Learning Without Labels

Unsupervised and Weakly Supervised Learning of Deep Models

Presented by Shazia Akbar
shazia@altislabs.com

AISC Premium Workshop 2019

Last time...

Types of learning

Unsupervised learning

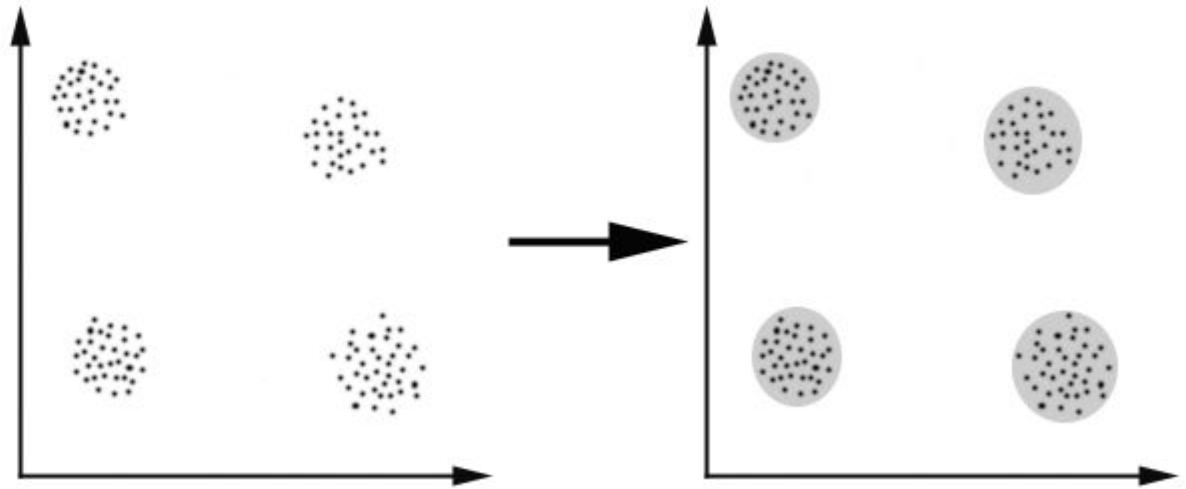
- Autoencoders
- Variational Autoencoders
    - Beta VAE (Disentangled features)
- GANs

Outline

- Traditional clustering
- Deep Clustering (VaDE)
    - Other clustering techniques
    - Non-parametric clustering
- Predictive Modeling
- Summarize

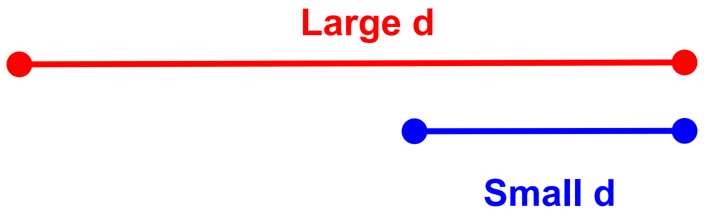
Clustering

A cluster is a collection of objects which are “similar” between them and are “dissimilar” to the objects belonging to other clusters.

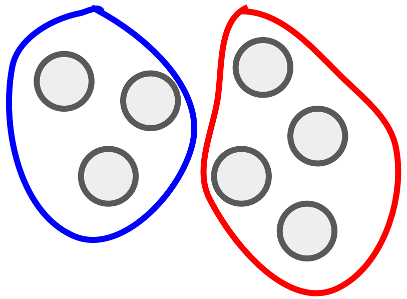


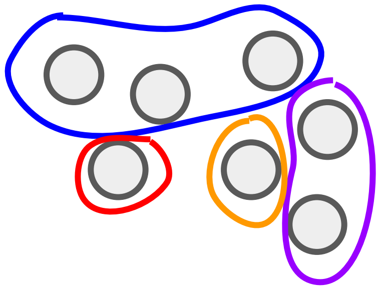
Key components of clustering algorithm

1) Proximity measure: either similarity or dissimilarity



1) Criterion function to evaluate a clustering


“Good” Clustering


“Bad” Clustering

KMeans Clustering

Simplest clustering technique

Assumes euclidean space

Iteratively

1. Measures the distance between cluster centers and instances
2. Assign cluster labels depending on minimum distance
3. Update cluster centers (mean); repeat until no further changes occur

Randomly assigned cluster centers to begin with

Expectation-Maximization (EM)

Optimize fit between data and a model (usually Gaussian)

Maximizes a likelihood function when some of the variables in your model are unobserved

1. Guess parameters to estimate model parameters
2. **E Step:**
    a. Use the current model to estimate label for unobserved data
    b. Calculate expectation of likelihood function with respect to unobserved data
3. **M Step:** Update model
4. Repeat until convergence

Expectation-Maximization (EM)

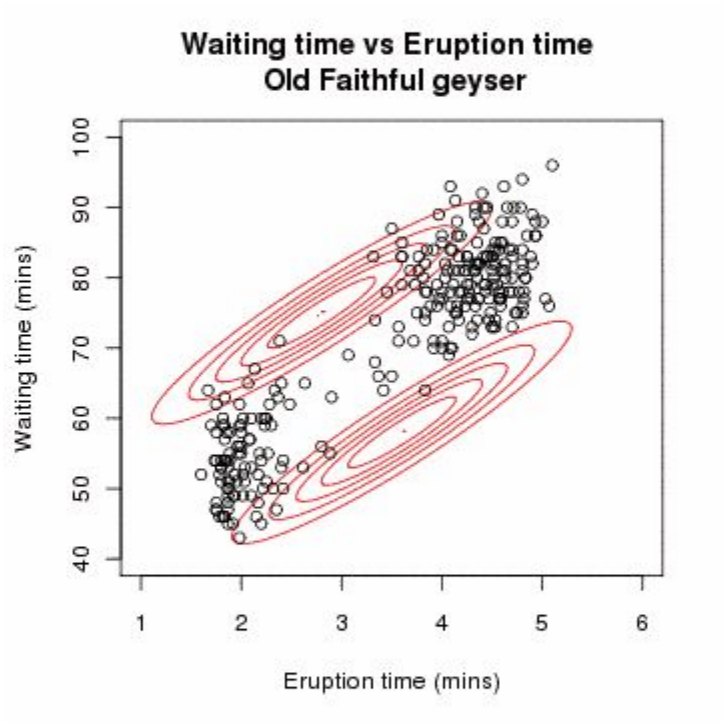
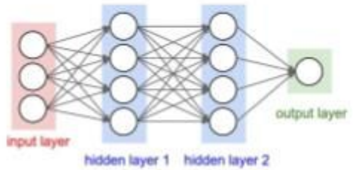
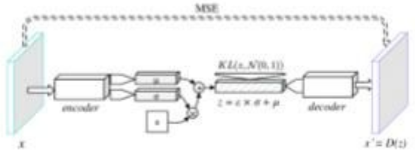


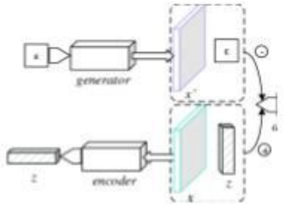
Image credits: Wikipedia)

Deep Clustering

DNN







Network Loss

$$H((x_1, y_1), D) = -y_1 \log D(x_1) - (1 - y_1) \log(1 - D(x_1))$$

$$L_R = ||\boldsymbol{x} - g_\phi(f_\theta(\boldsymbol{x}))||^2$$

Clustering Loss

$$q_{ij} = \frac{\left(1+\|z_i-\mu_j\|^2/\alpha\right)^{-\frac{\alpha+1}{2}}}{\sum_{j'}\left(1+\|z_i-\mu_{j'}\|^2/\alpha\right)^{-\frac{\alpha+1}{2}}}$$

$$\|\boldsymbol{f}(\boldsymbol{x}_i) - \boldsymbol{M}\boldsymbol{s}_i\|_2^2$$

$$\mathcal{L}_{\text{ELBO}}(\mathbf{x}) = E_{q(\mathbf{z},c|\mathbf{x})}[\log p(\mathbf{x}|\mathbf{z})] - D_{KL}(q(\mathbf{z},c|\mathbf{x})\|p(\mathbf{z},c))$$

