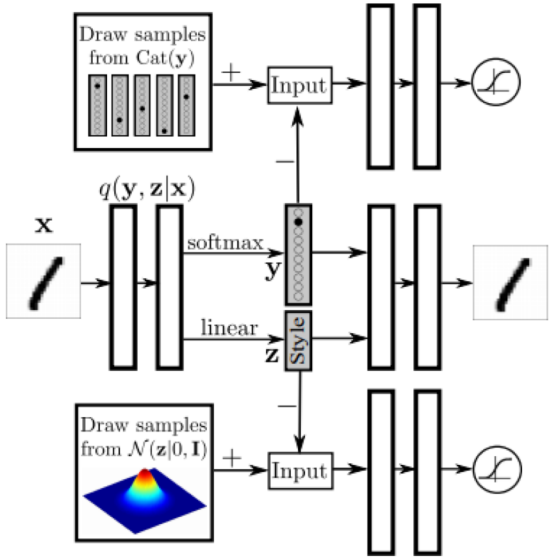


Figure 8: Semi-Supervised AAE: the top adversarial network imposes a Categorical distribution on the label representation and the bottom adversarial network imposes a Gaussian distribution on the style representation. $q(\mathbf{y}|\mathbf{x})$ is trained on the labeled data in the semi-supervised settings.

- 2 adversarial losses
- Reconstruction loss
- Cross Entropy Loss on ***y*** on known labels

Results

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

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

Unsupervised Clustering

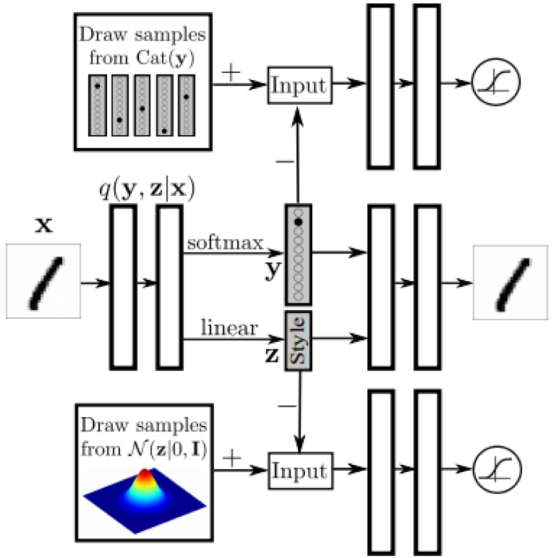


Figure 8: Semi-Supervised AAE: the top adversarial network imposes a Categorical distribution on the label representation and the bottom adversarial network imposes a Gaussian distribution on the style representation. $q(\mathbf{y}|\mathbf{x})$ is trained on the labeled data in the semi-supervised settings.

- 2 adversarial losses
- Reconstruction loss
- ~~Cross Entropy Loss on **y** on known labels~~
- Size of vector y is equal to number of clusters

Unsupervised Clustering (16 clusters)

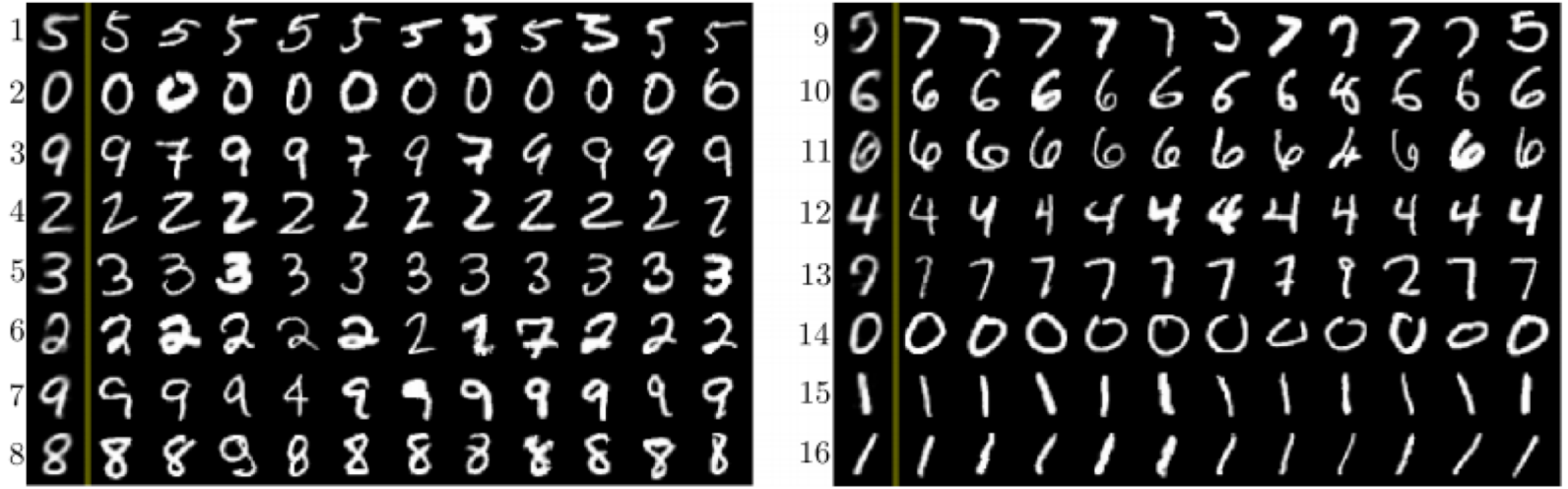


Figure 9: Unsupervised clustering of MNIST using the AAE with 16 clusters. Each row corresponds to one cluster with the first image being the cluster head. (see text)

