Unsupervised
Visual Learning

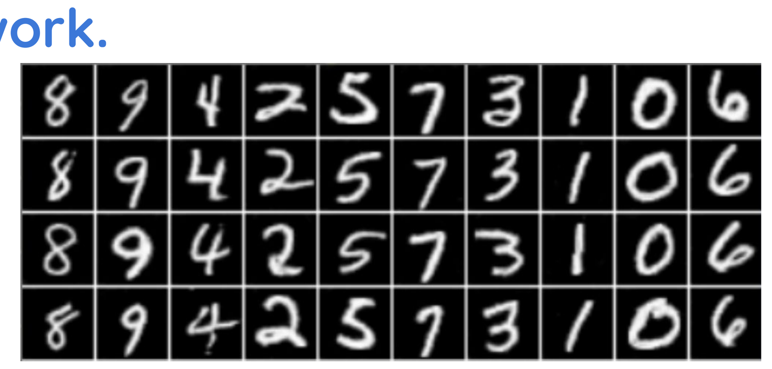
Yunze Man · Zhengyi Luo · Yiming Zuo
Advisor - Himanshi Yadav

Problem Setting

Unsupervised learning aims to extract the underlying patterns from a dataset without any labels or human supervision.

In this project we implement and analyze classical methods and SOTA learning methods for unsupervised visual learning.

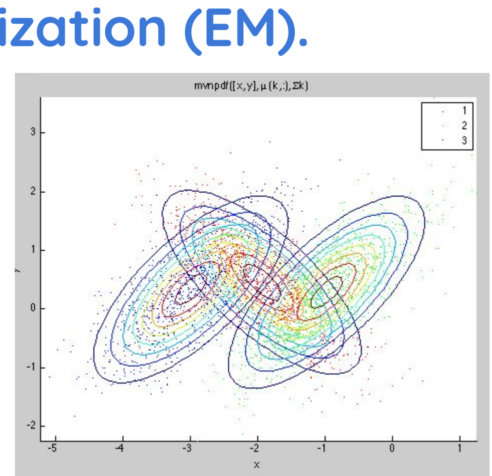
We use the accuracy metric and MNIST dataset through out our work.



K-means/GMM

K-means: K-means clusters data points through iteratively calculate centroids for images based on their euclidean distance in the pixel space. Each centroid is estimated through taking an average of all current data points inside the cluster.

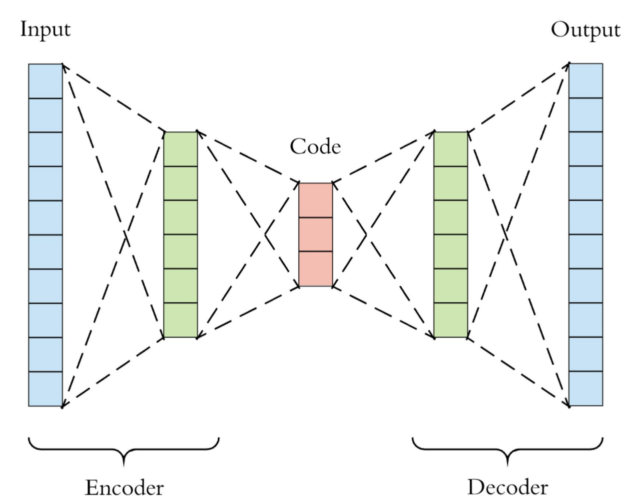
Gaussian Mixture Model: The Gaussian mixture model assumes pixels from an image are from multivariate gaussian distribution, and automatically estimate these gaussian clusters through Expectation Maximization (EM).



Autoencoder

Intuition: Autoencoder uses a bottleneck architecture to learn efficient data coding. We can apply clustering methods on the embedding space.