Performance Metrics

Widely used metrics include:

Unsupervised clustering accuracy (ACC)

$$ACC = \max_m \frac{\sum_{i=1}^{n} \mathbf{1}\{y_i = m(c_i)\}}{n}$$

Similar to classification accuracy but with a mapping function, *m* to match cluster to labels

Performance Metrics

Widely used metrics include:

Normalized Mutual Information (NMI)

$$NMI(Y,C) = \frac{I(Y,C)}{\frac{1}{2}[H(Y) + H(C)]}$$

A measure of correlation between two disjoint clusters

`sklearn.metrics.`**`normalized_mutual_info_score`**

Performance Metrics

Widely used metrics include:

Adjusted Rand Index (ARI)

$$ARI = \frac{\sum_{ij}\binom{n_{ij}}{2} - [\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2}]/\binom{n}{2}}{\frac{1}{2}[\sum_i \binom{a_i}{2} + \sum_j \binom{b_j}{2}] - [\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2}]/\binom{n}{2}}$$

Considers all pairs of samples in same or different clusters

`sklearn.metrics.`**`adjusted_rand_score`**

Deep Clustering Algorithms

Received May 12, 2018, accepted July 2, 2018, date of publication July 17, 2018, date of current version August 7, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2855437

A Survey of Clustering With Deep Learning: From the Perspective of Network Architecture

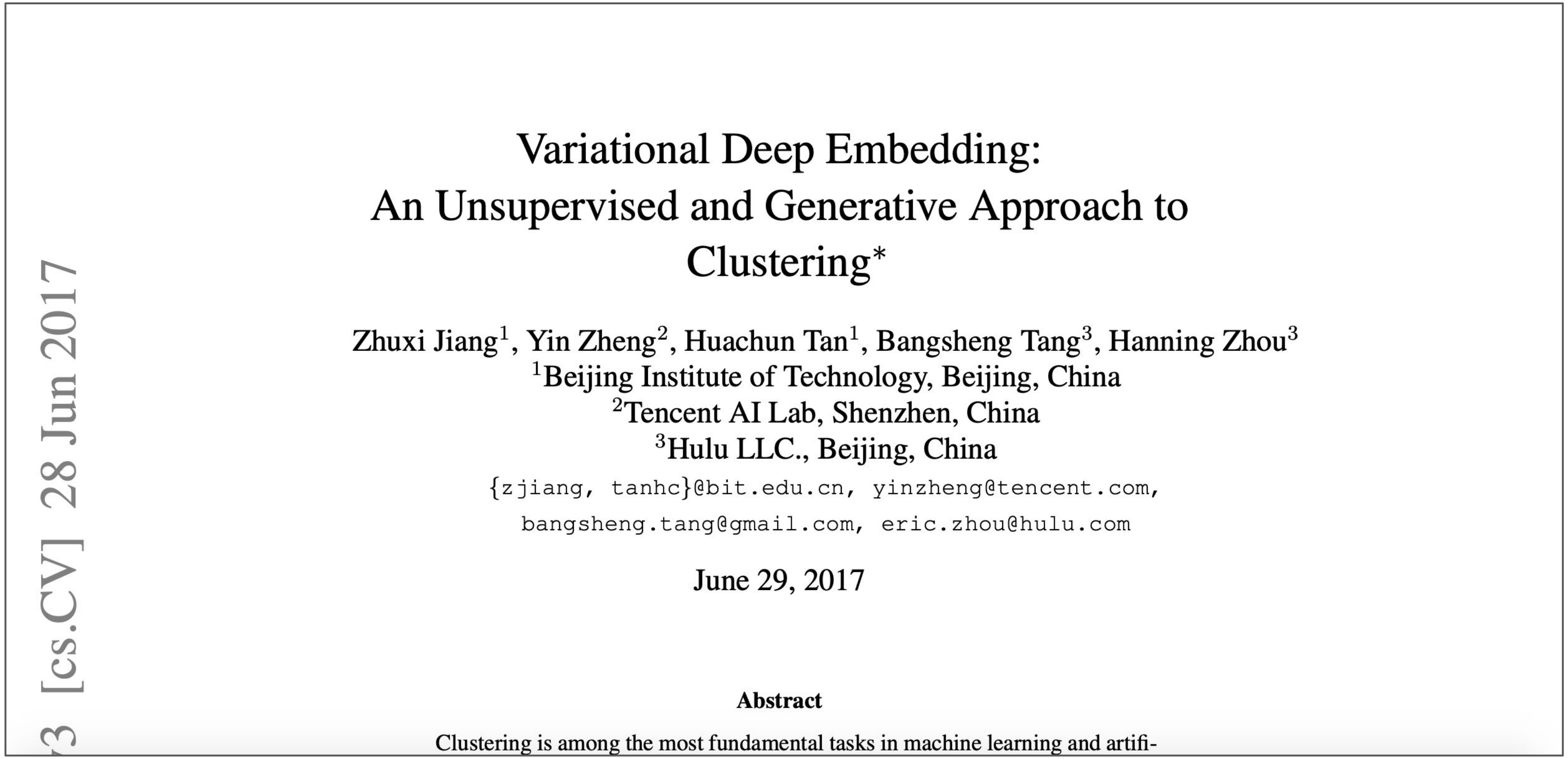
ERXUE MIN, XIFENG GUO, QIANG LIU, (Member, IEEE), GEN ZHANG, JIANJING CUI, AND JUN LONG
College of Computer, National University of Defense Technology, Changsha 410073, China

Corresponding authors: Erxue Min (minerxue12@nudt.edu.cn) and Qiang Liu (qiangliu06@nudt.edu.cn)

This work was supported by the National Natural Science Foundation of China under Grant 60970034, Grant 61105050, Grant 61702539, and Grant 6097003.

ABSTRACT Clustering is a fundamental problem in many data-driven application domains, and clustering

VaDE



3 [cs.CV] 28 Jun 2017

Variational Deep Embedding:
An Unsupervised and Generative Approach to Clustering*