Dimensionality reduction

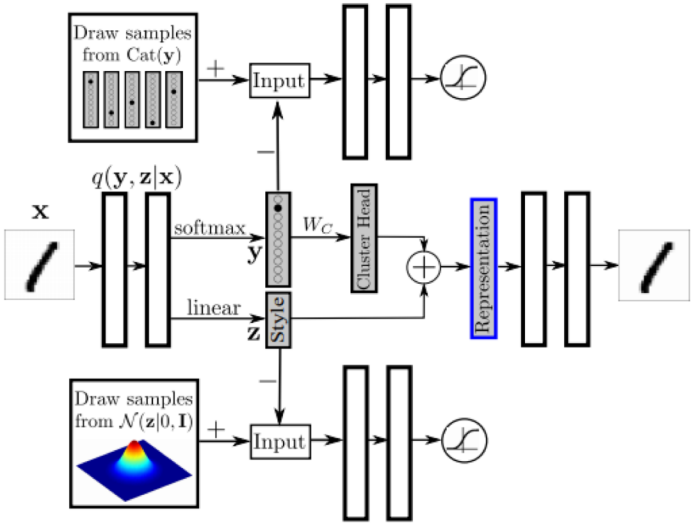
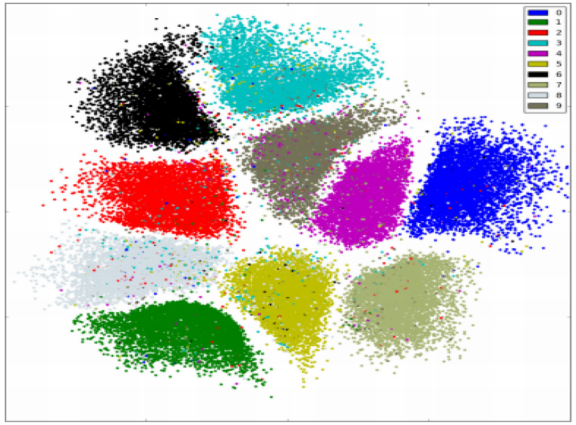
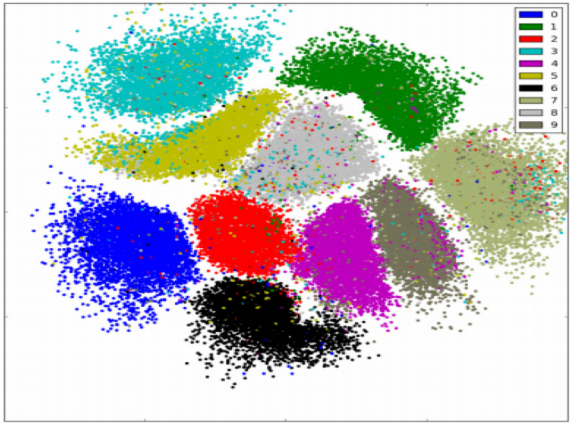


Figure 10: Dimensionality reduction with adversarial autoencoders: There are two separate adversarial networks that impose Categorical and Gaussian distribution on the latent representation. The final n dimensional representation is constructed by first mapping the one-hot label representation to an n dimensional cluster head representation and then adding the result to an n dimensional style representation. The cluster heads are learned by SGD with an additional cost function that penalizes the Euclidean distance between of every two of them.

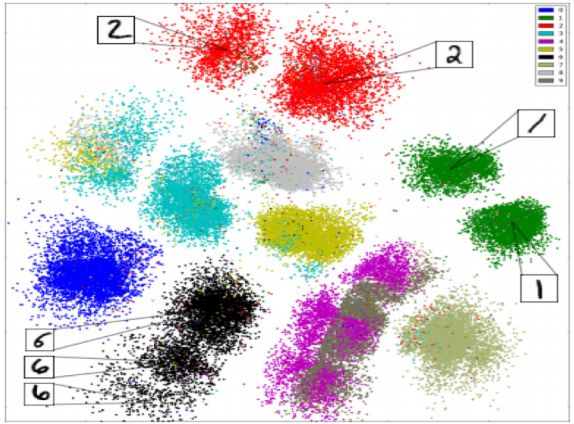
- Cluster head to project m dimensional one hot vector to n dim representation
- Penalize if euclidean distance is less b/w two two heads of cluster head matrix W (m*n dim)



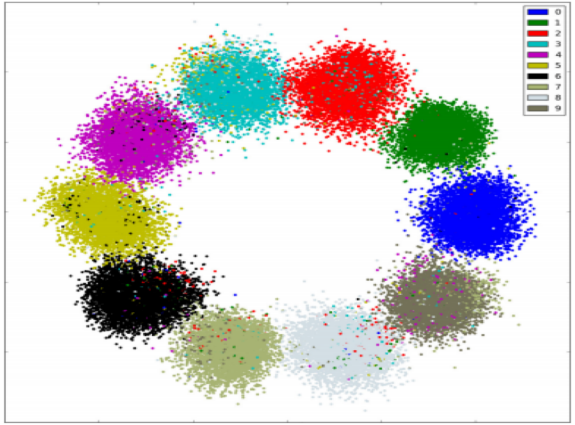
(a) 2D representation with 1000 labels (4.20% error)



(b) 2D representation with 100 labels (6.08% error)



(c) 2D representation learnt in an unsupervised fashion with 20 clusters (13.95% error)



(d) 10D representation with 100 labels projected onto the 2D space (3.90% error)

Conclusion

- Good performance + good interpretation (allows to model posterior distribution)
- Flexible framework for all scenarios with different levels of supervision