Architecture:



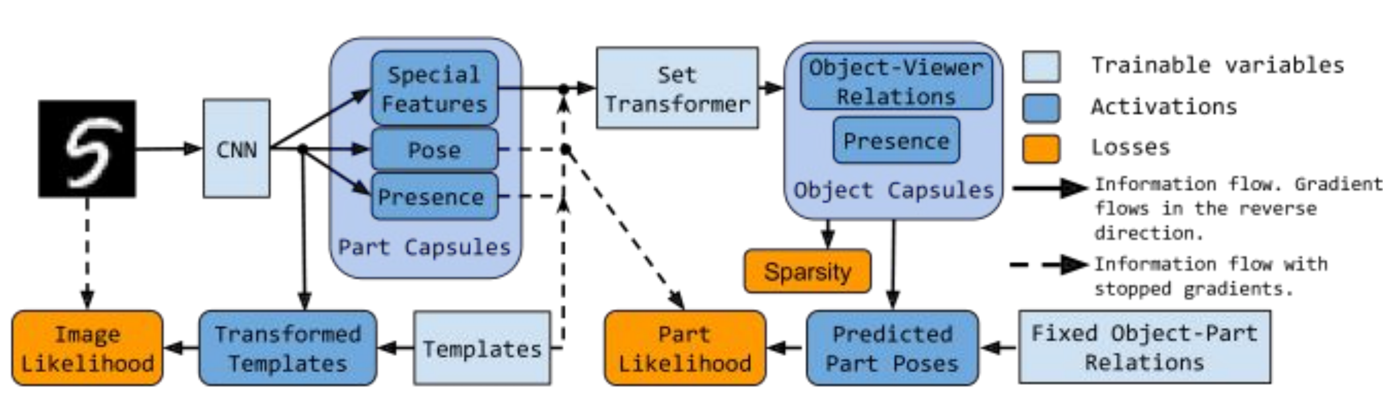
Objective:

$$\phi, \psi = \arg\min_{\phi,\psi} \|X - (\psi \circ \phi)X\|^2$$

Capsule Autoencoder

Intuition: Capsules explicitly model object pose, appearance, identity, and part-to-object relationship. Capsule autoencoders discover parts & objects through image reconstruction, estimate part poses through affine transformation, and discover objects through maximum likelihood estimation.

Architecture:



Objective:

Image likelihood (part capsules):

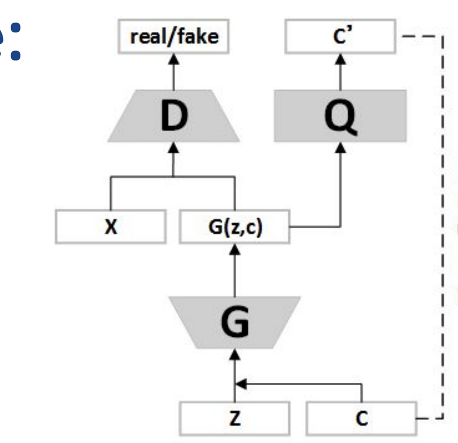
Pose likelih $p(\mathbf{y}) = \prod_{i,j} \sum_{m=1}^{M} p^{y}_{m,i,j} \mathcal{N}\left(y_{i,j} \mid \mathbf{c}_m \cdot \widehat{T}^{c}_{m,i,j}; \sigma^2_y\right)$

$$p(\mathbf{x}_{1:M}, d_{1:M}) = \prod_{m=1}^{M} \left[\sum_{k=1}^{K} \frac{a_k a_{k,m}}{\sum_i a_i \sum_j a_{i,j}} p(\mathbf{x}_m \mid k, m) \right]^{d_m}$$

InfoGAN

Intuition: InfoGAN Explicitly maximize the mutual information between generated image x and the latent code c, and thus enabling the network to decompose latent code into a set of semantically meaningful factors, e.g. digits, rotation, etc.