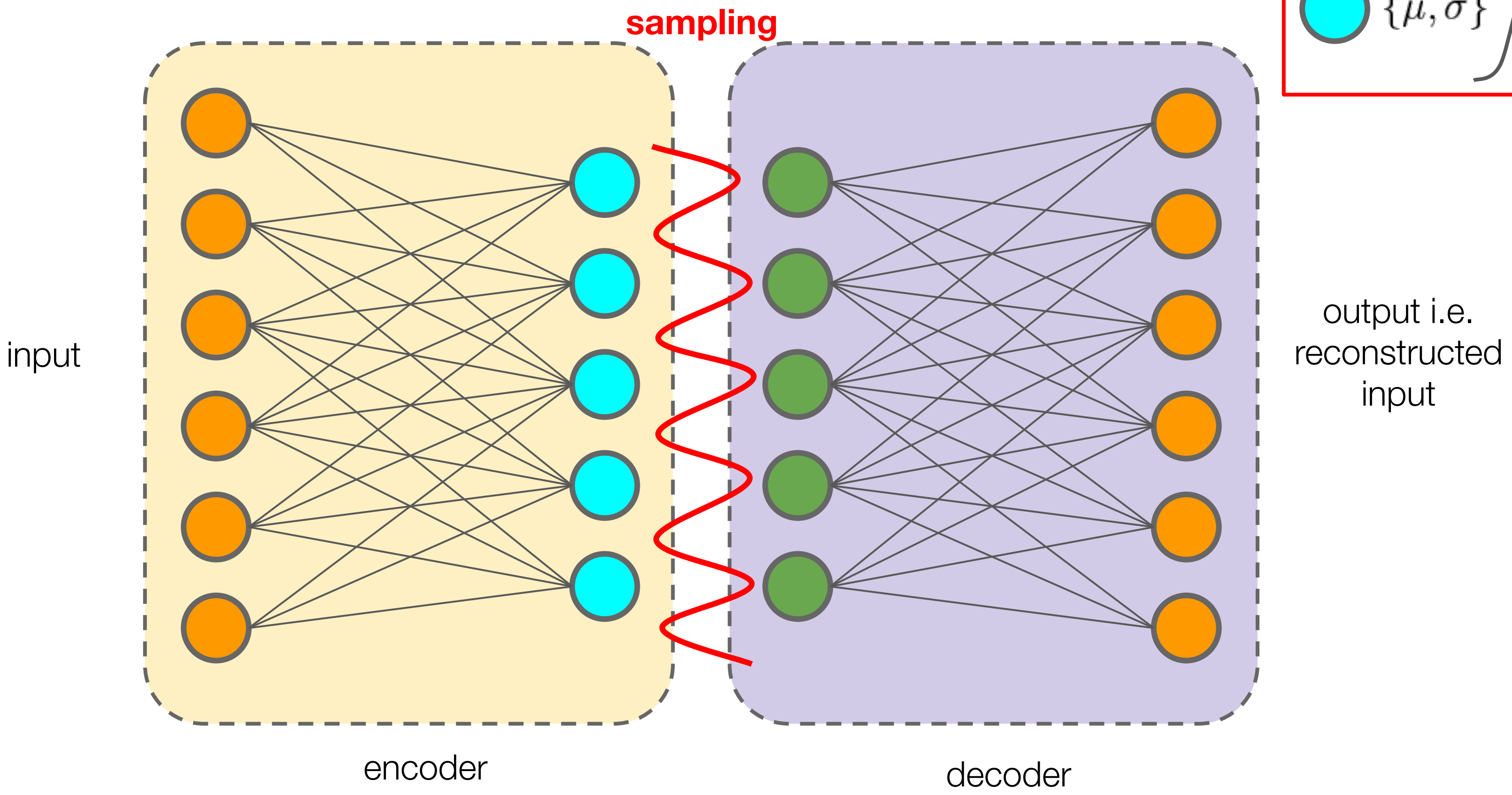
Zhuxi Jiang[1], Yin Zheng[2], Huachun Tan[1], Bangsheng Tang[3], Hanning Zhou[3]
[1]Beijing Institute of Technology, Beijing, China
[2]Tencent AI Lab, Shenzhen, China
[3]Hulu LLC., Beijing, China
{zjiang, tanhc}@bit.edu.cn, yinzheng@tencent.com,
bangsheng.tang@gmail.com, eric.zhou@hulu.com

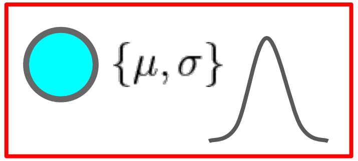
June 29, 2017

Abstract

Clustering is among the most fundamental tasks in machine learning and artifi-

Quick Recap of Variational Autoencoder



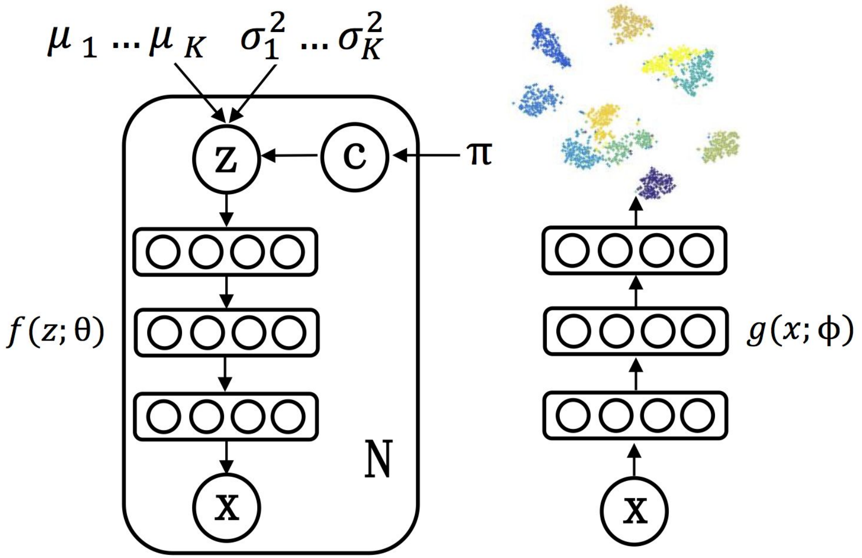


VaDE

Variational Deep Embedding

Performs 2 tasks:

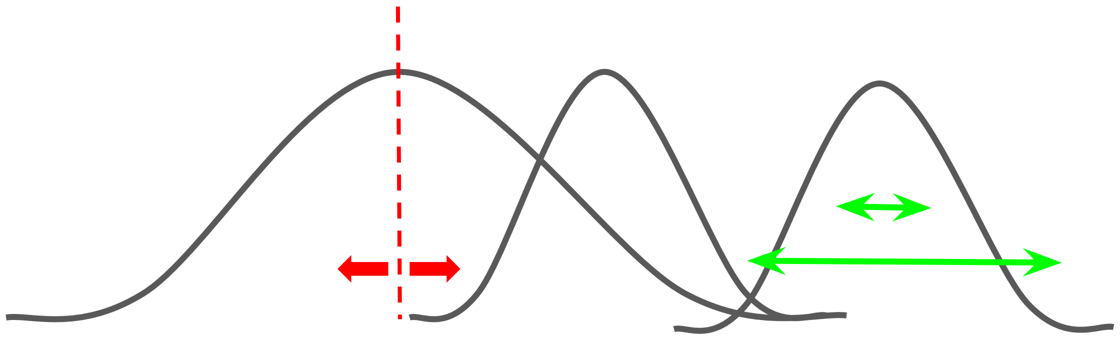
1. Clusters data by learning a deep representation (right)
2. Generates new samples (left)



https://arxiv.org/pdf/1611.05148.pdf

GMM

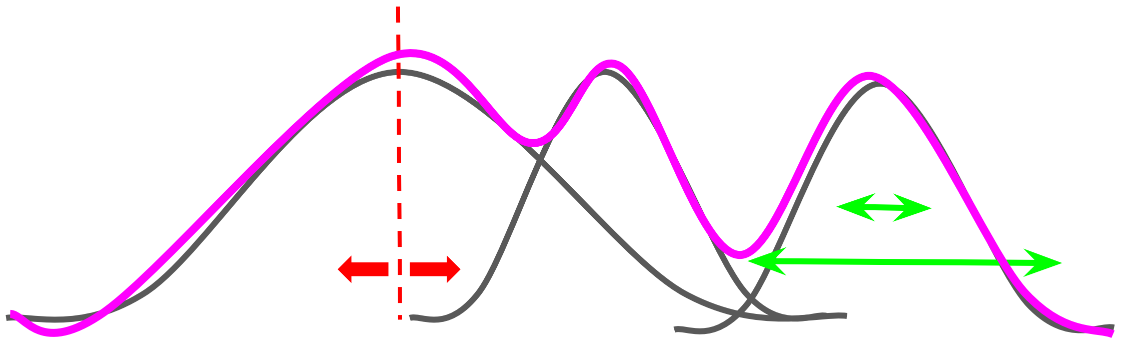
Series of unit Gaussians which fit to the data, so that we can eventually sample from it and get something similar to our training set



We may very well use EM to update this model as well!

