DEEP AUTOENCODING GAUSSIAN MIXTURE MODEL FOR UNSUPERVISED ANOMALY DETECTION

Bo Zong , Qi Song, Martin Renqiang Min , Wei Cheng Cristian Lumezanu , Daeki Cho , Haifeng Chen

EE5180:Introduction to Machine Learning

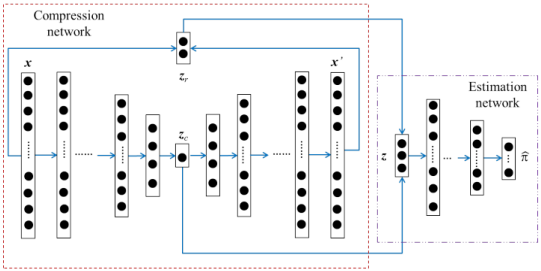
Hemant Kumar EE19B012, EE19B060 Viswa Koduru, M.N.Gayathri EE21S048 ,Koushik Bhat EE21S055

October 18, 2022

Overview

DAGMM

- Deep Autoencoding Gaussian Mixture Model (DAGMM) is our model for unsupervised anomaly detection.
- Our model utilizes a deep autoencoder to generate a low-dimensional representation and reconstruction error for each input data point, which is further fed into a Gaussian Mixture Model (GMM)
- The core of anomaly detection is density estimation: given a lot of input samples, anomalies are those ones residing in low probability density areas.
- Curse of dimensionality: When the dimensionality of input data becomes higher, it is more difficult to perform density estimation in the original feature space, as any input sample could be a rare event with low probability to observe.



The above figure gives an overview on Deep Auto-encoding Gaussian Mixture Model

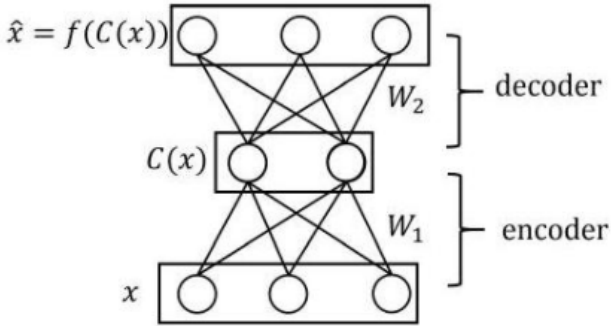
$$z_c = h(\mathbf{x}; \theta_e), \mathbf{x}' = g(\mathbf{z}_c; \theta_d), \mathbf{z}_r = f(\mathbf{x}, \mathbf{x}')$$

$$z = [\mathbf{z}_c, \mathbf{z}_r]$$

where $\mathbf{z}_c$ is the reduced low-dimensional representation, $\mathbf{z}_r$ includes the features derived from the reconstruction error θ_e and θ_d are the parameters of the deep autoencoder $\mathbf{x}'$ is the reconstructed counterpart of $\mathbf{x}$, $h(\cdot)$ denotes the encoding function, $g(\cdot)$ denotes the decoding function, and $f(\cdot)$ denotes the function of calculating

AutoEncoders and Decoders

- The low-dimensional representations provided by the compression network contains two sources of features: (1) the reduced low-dimensional representations learned by a deep auto-encoder; and (2) the features derived from reconstruction error.



matrix W_1 is the collection of weights connecting the bottom and the middle layers and W_2 the middle and the top

AutoEncoders and Decoders

The input x is fed into the bottom layer to produce

$$h = \sigma(Wx + b),$$

The output in the top layer

$$\hat{x} = \sigma\left(W^T h + c\right)$$

$$h = C(x), \hat{x} = f(C(x))$$

- The error is usually the L^2-error given by

$$E = \frac{1}{2}\sum_i \left|\hat{x}^{(i)} - x^{(i)}\right|^2$$

$$= \frac{1}{2}\sum_i \sum_k \left|\hat{x}_k^{(i)} - x_k^{(i)}\right|^2,$$

GMM

- The estimation network performs density estimation under the framework of GMM
- In the training phase with unknown mixture-component distribution , mixture means μ, and mixture co-variance , the estimation network estimates the parameters of GMM and evaluates the likelihood/energy for samples without alternating procedures such as EM.