Architecture:



Objective:

$$\min_{G,Q} \max_{D} V_{\text{InfoGAN}}(D, G, Q) = V(D, G) - \lambda L_I(G, Q)$$

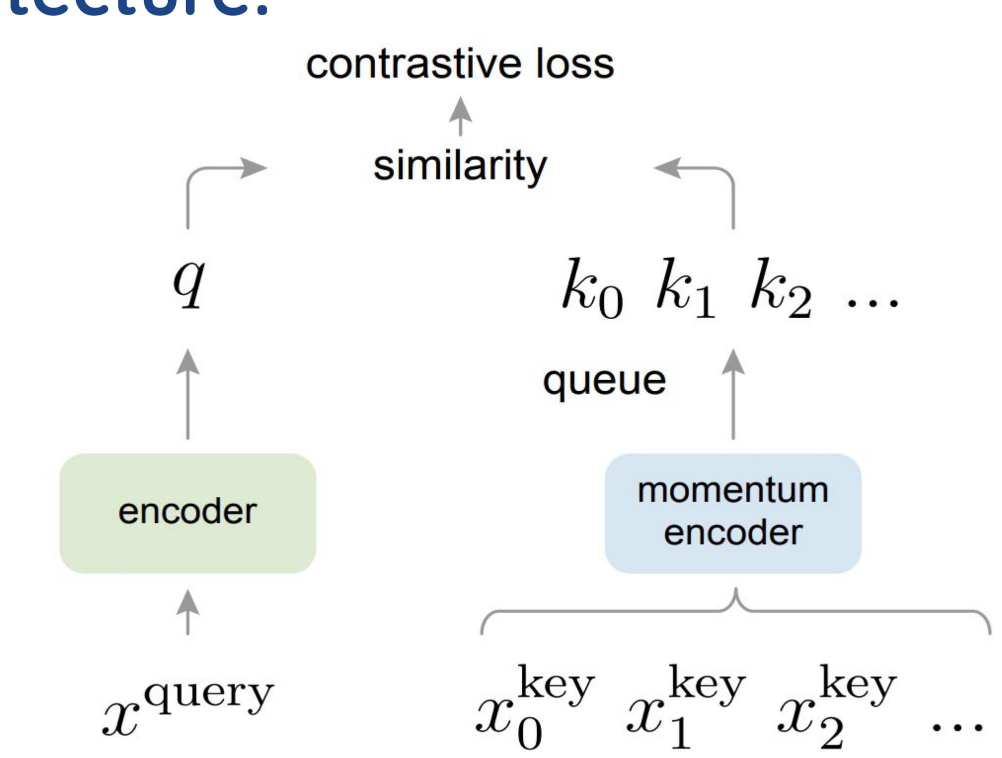
$$L_I(G, Q) = E_{c \sim P(c), x \sim G(z,c)}[\log Q(c|x)] + H(c)$$

Momentum Contrast (MoCo)

Intuition: Contrastive learning are built on the assumption that samples of the same class will have higher similarity and vice versa.

In this light, MoCo proposes to build a dynamic dict in which the key is represented by a momentum-based slowly progressing encoder.

Architecture:



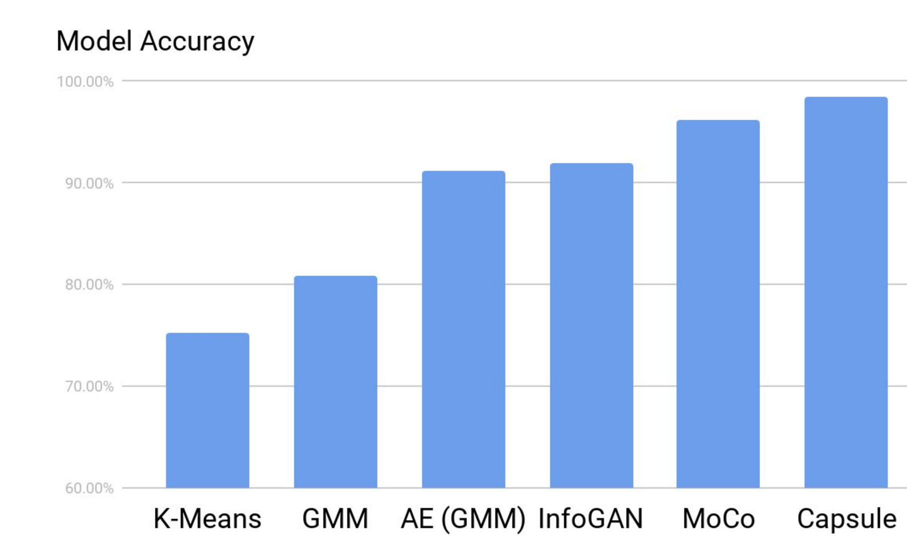
Objective:

$$\mathcal{L}_q = -\log \frac{\exp(q \cdot k_+ / \tau)}{\sum_{i=0}^{K} \exp(q \cdot k_i / \tau)}$$

$$\theta_k \leftarrow m\theta_k + (1 - m)\theta_q$$

Experiment Results

	Accuracy	Training Time
K-Means	75.2%	22s
GMM	80.8%	3m
AE (GMM)	91.2%	20m
InfoGAN	92.0%	-
MoCo	96.9%	5m
Capsule	98.5%	2880m



Ablation Studies

Table: MoCo Ablation Study

Queue Length	16	32	64	128	256
Momentum = 0.9	91.38%	95.11%	95.16%	95.31%	94.42%
Momentum = 0.99	92.98%	93.27%	**96.91%**	96.14%	95.36%
Momentum = 0.999	92.91%	94.29%	96.23%	94.87%	93.39%
Momentum = 0.9999	91.2%	93.80%	94.46%	93.67%	94.03%

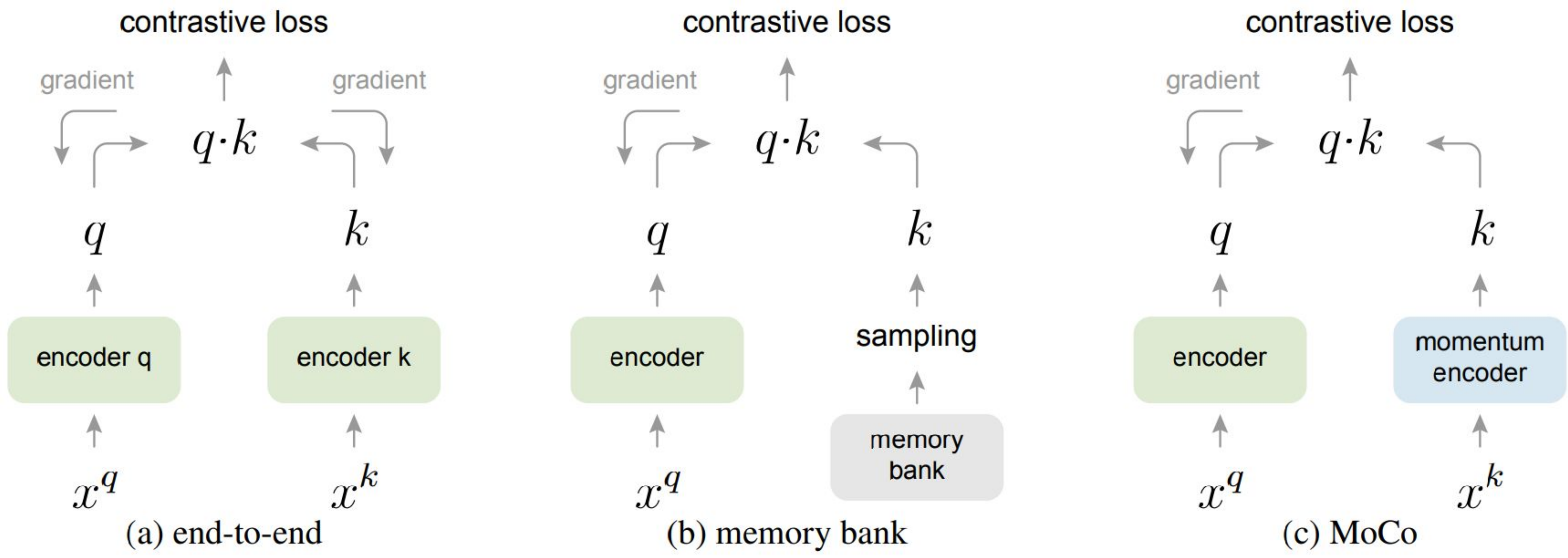
Table: Capsule Ablation Study

	Accuracy (Linear Classification)	Accuracy (Bipartite)
30 Part capsules 16 Object capsules	98.47%	97.50%
40 Part capsules 32 Object capsules	99.06%	98.54%
60 Part capsules 40 Object capsules	96.74%	91.65%