GMM

Series of unit Gaussians which fit to the data, so that we can eventually sample from it and get something similar to our training set



We may very well use EM to update this model as well!

VaDE: GMM

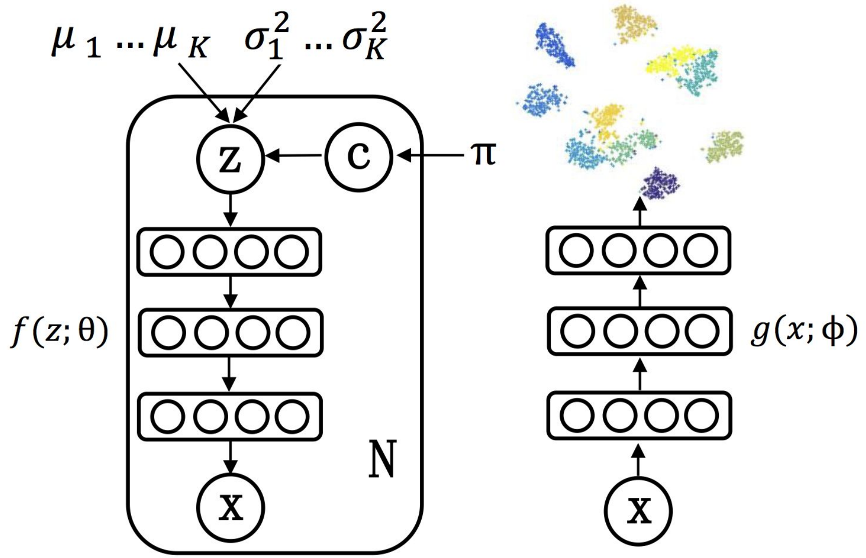
All three of these parameters can be learning in a single GMM

```
gmm = GaussianMixture(n_components=10, covariance_type='diag')
gmm.fit(z)


_pi = torch.log(torch.from_numpy(gmm.weights_)).float()
mu = torch.from_numpy(gmm.means_).float()
logvar = torch.log(torch.from_numpy(gmm.covariances_)).float()
```

VaDE

1. Cluster is picked from Gaussian Mixture Model prior
2. **z** is Sampled based on cluster selected
3. DNN1 decodes **z** to an observation **x**
4. DNN2 encodes **x** into a latent representation **z** (optimization step)

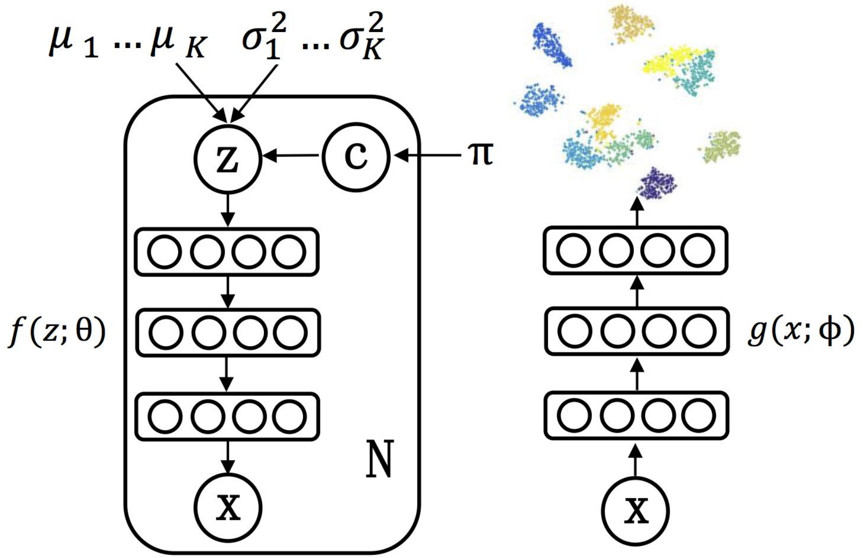


https://arxiv.org/pdf/1611.05148.pdf

VaDE: GMM

π : Categorical distribution parameters in GMM (learned)

μ, σ: mean and variance of Gaussian distribution (length = number of clusters)



https://arxiv.org/pdf/1611.05148.pdf

VaDE: Loss Function

$$q(\mathbf{z}, c|\mathbf{x}) = q(\mathbf{z}|\mathbf{x})q(c|\mathbf{x}).$$

$$\begin{aligned}
\mathcal{L}_{\mathrm{ELBO}}(\mathbf{x}) &= E_{q(\mathbf{z},c|\mathbf{x})}\left[\log \frac{p(\mathbf{x}, \mathbf{z}, c)}{q(\mathbf{z}, c|\mathbf{x})}\right] \\
&= E_{q(\mathbf{z},c|\mathbf{x})}\left[\log p(\mathbf{x}, \mathbf{z}, c) - \log q(\mathbf{z}, c|\mathbf{x})\right] \\
&= E_{q(\mathbf{z},c|\mathbf{x})}[\log p(\mathbf{x}|\mathbf{z}) + \log p(\mathbf{z}|c) \\
&\quad + \log p(c) - \log q(\mathbf{z}|\mathbf{x}) - \log q(c|\mathbf{x})]
\end{aligned}$$

VaDE: Loss Function

$$\boxed{q(\mathbf{z}, c|\mathbf{x}) = q(\mathbf{z}|\mathbf{x})q(c|\mathbf{x}).}$$

$$\begin{aligned}
\mathcal{L}_{\text{ELBO}}(\mathbf{x}) &= E_{q(\mathbf{z},c|\mathbf{x})}\left[\log \frac{p(\mathbf{x}, \mathbf{z}, c)}{\boxed{q(\mathbf{z}, c|\mathbf{x})}}\right] \\
&= E_{q(\mathbf{z},c|\mathbf{x})}\left[\log p(\mathbf{x}, \mathbf{z}, c) - \log q(\mathbf{z}, c|\mathbf{x})\right] \\
&= E_{q(\mathbf{z},c|\mathbf{x})}[\log p(\mathbf{x}|\mathbf{z}) + \log p(\mathbf{z}|c) \\
&\quad + \log p(c) - \log q(\mathbf{z}|\mathbf{x}) - \log q(c|\mathbf{x})]
\end{aligned}$$

VaDE: Loss Function

$$\begin{aligned}\mathcal{L}_{\text{ELBO}}(\mathbf{x}) =& \frac{1}{L}\sum_{l=1}^{L}\sum_{i=1}^{D} x_i \log \boldsymbol{\mu}_x^{(l)}|_i + (1 - x_i)\log(1 - \boldsymbol{\mu}_x^{(l)}|_i) \\ &- \frac{1}{2}\sum_{c=1}^{K}\gamma_c \sum_{j=1}^{J}(\log \boldsymbol{\sigma}_c^2|_j + \frac{\tilde{\boldsymbol{\sigma}}^2|_j}{\boldsymbol{\sigma}_c^2|_j} + \frac{(\tilde{\boldsymbol{\mu}}|_j - \boldsymbol{\mu}_c|_j)^2}{\boldsymbol{\sigma}_c^2|_j}) \\ &+ \sum_{c=1}^{K}\gamma_c \log \frac{\pi_c}{\gamma_c} + \frac{1}{2}\sum_{j=1}^{J}(1 + \log \tilde{\boldsymbol{\sigma}}^2|_j)\end{aligned}$$

