

# Variational Graph Based Auto Encoder

by Thomas N. Kipf & Max Welling

**DAIICT**

*Term Paper Presentation (201911009)*

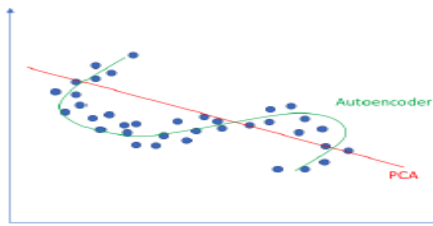
PRML

# Introduction

## What is an Auto Encoder(AE)?

Auto Encoder is kind of neural network which use to represent original data(e.g, text, image) attributes into its lower dimensional form while also preserving all the important features in its latent space such that original data can be retrieved whenever needed. One significant advantage compare to PCA is that it's able to keep all non-linearity present in the data.

Figure: Linear Vs Non-Linear Dimensionality Reduction

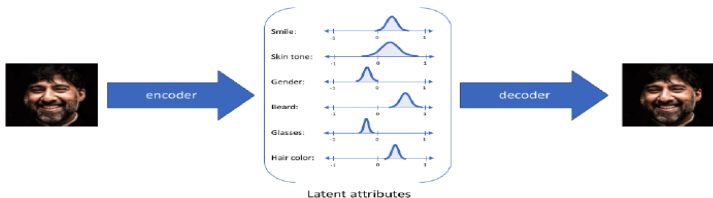


# Introduction

## What is a Variational Auto Encoder(VAE)?

Rather than building an encoder which outputs a single value to describe each latent state attribute, VAE formulate a probability distribution for each latent attribute. Generally the distribution is parameterized by Gaussian distribution having two parameters for each hidden dimension  $\mu$  and  $\log \sigma^2$ . During the decoding one random sample is taken from each distribution.

Figure: Latent Space Representation in VAE



# Introduction

## What is a Variational Graph Auto Encoder(VGAE)?

In the VAE both encoder and decoder network are fully connected neural network where as in VGAE they are Graph convolution Network(GCN)<sup>1</sup> and Inner Product respectively. The Convolution Neural Network can able to extract unique features from the images similarly GCN extracts unique features from the data in graphical format. GCN is motivated from first order approximation of spectral graph convolution.

- Applications

- ▶ Dimensionality Reduction
- ▶ Link Prediction
- ▶ Recommendation system
- ▶ Classification

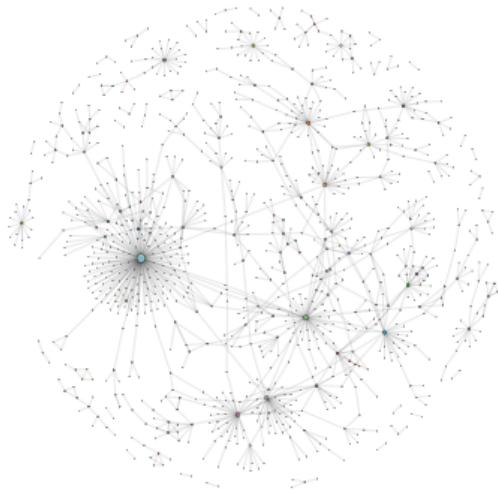
---

<sup>1</sup>Thomas N. Kipf and Max Welling. "Semi-Supervised Classification with Graph Convolutional Networks". In: *International Conference on Learning Representations (ICLR)*. 2017.

# Dataset

- In the paper three different (Cora, Citeseer and Pubmed) citation network dataset is used for the experiment.
- Cora dataset has 2708 Published papers.
- Each paper represented as 1433 unique words/features where each features/word represent as binary value. In the dataset those words/features are avoided which has Document frequency less than 10.
- The citation network consists total 5429 links.
- Each paper classified in one of the following class.
  - ▶ Case Based
  - ▶ Genetic Algorithms
  - ▶ Neural Networks
  - ▶ Probabilistic Methods
  - ▶ Reinforcement Learning
  - ▶ Rule Learning
  - ▶ Theory