Momentum Contrast (MoCo)

Contrastive learning: are built on the assumption that samples of the same class will have higher similarity, sample of different class will have low similarity.

$$\mathcal{L}_q = -\log \frac{\exp(q \cdot k_+ / \tau)}{\sum_{i=0}^{K} \exp(q \cdot k_i / \tau)} \quad (1)$$

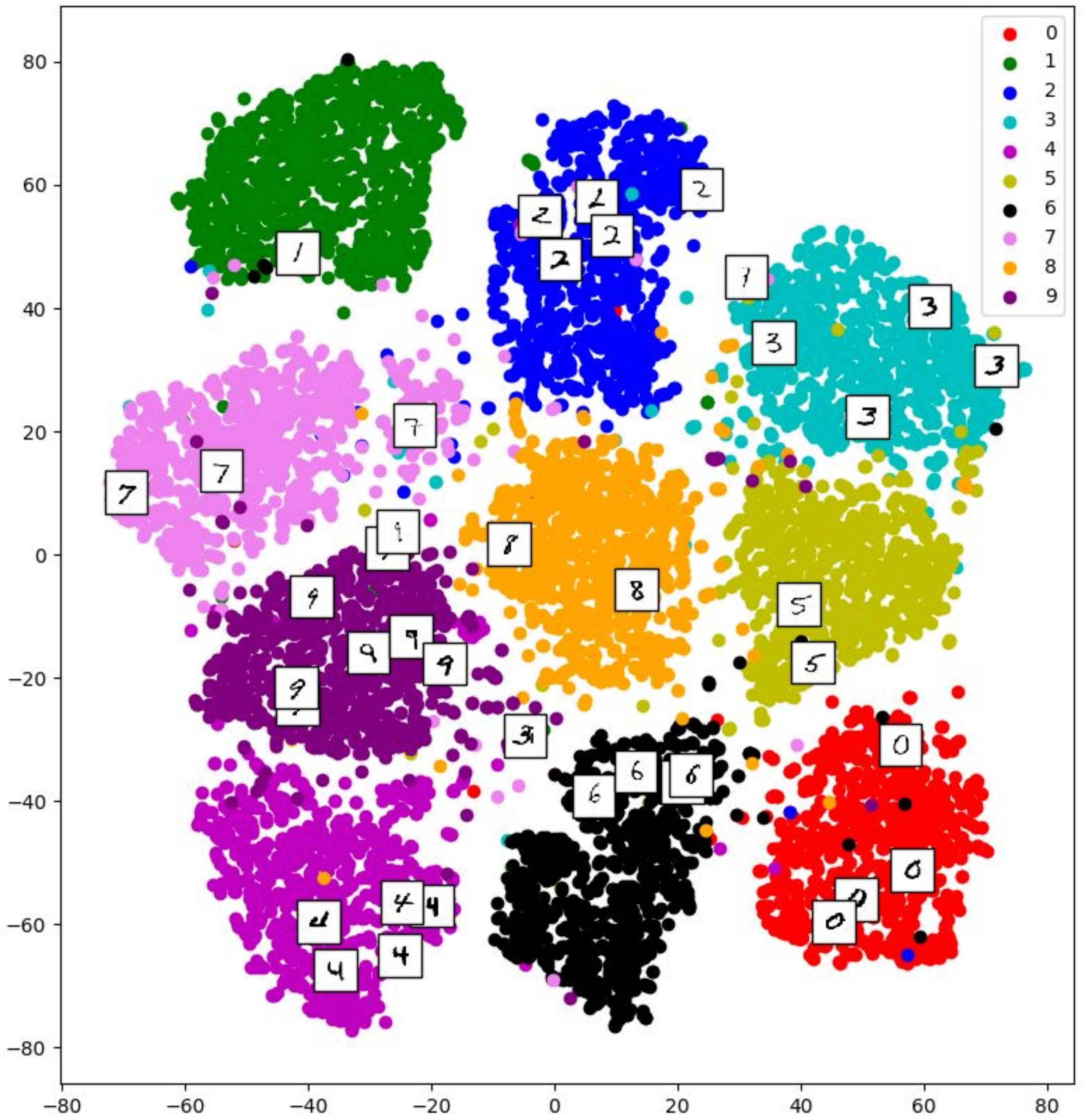


As shown in equation (1), similarity is represented with dot product. And a log softmax forces a intra-class similarity and intra-class difference.