VaDE: Loss Function

Cross entropy

$$
\begin{aligned}
\mathcal{L}_{\text{ELBO}}(\mathbf{x}) =& \boxed{\frac{1}{L}\sum_{l=1}^{L}\sum_{i=1}^{D} x_i \log \boldsymbol{\mu}_x^{(l)}|_i + (1-x_i)\log(1-\boldsymbol{\mu}_x^{(l)}|_i)} \\
&-\frac{1}{2}\sum_{c=1}^{K}\gamma_c \sum_{j=1}^{J}(\log \boldsymbol{\sigma}_c^2|_j + \frac{\tilde{\boldsymbol{\sigma}}^2|_j}{\boldsymbol{\sigma}_c^2|_j} + \frac{(\tilde{\boldsymbol{\mu}}|_j - \boldsymbol{\mu}_c|_j)^2}{\boldsymbol{\sigma}_c^2|_j}) \\
&+\sum_{c=1}^{K}\gamma_c \log \frac{\pi_c}{\gamma_c} + \frac{1}{2}\sum_{j=1}^{J}(1+\log \tilde{\boldsymbol{\sigma}}^2|_j)
\end{aligned}
$$

VaDE: Loss Function

Cross entropy

$$
\begin{aligned}
\mathcal{L}_{\mathrm{ELBO}}(\mathbf{x}) = & \boxed{\frac{1}{L}\sum_{l=1}^{L}\sum_{i=1}^{D} x_i \log \boldsymbol{\mu}_x^{(l)}|_i + (1-x_i)\log(1-\boldsymbol{\mu}_x^{(l)}|_i)} \\
& -\frac{1}{2}\sum_{c=1}^{K}\gamma_c \sum_{j=1}^{J} \boxed{\left(\log \boldsymbol{\sigma}_c^2|_j + \frac{\tilde{\boldsymbol{\sigma}}^2|_j}{\boldsymbol{\sigma}_c^2|_j} + \frac{(\tilde{\boldsymbol{\mu}}|_j - \boldsymbol{\mu}_c|_j)^2}{\boldsymbol{\sigma}_c^2|_j}\right)} \\
& +\sum_{c=1}^{K}\gamma_c \log\frac{\pi_c}{\gamma_c} + \frac{1}{2}\sum_{j=1}^{J}(1+\log\tilde{\boldsymbol{\sigma}}^2|_j)
\end{aligned}
$$

Comparison of cluster *c* to all clusters

VaDE: Loss Function

Cross entropy

$$\mathcal{L}_{\text{ELBO}}(\mathbf{x}) = \boxed{\frac{1}{L}\sum_{l=1}^{L}\sum_{i=1}^{D} x_i \log \boldsymbol{\mu}_x^{(l)}|_i + (1-x_i)\log(1-\boldsymbol{\mu}_x^{(l)}|_i)}$$

$$- \frac{1}{2}\sum_{c=1}^{K}\gamma_c\sum_{j=1}^{J}\boxed{(\log\boldsymbol{\sigma}_c^2|_j + \frac{\tilde{\boldsymbol{\sigma}}^2|_j}{\boldsymbol{\sigma}_c^2|_j} + \frac{(\tilde{\boldsymbol{\mu}}|_j - \boldsymbol{\mu}_c|_j)^2}{\boldsymbol{\sigma}_c^2|_j})}$$

$$+ \sum_{c=1}^{K}\gamma_c \boxed{\log\frac{\pi_c}{\gamma_c}} + \frac{1}{2}\sum_{j=1}^{J}(1+\log\tilde{\boldsymbol{\sigma}}^2|_j)$$

Comparison of cluster *c* to all clusters

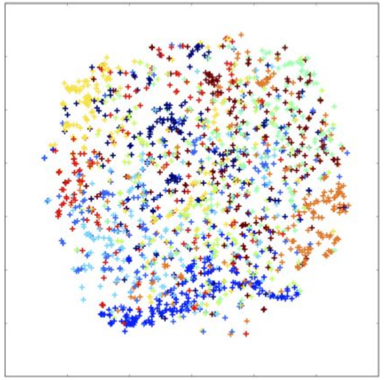
p(c)

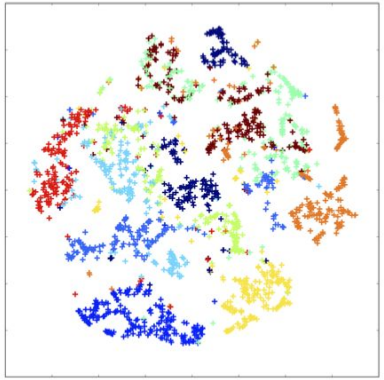
Implementation of Loss

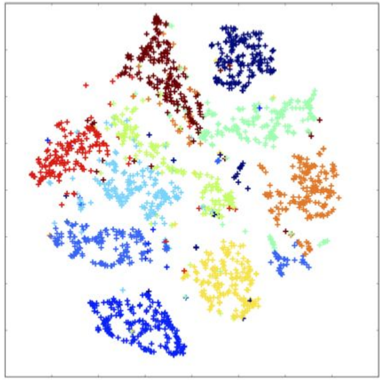
During the practical, refer to original implementation:
https://github.com/slim1017/VaDE

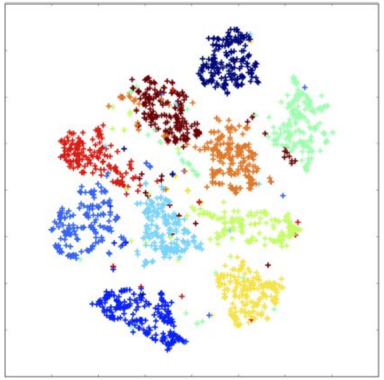
The code is in tensorflow - try and understand it and translate it to PyTorch :)

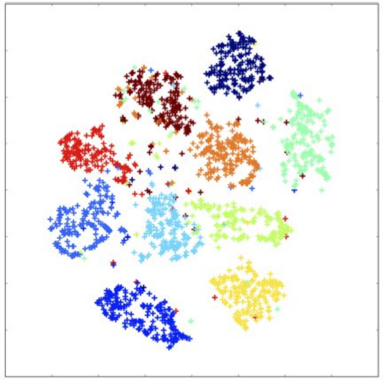
VaDE

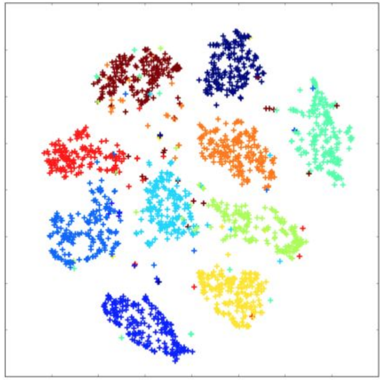

(a) Epoch 0 (11.35%)


(b) Epoch 1 (55.63%)


(c) Epoch 5 (72.40%)