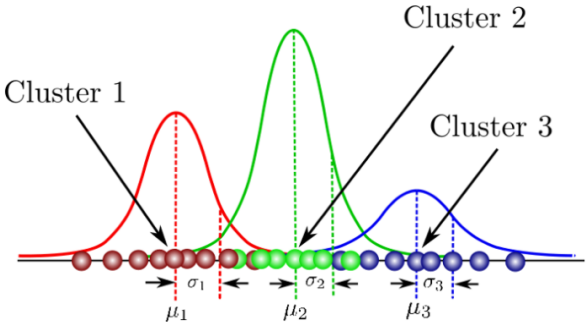
$$\mathbf{p} = MLN(\mathbf{z}; \theta_m), \quad \hat{\gamma} = \text{softmax}(\mathbf{p}) \tag{1}$$

$$\pi_k = \frac{\sum_{n=1}^{N} \gamma(z_{nk})}{N} \tag{2}$$

$$\mu_k = \frac{\sum_{n=1}^{N} \gamma(z_{nk}) x_n}{\sum_{n=1}^{N} \gamma(z_{nk})} \tag{3}$$

$$\sum_k = \frac{\sum_{n=1}^{N} \gamma(z_{nk})(x_n - \mu_k)(x_n - \mu_k)^T}{\sum_{n=1}^{N} \gamma(z_{nk})} \tag{4}$$

GMM1



$$\sum_{k=1}^{K} \pi_k = 1 \tag{5}$$

$$\mathcal{N}(x/\mu, \sum) = \frac{1}{(2\pi)^{D/2} * \sum^{1/2}} \exp((x-\mu)^T (\sum)^{-1} (x-\mu)) \tag{6}$$

GMM2

- The overall probability of observing a point that comes from Gaussian k is actually equivalent to the mixing coefficient for that Gaussian.

$$\pi_k = p\left(z_k = 1\right); \mathbf{z} = \{z_1, \ldots, z_K\} \tag{7}$$

- z occurs independently of others and that they can only take the value of one when k is equal to the cluster the point comes from. Therefore:

$$p\left(\mathbf{x}_n \mid \mathbf{z}\right) = \prod_{k=1}^{K} \mathcal{N}\left(\mathbf{x}_n \mid \mu_k, \mathbf{\Sigma}_k\right)^{z_k} \tag{8}$$

$$p\left(\mathbf{x}_n\right) = \sum_{k=1}^{K} p\left(\mathbf{x}_n \mid \mathbf{z}\right) p(\mathbf{z}) = \sum_{k=1}^{K} \pi_k \mathcal{N}\left(\mathbf{x}_n \mid \mu_k, \Sigma_k\right) \tag{9}$$

- To determine the optimal values for these we need to determine the maximum likelihood of the model. We can find the likelihood as the joint probability of all observations xn.

GMM3

$$p(\mathbf{X}) = \prod_{n=1}^{N} p(\mathbf{x}_n) = \prod_{n=1}^{N} \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n \mid \mu_k, \Sigma_k) \tag{10}$$

$$\ln p(\mathbf{X}) = \sum_{n=1}^{N} \ln \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n \mid \mu_k, \Sigma_k) \tag{11}$$

From Bayes rule, we know that:

$$p(z_k = 1 \mid \mathbf{x}_n) = \frac{p(\mathbf{x}_n \mid z_k = 1)\, p(z_k = 1)}{\sum_{j=1}^{K} p(\mathbf{x}_n \mid z_j = 1)\, p(z_j = 1)} \tag{12}$$

$$p(z_k = 1 \mid \mathbf{x}_n) = \frac{\pi_k \mathcal{N}(\mathbf{x}_n \mid \mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}_n \mid \mu_j, \Sigma_j)} = \gamma(z_{nk}) \tag{13}$$

Expectation Maximization

- The EM algorithm is used for obtaining maximum likelihood estimates of parameters when some of the data is missing
- More generally, the EM algorithm can also be applied when there is latent, i.e. unobserved, data which was never intended to be observed in the first place. In that case, we simply assume that the latent data is missing and proceed to apply the EM algorithm

EM

- E Step :The E-step of the EM algorithm computes the expected value of $l(\theta; \mathcal{X}, \mathcal{Y})$ given the observed data,

$$Q(\theta; \theta_{old}) := \mathrm{E}\left[l(\theta; \mathcal{X}, \mathcal{Y}) \mid \mathcal{X}, \theta_{\text{old}}\right]$$
$$= \int l(\theta; \mathcal{X}, y) p(y \mid \mathcal{X}, \theta_{old})\, dy$$

  where$p(\cdot \mid \mathcal{X}, \theta_{\text{old}})$ is the conditional density of $\mathcal{Y}$ given the observed data, $\mathcal{X}$, and assuming $\theta = \theta_{\text{old}}$

- M Step :The M-step consists of maximizing over θ

$$\theta_{new} := \max_{\theta} Q(\theta; \theta_{old}).$$

  We then set $\theta_{old} = \theta_{new}$

- The two steps are repeated as necessary until the sequence of θ_{new} 's converges.