Compared with previous end2end and memory bank methods, a queue dictionary is a much more efficient way of preserving negative samples and learning representation encoder.

We do not use ResNet-50 as in the original paper, instead we choose a very simple backbone ConvNet (2 Conv2d, 1 maxpooling followed by a FC layer) and achieve reasonably high accuracy.

The TSNE visualization of our MoCo best model is shown right. We can see that MoCo project the MNIST image into a space where samples of the same label are clustered together. Some common failure cases can also be seen in this figure, for example “7” and “9”; “6” and “0”; “3” and “5” which people sometimes make wrong are learned to be adjacent clusters.



He, Kaiming, et al. "Momentum contrast for unsupervised visual representation learning." *arXiv preprint arXiv:1911.05722* (2019).

Stacked Capsule Autoencoder:

Method:

Capsule network explicitly models part-to-object relationship and pose-to-object relationship. Each “capsule” represents a part or object, and each has an associated identity & pose & appearance latent embedding. Capsule networks are trained through maximize image reconstruction and part to object likelihood.

Part Capsule Likelihood:

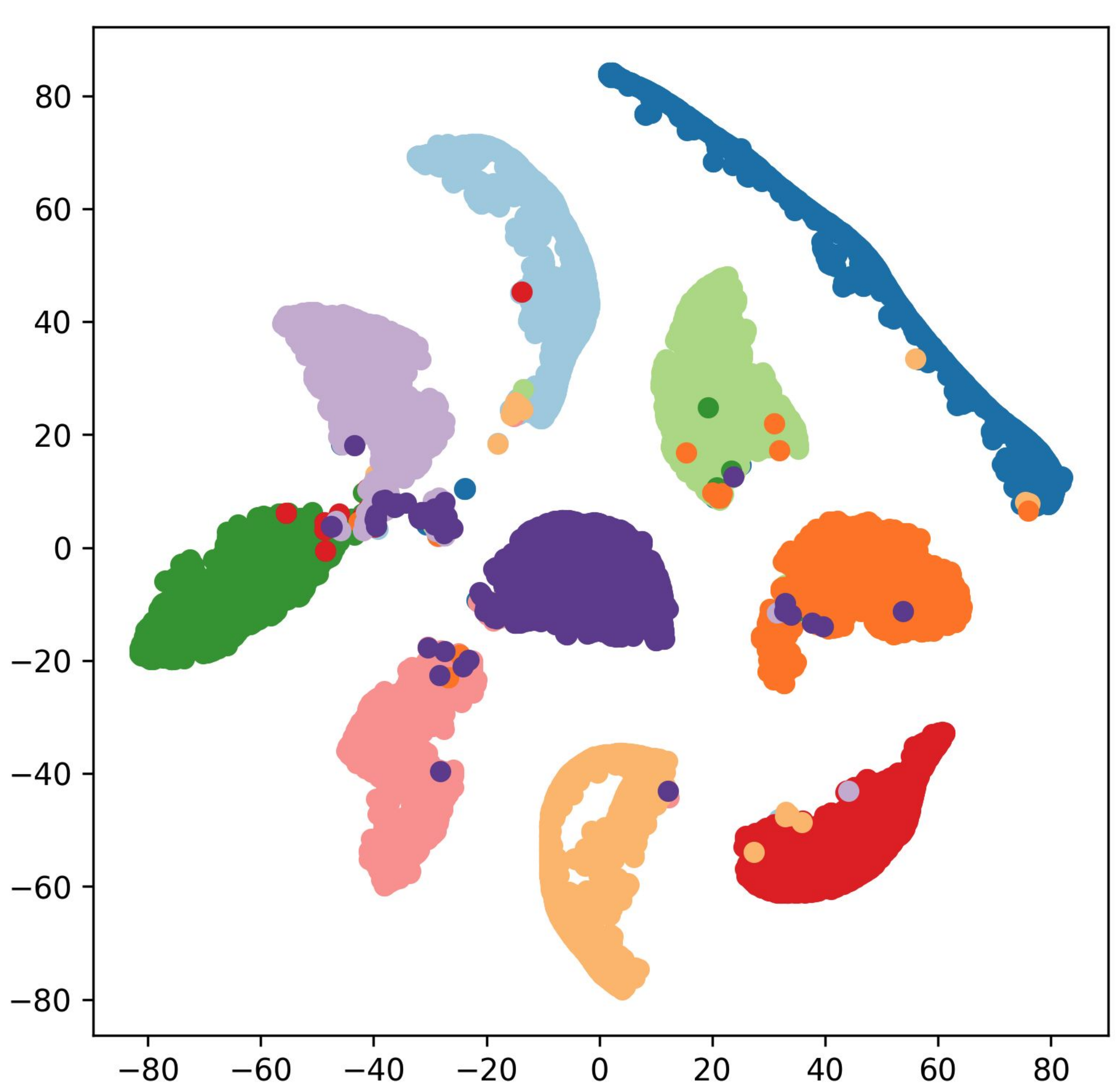
$$
\begin{aligned}
&\mathrm{OV}_{1:K}, \mathbf{c}_{1:K}, a_{1:K} = \mathrm{h}^{\mathrm{caps}}(\mathbf{x}_{1:M}) && \text{predict object capsule parameters,}\\
&\mathrm{OP}_{k,1:N}, a_{k,1:N}, \lambda_{k,1:N} = \mathrm{h}^{\mathrm{part}}_{\mathrm{k}}(\mathbf{c}_k) && \text{decode candidate parameters from } c_k\text{'s,}\\
&V_{k,n} = \mathrm{OV}_k \mathrm{OP}_{k,n} && \text{decode a part pose candidate,}\\
&p(\mathbf{x}_m \mid k,n) = \mathcal{N}(\mathbf{x}_m \mid \mu_{k,n}, \lambda_{k,n}) && \text{turn candidates into mixture components,}
\end{aligned}
$$