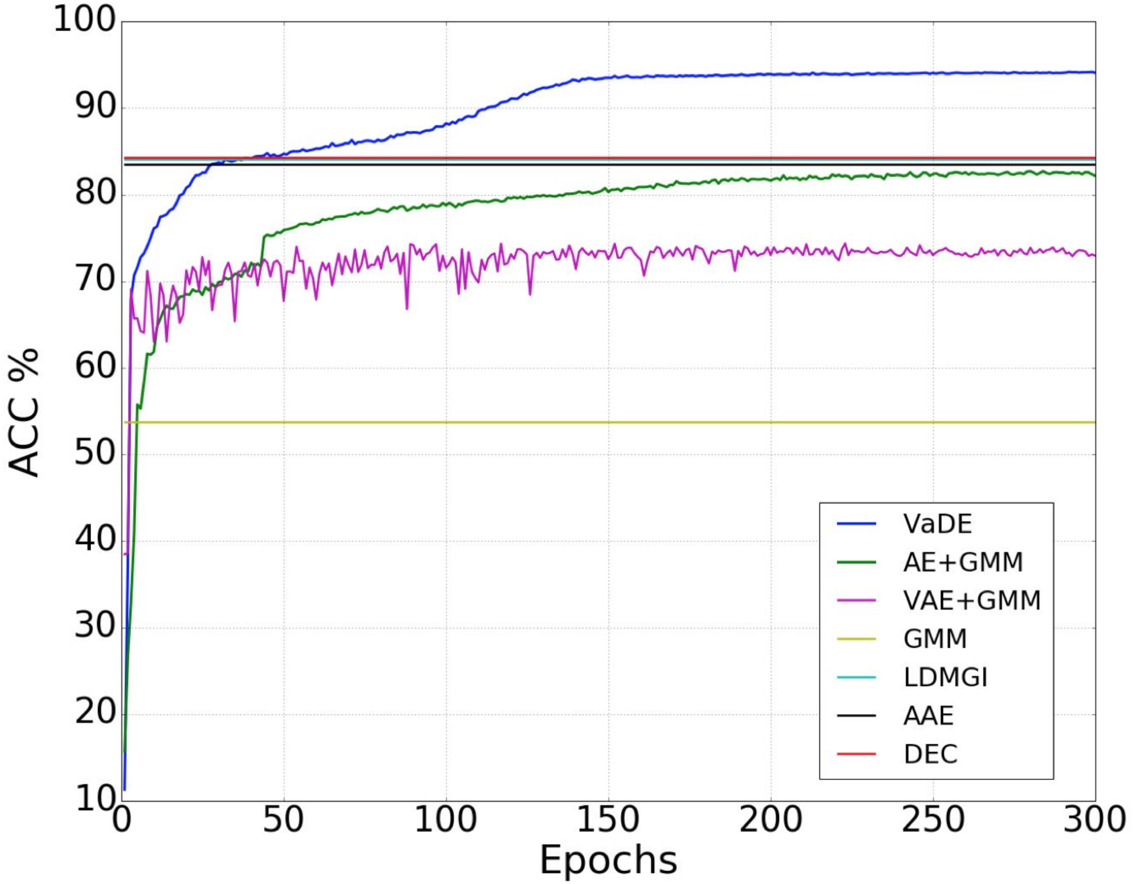

(d) Epoch 50 (84.59%)


(e) Epoch 120 (90.76%)


(f) Epoch End (94.46%)

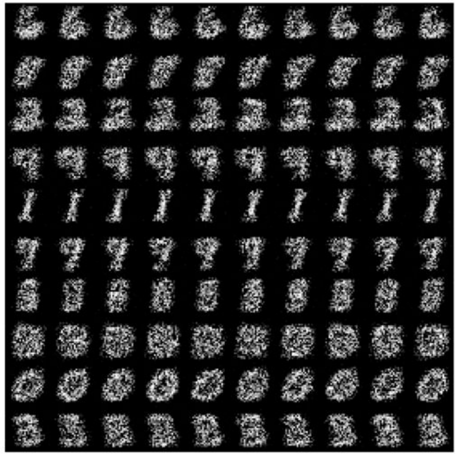
https://arxiv.org/pdf/1611.05148.pdf

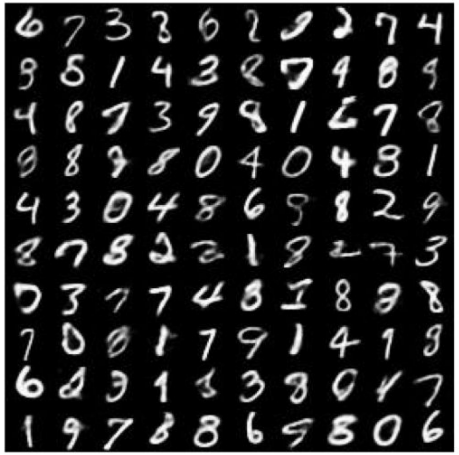
VaDE

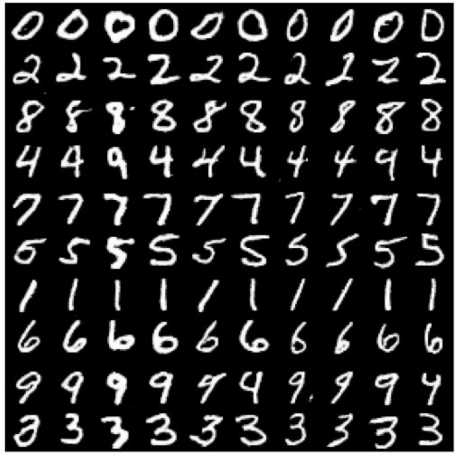


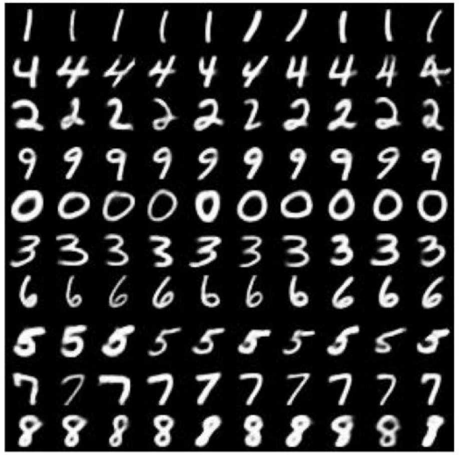
ACC %
Epochs
100
90
80
70
60
50
40
30
20
10
0
50
100
150
200
250
300
VaDE
AE+GMM
VAE+GMM
GMM
LDMGI
AAE
DEC

VaDE


(a) GMM


(b) VAE


(c) InfoGAN


(d) VaDE

Weakness of VaDE

“Similar to other VAE-based models...VaDE suffers from the problem that the reconstruction term in Equation 17 would be so weak in the beginning of training that the model might get stuck in an undesirable local minima or saddle point, from which it is hard to escape. **In this work, pretraining is used to avoid this problem**. Specifically, we use a Stacked Auto- Encoder to pretrain the networks f and g.”

Week 2 Exercise

Build a VAE, including loss function in PyTorch

Other clustering techniques

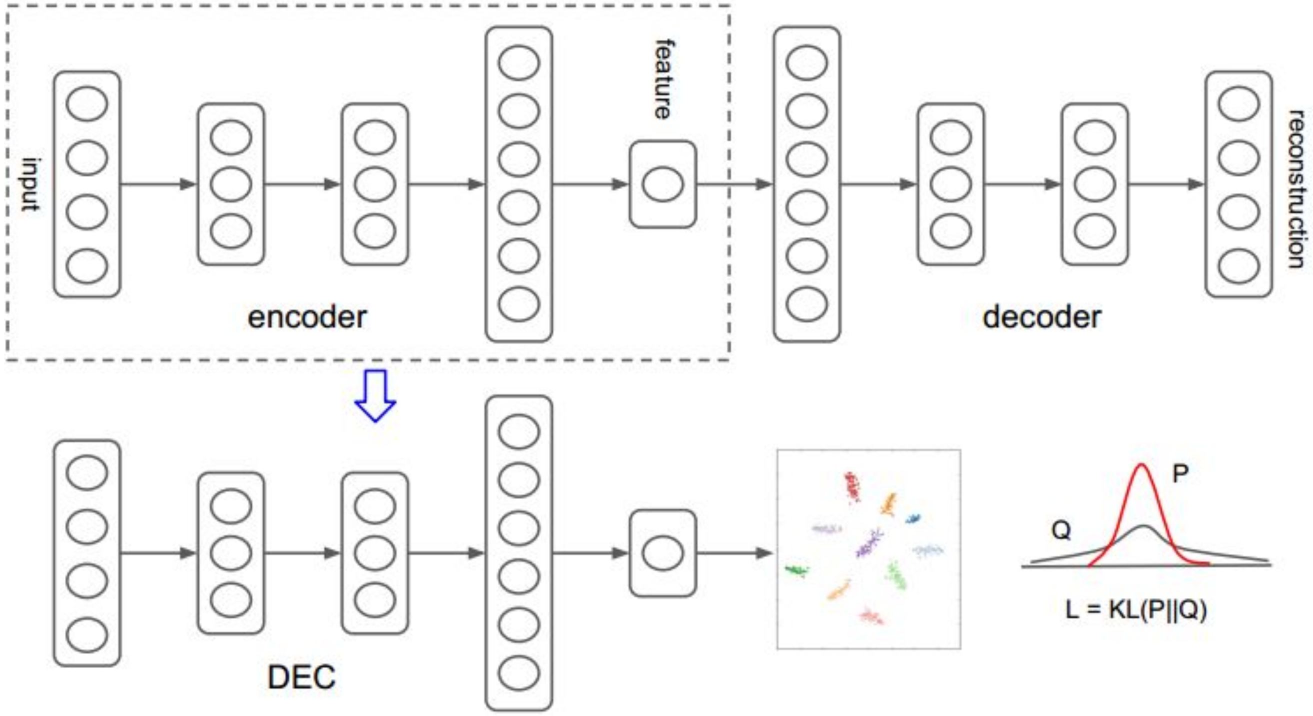
Deep Adversarial Clustering (DAC)