General Version of the EM Algorithm

$$L(\theta; \mathcal{X}) = p(\mathcal{X} \mid \theta) = \int_y p(\mathcal{X}, y \mid \theta) dy$$

EM algorithm is that it is difficult to optimize $p(\mathcal{X} \mid \theta)$ with respect to θ but that it is much easier to optimize $p(\mathcal{X}, \mathcal{Y} \mid \theta) I(\theta; \mathcal{X}) := \ln p(\mathcal{X} \mid \theta) = \underbrace{\mathcal{L}(q, \theta)}_{\text{"energy"}} + \mathrm{KL}(q \| p\mathcal{Y} \mid \mathcal{X})$

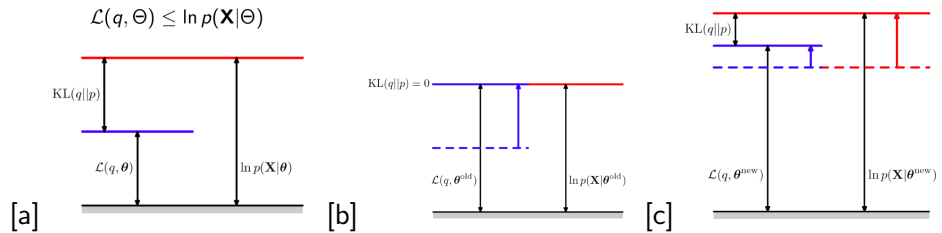
where $\mathcal{L}(q, \theta)$ and $\mathrm{KL}(q \| p, \mathcal{Y} \mid \mathcal{X})$ are the likelihood and $\mathrm{Kullback} - \mathrm{Leibler}(\mathrm{KL})$ divergence which are given by

$$\mathcal{L}(q, \theta) = \int_{\mathcal{Y}} q(\mathcal{Y}) \ln \left(\frac{p(\mathcal{X}, \mathcal{Y} \mid \theta)}{q(\mathcal{Y})} \right)$$

$$\mathrm{KL}\left(q \| p_{\mathcal{Y}|\mathcal{X}}\right) = -\int_{\mathcal{Y}} q(\mathcal{Y}) \ln \left(\frac{p(\mathcal{Y} \mid \mathcal{X}, \theta)}{q(\mathcal{Y})} \right)$$

General Version of the EM Algorithm EM Step

- E-Step: The E-step maximizes the lower bound, $\mathcal{L}(q, \theta_{\text{old}})$, with respect to $q(\cdot)$ while keeping θ_{old} fixed.
- M-Step: In the M-step we keep q(Y) fixed and maximize L(q, θ) over θ to obtain θ_{new}. This will therefore cause the lower bound to increase , which in turn means that the log-likelihood must also increase. Moreover, at this new value new it will no longer be the case that $\mathrm{KL}\left(q \| p_{\mathcal{Y}|\mathcal{X}}\right) = 0$and so by the increase in the log-likelihood will be greater than the increase in the lower bound



Objective Function

Given a dataset of N samples

$$J\left(\theta_e, \theta_d, \theta_m\right) = \frac{1}{N}\sum_{i=1}^{N} L\left(\mathbf{x}_i, \mathbf{x}_i'\right) + \frac{\lambda_1}{N}\sum_{i=1}^{N} E\left(\mathbf{z}_i\right) + \lambda_2 P(\hat{\mathbf{\Sigma}})$$

- $L\left(\mathbf{x}_i, \mathbf{x}_i'\right)$ is the loss function .In practice, L_2-norm usually gives desirable results, as $L\left(\mathbf{x}_i, \mathbf{x}_i'\right) = \|\mathbf{x}_i - \mathbf{x}_i'\|_2^2$.
- $E\left(\mathbf{z}_i\right)$ models the probabilities that we could observe the input samples.
- $P(\hat{\mathbf{\Sigma}}) = \sum_{k=1}^{K}\sum_{j=1}^{d}\frac{1}{\hat{\mathbf{\Sigma}}_{kjj}}$, where d is the number of dimensions in the low-dimensional
- λ_1 and λ_2 are the meta parameters in DAGMM

Next Evaluation Targets

- DAGMM employs end-to-end training
- There are four datasets
  - KDDCUP
  - Thyroid
  - Arrhythmia
  - KDDCUP-Rev
- Take one of the datasets and perform the Fully Connected Layer functions with different input and output neurons along with different activation functions(eg. Sigmoid,Softmax etc)
- Find the Precision, Recall and F_1 score
- Compare with the other existing methods

References

[1]o Zong , Qi Song, Martin Renqiang Min , Wei Cheng Cristian Lumezanu , Daeki Cho , Haifeng Chen Deep Autoencoding Gaussian Mixture Model for Unsupervised Anomaly Detection.

[2] http://www.cse.iitm.ac.in/~vplab/courses/DVP/PDF/gmm.pdf

[3] https://www.cs.toronto.edu/~jlucas/teaching/csc411/lectures/lec15_16_handout.pdf

[4] https://www.cs.cmu.edu/~aarti/Class/10315_Fall19/lecs/Lecture20.pdf

[5] http://www.columbia.edu/~mh2078/MachineLearningORFE/EM_Algorithm.pdf

The End