$$
p(\mathbf{x}_{1:M}) = \prod_{m=1}^{M} \sum_{k=1}^{K} \sum_{n=1}^{N} \frac{a_k a_{k,n}}{\sum_i a_i \sum_j a_{i,j}} p(\mathbf{x}_m \mid k,n)\,.
$$

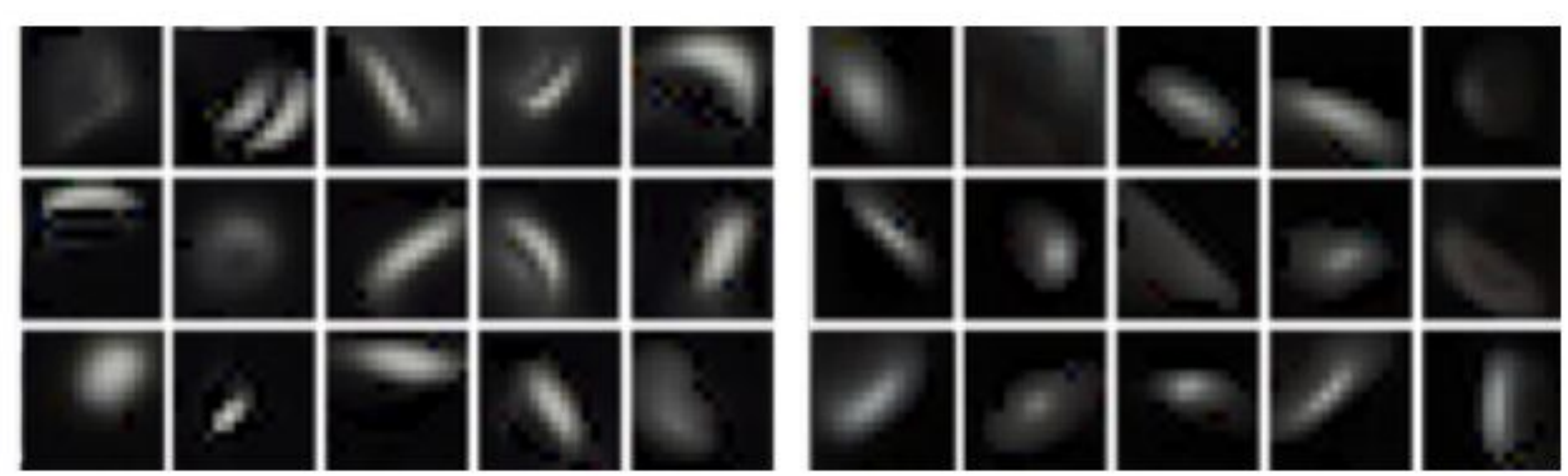
Object Capsule likelihood

$$
\begin{aligned}
&\mathbf{x}_{1:M}, d_{1:M}, \mathbf{z}_{1:M} = \mathrm{h}^{\mathrm{enc}}(\mathbf{y}) && \text{predict part capsule parameters,}\\
&\mathbf{c}_m = \mathrm{MLP}(\mathbf{z}_m) && \text{predict the color of the m}^{\text{th}}\text{ template,}\\
&\widehat{T}_m = \mathrm{TransformImage}(T_m, \mathbf{x}_m) && \text{apply affine transforms to image templates,}\\
&p^y_{m,i,j} \propto d_m \widehat{T}^a_{m,i,j} && \text{compute mixing probabilities,}\\
&p(\mathbf{y}) = \prod_{i,j} \sum_{m=1}^{M} p^y_{m,i,j} \mathcal{N}\left(y_{i,j} \mid \mathbf{c}_m \cdot \widehat{T}^c_{m,i,j}; \sigma^2_y\right) && \text{calculate the image likelihood.}
\end{aligned}
$$

TSNE Visualization of trained *identity embeddings* for object capsule:



Visualization of trained part capsule’s *appearance embedding*



Reference: Kosiorek, A. R., Sabour, S., Teh, Y. W., & Hinton, G. E. (2019). Stacked Capsule Autoencoders. (NeurIPS).

InfoGAN:

Method:

GAN: a minimax game between Generator (G) and Discriminator (D).

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

InfoGAN: explicitly maximize the mutual information between generated image “x” and the latent code “c”, and thus enabling the network to decompose latent code into a set of semantically meaningful factors, e.g. digits, rotation, etc.