# Dataset(Graph)



<http://networkrepository.com/cora.php>

# Implementation(Encoder)

- Given a graph first task is to construct Adjacency Matrix(A) and Feature Matrix(X).
- If we have total N number of Nodes,F number features than first layer GCN defined as following.

$$\bar{X} = GCN(X, A) = ReLU(\tilde{A}XW_0) \quad (1)$$

Where,

$X = N \times F$  matrix

$A = N \times N$  matrix

$W = F \times F$  matrix

$\tilde{A} = D^{-1/2}AD^{-1/2}$  (symmetrically normalized adjacency matrix)

$D_i = \sum_{j=1}^N A_{ij} \quad \forall i \in [1, N]$

# Implementation(Encoder)

- Second layer:

$$\mu = GCN_{\mu}(X, A) = \tilde{A}XW_1 \quad (2)$$

$$\log \sigma^2 = GCN_{\log \sigma^2}(X, A) = \tilde{A}XW_1 \quad (3)$$

- While if sample is directly taken from the generated distribution for decoding purpose than calculation of back-propagation on random sampling is not possible. So to overcome this issue reparameterization trick is used in which one random sample from normal distribution is taken and then it's shifted by  $\mu$  and scaled  $\log \sigma^2$

$$Z = \mu + \log \sigma^2 * \varepsilon; \quad \varepsilon \sim N(0, 1) \quad (4)$$



# Implementation(Decoder)

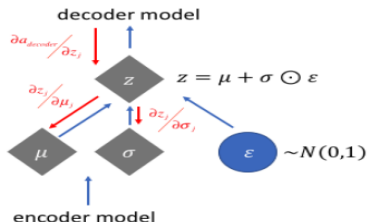


Figure: Reparameterization

- At the Decoder side VGAN uses the Inner Product decoder as described by the following equation

$$\hat{A} = \sigma(ZZ^T) \quad (5)$$

Where,  $\sigma(\bullet)$  = Logistic Sigmoid Function

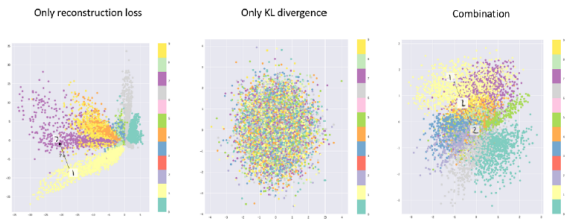
- $\hat{A}$  is new Adjacent Matrix which also incorporate new links which in previous Adjacent Matrix is not present.

# Implementation(Loss Function)

- The VGAN loss function has two parts. The first part is the reconstruction loss(binary cross-entropy) between the input adjacency matrix( $A$ ) and the reconstructed adjacency matrix( $\hat{A}$ ). The second part is the KL-divergence between  $q(Z|X,A)$  and  $p(Z)$ , where  $p(Z) = N(0,1)$ . It measures how closely  $q(Z|X,A)$  matches to  $p(Z)$ .

$$Loss = E_{q(Z|X,A)} \log [p(A | Z)] - KL[q(Z | X, A) || p(Z)] \quad (6)$$

Figure: Concept Behind Two Loss Terms



# Result & Observations

Table: Link prediction task in citation networks.


Implementation (Dataset)	Area Under Curve(AUC)	Average Precision(AP)
Paper(Cora)	91.4	92.6
Paper(Citeseer)	90.8	92.0
Paper(Pubmed)	94.4	94.7
Self(Cora)	86.2	86.3

- There is scope for better probability distribution because a Gaussian prior is potentially a poor choice in combination with an inner product decoder, as the latter tries to push embeddings away from the zero-center.
- Model will fail if large number of nodes presents in the data because many of the core assumptions underlying within design are violated when working in a big data environment.

# Observations(Contd.)

- The reason behind is system require operating on the full graph Laplacian during training—an assumption that is infeasible when the underlying graph has billions of nodes and whose structure is constantly evolving.
- To deal with PySage<sup>2</sup> come with improvement with highly-scalable GCN framework deployed in production at Pinterest.
- More flexible generative models(decoder) can result in better improvement.

---

<sup>2</sup>Rex Ying, Ruining He, and other co-authors. "Graph Convolutional Neural Networks for Web-Scale Recommender Systems". In: *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (2018). 

# Thank You