- Very similar to VaDE except Adversarial Autoencoders
  - Loss = Cross entropy with true and fake generated latent codes
- Two loss function: reconstruction and clustering

Deep Embedded Clustering (DEC)

- First trains an AE and then drops the decoder part
- Continues training a *cluster assignment hardening loss* with latent representation
- Other similar methods include DNC and DBC

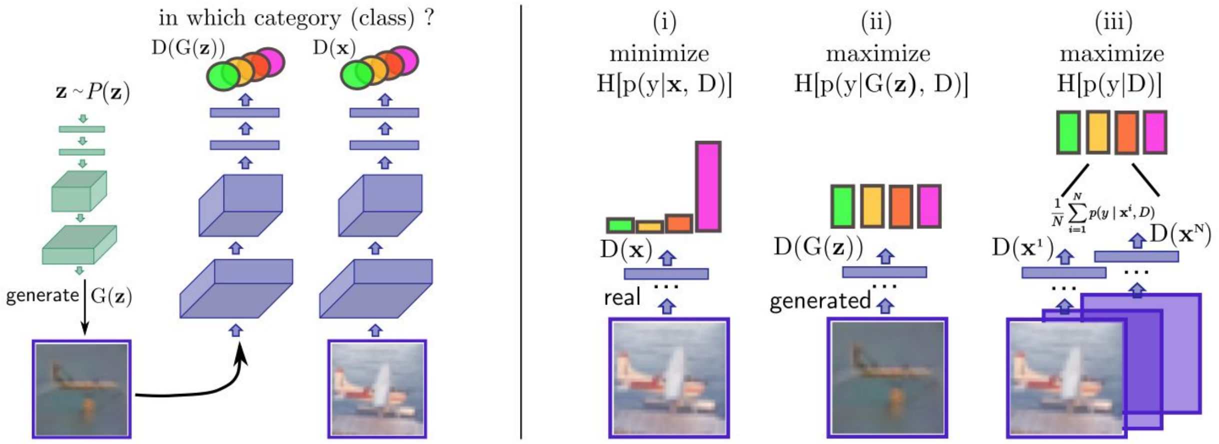
DEC


input
feature
reconstruction
encoder
decoder
DEC
P
Q
L = KL(P||Q)

Other clustering techniques

CatGAN

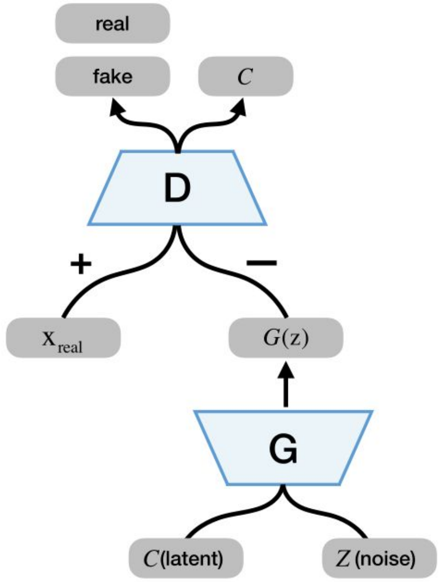
- Extends GAN to multiple classes
- Instead of real/fake, predict the probability of data point belonging to a cluster *with* level of uncertainty



Other clustering techniques

InfoGAN

- Learn disentangled features = clusters
- Input to generator is a latent representation as well as noisy input
- Add a regularizer which enforces high MI between codes and generator distribution

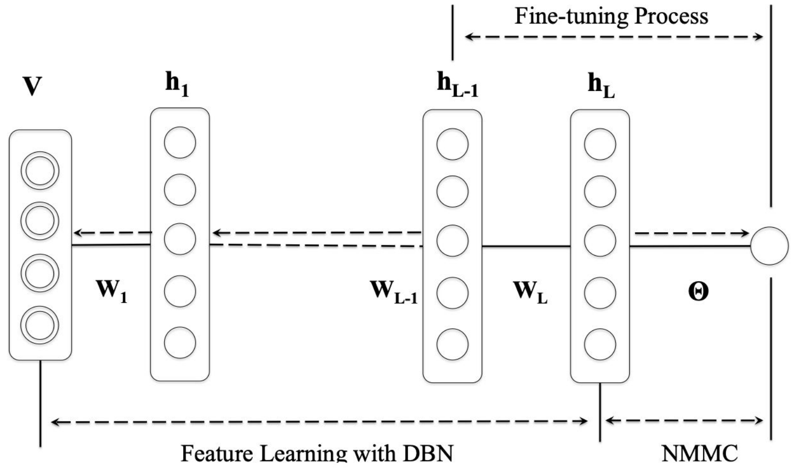


Survey of Clustering With Deep Learning: From the Perspective of Network Architecture

Non-Parametric Clustering

Deep Learning with Nonparametric Clustering (https://arxiv.org/pdf/1501.03084.pdf)

1. Train a Deep Belief Network (DBN)
2. Learn cluster weights using Nonparametric Maximum Margin Clustering (NMMC)
3. Finetune DBN



Predictive Modeling

Time series data such as video

Predict future frames based on what has been learned thus far

PredNet: https://arxiv.org/abs/1605.08104

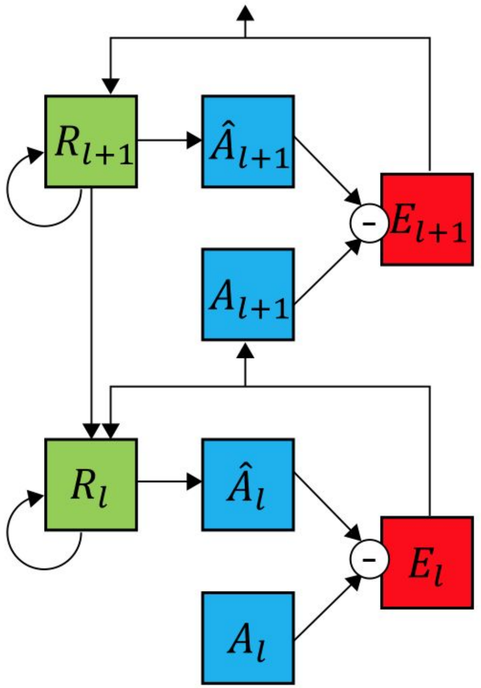
PredNet

Inspired by the concept of “predictive coding” from the neuroscience literature

“Predictive coding posits that the brain is continually making predictions of incoming sensory stimuli”

Benefit of PredNet: “the architecture is general with respect to the kinds of data it models..”

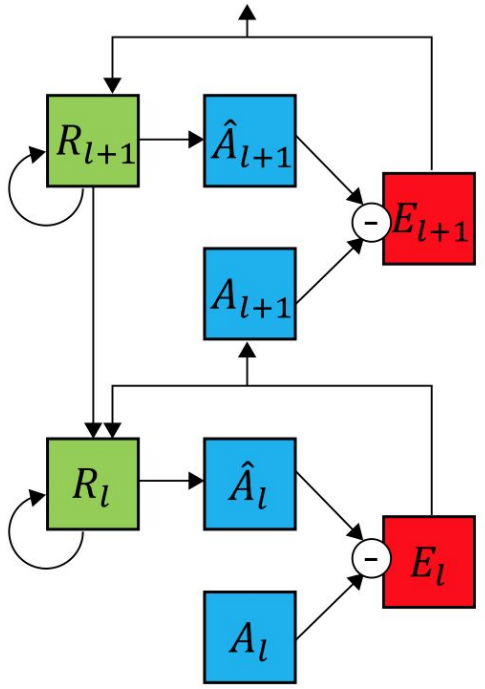
PredNet

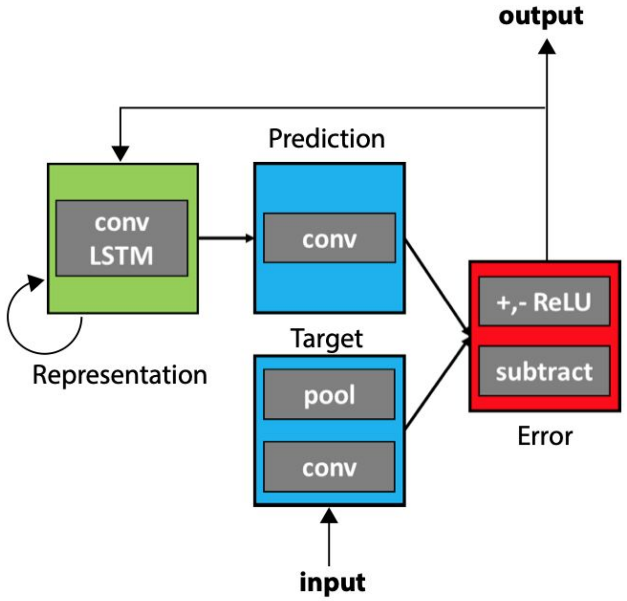


A recurrent neural network which predicts what the next frame will look like

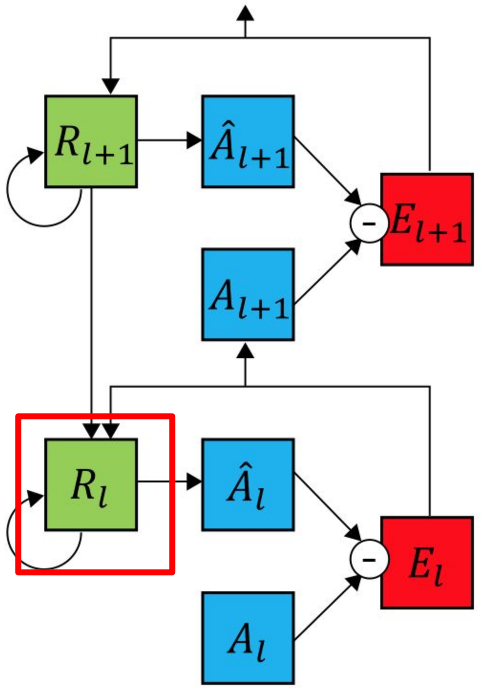
- Generative model

PredNet: Architecture





PredNet



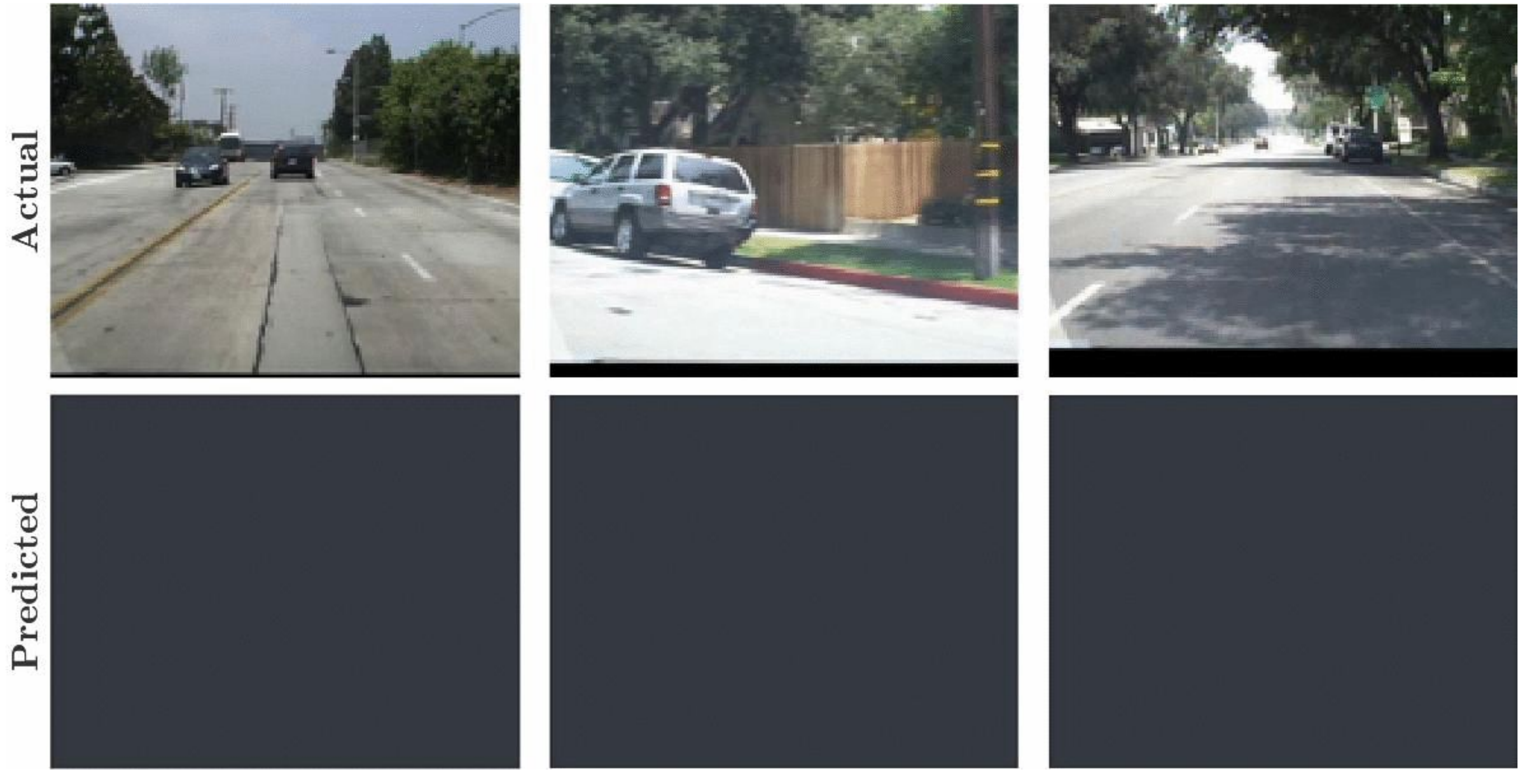
A recurrent neural network which predicts what the next frame will look like

- Generative model (conv LSTM)

Difference between prediction of next frame and current frame

Error becomes the input to the next layer, A_{l+1}

PredNet



Deep Clustering

Pros	Cons
Both benefits of clustering and reconstruction	Complex implementation
Deep disentangled features	Initialization is poor (often need pretrained models)
Visualizable	Need to choose k

Predictive Modeling

Pros	Cons
Unsupervised manner of learning entire scenes	Many parameters to tune
Simple to understand	Memory intensive
	Blurry reconstructions

Next week

Semi supervised learning!