Practically: Estimate $P(c|x)$ with neural network “Q”, sample P(c) with discrete uniform distribution

Implementation Detail:

discriminator D **/ recognition network** Q	**generator** G
Input 28 × 28 Gray image	Input $\in \mathbb{R}^{74}$
4 × 4 conv. 64 lRELU. stride 2	FC. 1024 RELU. batchnorm
4 × 4 conv. 128 lRELU. stride 2. batchnorm	FC. 7 × 7 × 128 RELU. batchnorm
FC. 1024 lRELU. batchnorm	4 × 4 upconv. 64 RELU. stride 2. batchnorm
FC. output layer for D, FC.128-batchnorm-lRELU-FC.output for Q	4 × 4 upconv. 1 channel

Visualization:

Reference: Hinton, G. E., & Zemel, R. S. (1994). Autoencoders, minimum description length and Helmholtz free energy. In Advances in neural information processing systems 6 (pp. 3-10)

Autoencoder

Method: Autoencoder consists of a encoder and a decoder, each with several fully connected or convolution layers.
It uses a bottleneck architecture to learn efficient data codings in an unsupervised manner. We usually apply a l2 loss between the input and output to enforce data reconstruction.

Implementation Detail:

For the encoder, we use two fully connected layer with the hidden layer dimension 128 and embedding dimension 12. The decoder architecture is the inverse of the encoder. We use MSE loss with Adam Optimizer.

Visualization of the reconstruction:

Reference: Chen, X., Duan, Y., Houthooft, R., Schulman, J., Sutskever, I., & Abbeel, P. (n.d.). InfoGAN : Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets.