人工智慧專題

劉瓊如

October 7, 2022

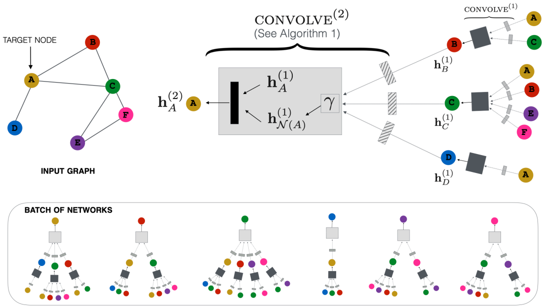
1. Recommendation by graph
   - PinSage
   - Contrastive Learning

2. GCN
   - Graph Theory
   - GCMC

PinSage

論文：Graph Convolutional Neural Networks for Web-Scale Recommender Systems
在 7.5 億的樣本裡，建立有 3 億個 nodes，18 億個邊的圖．PinSage 在 user studies, A/B test 裡，與單純的圖片相似度、語言相似度、混合模型、Pixie 都獲取較高的推薦．
算法解析：利用簡單的 GCN 為每個 pin 與 board 做 embedding，此 embedding 融合圖像與文字特徵來做高品質的 embedding。好的 embedding 不只在推薦或廣告投放有應用，用在 Pinterest 的實際服務上: related pins、搜尋、購物對下游任務非常重要，分類、聚類或重新排名。

PinSage 整體模型



整體模型：使用兩個 Convolve

PinSage Algorithm

Algorithm 1: CONVOLVE

Input : Current embedding $\mathbf{z}_u$ for node u; set of neighbor embeddings $\{\mathbf{z}_v | v \in \mathcal{N}(u)\}$, set of neighbor weights $\boldsymbol{\alpha}$; symmetric vector function $\gamma(\cdot)$

Output : New embedding $\mathbf{z}_u^{\text{NEW}}$ for node u

$\mathbf{n}_u \leftarrow \gamma\left(\{\text{ReLU}\left(\mathbf{Q}\mathbf{h}_v + \mathbf{q}\right) \mid v \in \mathcal{N}(u)\}, \boldsymbol{\alpha}\right);$

$\mathbf{z}_u^{\text{NEW}} \leftarrow \text{ReLU}\left(\mathbf{W} \cdot \text{CONCAT}(\mathbf{z}_u, \mathbf{n}_u) + \mathbf{w}\right);$

$\mathbf{z}_u^{\text{NEW}} \leftarrow \mathbf{z}_u^{\text{NEW}} / \|\mathbf{z}_u^{\text{NEW}}\|_2$

$V = P \cup B$: 所有 nodes

- 將鄰居的鄰居向量經 affine transformation→ReLU→aggregation (importance pooling) → 鄰居向量
- concatenate 自己與鄰居的向量 →affine transformation→ReLU → 自己的向量
- normalized 成自己新的 embedding

PinSage Algorithm

- **Importance-based neighborhood**
  For $u \in P$ ，$\mathcal{N}(u)$：使用 random walk starting from u，累計拜訪次數/random walk 長度 = 拜訪機率，選取 T 個分數最高的機率的 pin 當成 u 的 neighborhood。這樣會產生固定數量的鄰居。
  Algorithm 1 的 γ: weighted mean (weight：使用拜訪機率)，稱為 importance pooling。Pinterest 的 importance-based neighborhood 使用兩層結構（two-fold)，也就是鄰居的鄰居來做自身的特徵抽取。
- **Stacking convolution**
  Algorithm 1 是一層的 convolution
  Algorithm 2 是疊加 convolution 使得可以得到 mini-batch 裡的每個 node 都可以有 k-convolution iteration 的 embedding，層內的 Q,q,W,w 是分享權重，但不同層的權重不同

PinSage Algorithm2

Algorithm 2: MINIBATCH

Input : Set of nodes $\mathcal{M} \subset \mathcal{V}$; depth parameter K; neighborhood function $\mathcal{N} : \mathcal{V} \rightarrow 2^{\mathcal{V}}$

Output: Embeddings $\mathbf{z}_u, \forall u \in \mathcal{M}$

`/* Sampling neighborhoods of minibatch nodes. */`

$\mathcal{S}^{(K)} \leftarrow \mathcal{M}$;

for $k = K, \ldots, 1$ **do**

  $\mathcal{S}^{(k-1)} \leftarrow \mathcal{S}^{(k)}$;

  for $u \in \mathcal{S}^{(k)}$ **do**

    $\mathcal{S}^{(k-1)} \leftarrow \mathcal{S}^{(k-1)} \cup \mathcal{N}(u)$;

  end

end

`/* Generating embeddings */`

$\mathbf{h}_u^{(0)} \leftarrow \mathbf{x}_u, \forall u \in \mathcal{S}^{(0)}$;

for $k = 1, \ldots, K$ **do**

  for $u \in \mathcal{S}^{(k)}$ **do**

    $\mathcal{H} \leftarrow \left\{\mathbf{h}_v^{(k-1)}, \forall v \in \mathcal{N}(u)\right\}$;

    $\mathbf{h}_u^{(k)} \leftarrow \text{CONVOLVE}^{(k)}\left(\mathbf{h}_u^{(k-1)}, \mathcal{H}\right)$

  end

end

for $u \in \mathcal{M}$ **do**

  $\mathbf{z}_u \leftarrow \mathbf{G}_2 \cdot \mathrm{ReLU}\left(\mathbf{G}_1 \mathbf{h}_u^{(K)} + \mathbf{g}\right)$

end

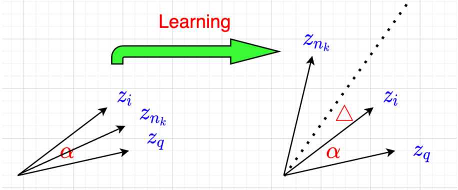
Loss function：Max-Margin ranking loss

利用 triplet loss 的概念，與正樣本拉近，負樣本拉遠到一個範圍外，但 triplet loss 使用的是歐式距離，這裡是內積．不過，作用的向量都是 unit vector，內積等於 cos 值．對於一個 query item z_q 與一個 candidate item z_i

$$J_G(z_q, z_i) = \mathbb{E}_{n_k \sim P_n(q)} \max\{0, z_q \cdot z_{n_k} - z_q \cdot z_i + \triangle\},$$

where $P_n(q)$ represents the distribution of negative examples for item q, $\triangle$ denotes the marginal hyper-parameter.
Note that $z_q \cdot z_{n_k} - z_q \cdot z_i + \triangle > 0$ 就會產生 loss，希望
$z_q \cdot z_{n_k} + \triangle \leq z_q \cdot z_i$, 負樣本在一定的角度之外．
negative sampling：選取 500 個共用在每一個 minibatch

Angle margin



- If $\alpha = \cos^{-1}(\frac{z_i^T z_q}{|z_i||z_q|}) > \beta = \cos^{-1}(\frac{z_{n_k}^T z_q}{|z_{n_k}||z_q|})$ or $\frac{z_i^T z_q}{|z_i||z_q|} < \frac{z_{n_k}^T z_q}{|z_{n_k}||z_q|}$, then produce loss
- 近似：$z_i^T z_q < z_{n_k}^T z_q$ 產生 loss
- $z_{n_k}^T z_q - z_i^T z_q > 0$ 產生 loss
- 在 margin $\triangle$ 範圍外 $z_{n_k}^T z_q + \triangle - z_i^T z_q > 0$ 產生 loss

Multi-GPU training with large minibatch

將一個 mini-batch 切分成等分 portion，每個 GPU 跑一個 portion，做 back propagation 時，同一個參數會在不同 portion 裡，累積 gradient 做一次更新.
假設 A, B node 在 portion 1（batch 1), C,D node 在 portion 2 (batch 2), E, F node 在 portion 3 (batch 3).

$$\hat{w}_{t+1} = w_t - \hat{\eta}\frac{1}{kn}\sum_{j<k}\sum_{x\in B_j}\nabla\ell(x, w_t)$$

在第 t 個 iteration 的參數 w_t，共有 k 個 minibatch.

Curriculum training

抽樣 500 個 negative sample 比起上億個母體，或許不夠比例．在這裡加入 hard examples．選取 hard negative example 的原則：在 Personalized PageRank 分數在對於 q 的排名 2000-5000 名間，主要找不見得相似於 positive sample 但相似於 q．

Query　Positive Example　Random Negative　Hard Negative

除此之外，在第 n 個 epoch training 加入 n-1 個 hard negative items 進 negative sampling set．
使用 1.2 billion pair positive example，每個 batch 500 個 negative example ，6 個 hard example per pin．

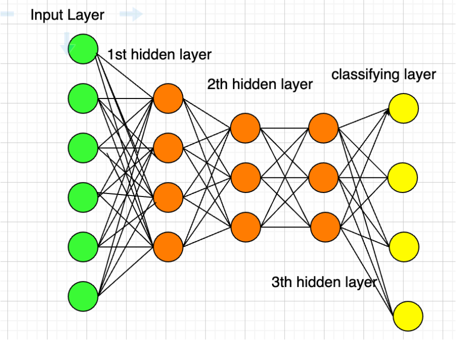
Feature used for learning

$x_q = [x_q^1, x_q^2, x_q^3]$

- $x_q^1 \in \mathbb{R}^{4096}$ visual embedding：VGG output
- $x_q^2 \in \mathbb{R}^{256}$ textual embedding：title+description 使用 w2v 的方式訓練結果
- $x_q^3 = \log |E(q)| \in \mathbb{R}^1$

假設 MLP：$\mathbb{R}^{4393} \to \mathbb{R}^{128}$ 將每個 item map 到低維空間，context item 指的是 item q 放在同一個 board 裡的其他 item concatenate 在一起，若此 item 被存到許多 board，group by context 會 group 出不同組向量，再 reduce by pooling，輸出單一向量，為此 item q 的 embedding
若要做 board embedding，同樣的執行過程，但輸入為上述的 item embedding(node embedding)．

Multilayer Perceptron



- MLP 是深度模型 (deep neural network, DNN) 的一種
- $\mathbf{h}_{\ell+1} = W\mathbf{h}_\ell + b$ or $\mathbf{h}_{\ell+1} = \sigma(W\mathbf{h}_\ell + b)$, where $W \in \mathbb{R}^{d_\ell \times d_{\ell+1}}$ for $\ell = 1, \cdots, L-1$

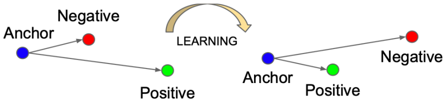
Baseline comparison

Method	Hit-rate	MRR
Visual	17%	0.23
Annotation	14%	0.19
Combined	27%	0.37
max-pooling	39%	0.37
mean-pooling	41%	0.51
mean-pooling-xent	29%	0.35
mean-pooling-hard	46%	0.56
PinSage	**67%**	**0.59**

Visual：單純使用 VGG embedding vector 找歐式距離最近的 itempin)
Annotation：單純使用 title+description vector
Combined：concatenate visual+annotation vector 進 2 層 MLP
hit-rate：推薦 500 個，有被點擊的機率
MRR=Mean Reciprocal Rank $MRR = \frac{1}{n} \sum_{(q,i)\in\mathcal{L}} \frac{1}{\lceil R_{i,q}/100 \rceil}$

Triplet Loss

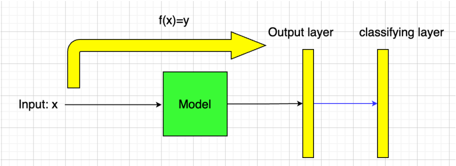
論文：FaceNet: A Unified Embedding for Face Recognition and Clustering



$$L = \sum_{i=1}^{N} \max(0, \|f(x_i^a) - f(x_i^p)\|_2^2 - \|f(x_i^a) - f(x_i^n)\|_2^2 + \alpha)$$

$f(x) \in \mathbb{R}^d$: feature vector, x_i^a: anchor image, x_i^p: positive image, x_i^n: negative image

Feature Engineering



萃取 x 特色的

$$f: \mathcal{X} \to \mathcal{Y}$$
$$x \mapsto f(x)$$

- Language Features 將字變向量
- Image Feature 將照片變相量

Multiple classification

參考論文：Improved deep metric learning with multi-class n-pair loss objective

The $(N+1)$-tuplet loss is to optimize identifying a positive sample from N negative samples. It is

$$L(x_i, x_j) = \mathbf{1}\{y_i = y_j\}\|f(x_i) - f(x_j)\|^2 + \mathbf{1}\{y_i \neq y_j\}\max(0, \alpha - \|f(x_i) - f(x_j)\|^2)$$

Graph Theory

An undirected simple graph is an ordered pair $G = (V, E)$ comprising

- V is a set of vertices (nodes)
- $E \subseteq \{\{x, y\} | x, y \in V, x \neq y\}$ a set of edges (links or lines) (不允許 loop)

Given a simple graph with n vertices, its Laplacian matrix L

$$L = D - A,$$

where D is the degree matrix and A is the adjacency matrix., which is a binary matrix.

Symmetric normalized Laplacian matrix

The symmetric normalized Laplacian matrix is

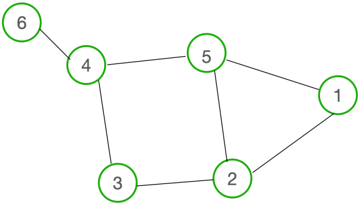
$$\mathcal{L} = L^{sym} = D^{-\frac{1}{2}} L D^{-\frac{1}{2}} = I - D^{-\frac{1}{2}} A D^{-\frac{1}{2}}$$

if G has no isolated points.

Example of finding Laplacian

Example

Find the degree matrix D, Adjacency matrix A, Laplacian matrix L, symmetric normalized Laplacian matrix L^{sym}



Example of finding Laplacian

$$A = \begin{pmatrix} 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}, D = \begin{pmatrix} 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}.$$ The Laplacian matrix is

$$L = D - A = \begin{pmatrix} 2 & -1 & 0 & 0 & -1 & 0 \\ -1 & 3 & -1 & 0 & -1 & 0 \\ 0 & -1 & 2 & -1 & 0 & 0 \\ 0 & 0 & -1 & 3 & -1 & -1 \\ -1 & -1 & 0 & -1 & 3 & 0 \\ 0 & 0 & 0 & -1 & 0 & 1 \end{pmatrix}$$ The normalized Laplacian matrix

$$\mathcal{L} = L^{sym} = I - D^{-\frac{1}{2}} A D^{-\frac{1}{2}} = \begin{pmatrix} 1 & -\frac{1}{\sqrt{6}} & 0 & 0 & -\frac{1}{\sqrt{6}} & 0 \\ -\frac{1}{\sqrt{6}} & 1 & -\frac{1}{\sqrt{6}} & 0 & -\frac{1}{3} & 0 \\ 0 & -\frac{1}{\sqrt{6}} & 1 & -\frac{1}{\sqrt{6}} & 0 & 0 \\ 0 & 0 & -\frac{1}{\sqrt{6}} & 1 & -\frac{1}{3} & -\frac{1}{\sqrt{3}} \\ -\frac{1}{\sqrt{6}} & -\frac{1}{3} & 0 & -\frac{1}{3} & 1 & 0 \\ 0 & 0 & 0 & -\frac{1}{\sqrt{3}} & 0 & 1 \end{pmatrix}$$

Laplace operator

Consider a function $f : V \to \mathbb{R}$ as a vector-valued of function $\mathbf{f}^T = (f(1), \cdots, f(n))$ with $|V| = n$.

- The adjacency matrix can be viewed as an operator

$$\mathbf{g} = A\mathbf{f}, \quad g(i) = A(f)(i) = \sum_j A_{ij} f(j)$$

- It can be viewed as a quadratic form

$$\mathbf{f}^T A \mathbf{f} = \sum_{i,j} f(i) A_{ij} f(j) = \sum_{e_{ij}} f(i) f(j)$$

The incidence matrix of a directed graph

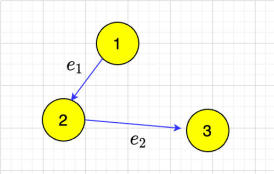
Definition

The incidence matrix of a graph is a $|E| \times |V|$ $(m \times n)$ matrix defined as follows:

$$\nabla := (\nabla)_{ev} = \begin{cases} -1 & \text{if edge e leaves v} \\ 1 & \text{if edge e enters v} \\ 0 & \text{otherwise} \end{cases}$$

The incidence matrix: A discrete differential operator

- The mapping $\nabla : \mathbf{f} \mapsto \nabla \mathbf{f}$ is known as the co-boundary mapping of the graph.
- $\nabla \mathbf{f}(e) = v_{in} - v_{out}$



$$\nabla(f) = \begin{pmatrix} -1 & 1 & 0 \\ 0 & -1 & 1 \end{pmatrix} \begin{pmatrix} f(1) \\ f(2) \\ f(3) \end{pmatrix} = \begin{pmatrix} f(2) - f(1) \\ f(3) - f(2) \end{pmatrix}$$

The Laplacian operator of a directed graph

- $L = \nabla^T \nabla$
- $L(\mathbf{f})(i) = \sum_{j=1}^{n} (\nabla)_{ji}(f(i) - f(j))$
- $L = D - A$ in matrix form

The Laplacian operator of an undirected graph

- The Laplacian as an operator:

$$L(\mathbf{f})(i) = \sum_{j=1}^{n} A_{ij}(f(i) - f(j))$$

- As a quadratic form

$$\mathbf{f}^T L\mathbf{f} = \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} A_{ij}(f(i) - f(j))^2 = (f, L(\mathbf{f}))$$
$$= (\nabla\mathbf{f}, \nabla\mathbf{f}) = \|\nabla f\|_A^2 = (-\triangle f, f)$$

- L is symmetric and positive semi-definite.
- L has n non-negative, real-valued eigenvalues: $0 = \lambda_1 \leq \cdots \leq \lambda_n$

Graph Fourier Transform

Definition

A graph signal $f: V \to \mathbb{R}$ is a function on the vertices of the graph G · Let λ_i and u_i be the i-th eigenvalue and eigenvector of the Laplacian matrix L. The graph Fourier transform $\hat{f}$:

$$\mathcal{GF}[f](\lambda_i) = \hat{f}(\lambda_i) = \langle f, u_i \rangle = \sum_{k=1}^{n} u_i(k) f(k),$$

where $\{u_i\}_{i=1}^{n}$ forms an orthonormal basis of $\mathbb{R}^n$. Thus

$$\mathcal{GF}[\mathbf{f}] = \hat{\mathbf{f}} = U^T \mathbf{f}$$

Recall: $\mathbf{f} = \sum_{i=1}^{n} \langle \mathbf{f}, u_i \rangle u_i = \sum_{i=1}^{n} \hat{\mathbf{f}}(i) u_i$

The inverse graph Fourier transform

Definition

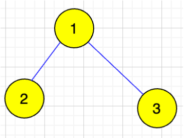
The inverse graph Fourier transform is

$$\mathcal{IGF}[\hat{f}](i) = f(i) = \sum_{k=1}^{n-1} \hat{f}(\lambda_k) u_k(i).$$

In matrix form

$$\mathcal{IGF}[\hat{\mathbf{f}}] = \mathbf{f} = U\hat{\mathbf{f}}$$

Example



$L = D - A = \begin{pmatrix} 2 & -1 & -1 \\ -1 & 1 & 0 \\ -1 & 0 & 1 \end{pmatrix}$, The characteristic polynomial of L is $p(\lambda) = \det(L - \lambda I) = -\lambda(\lambda - 1)(\lambda - 3)$. The unit corresponding eigenvector are $u_{\lambda_1} = \begin{pmatrix} \frac{1}{\sqrt{3}} \\ \frac{1}{\sqrt{3}} \\ \frac{1}{\sqrt{3}} \end{pmatrix}$, $u_{\lambda_2} = \begin{pmatrix} 0 \\ \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix}$, $u_{\lambda_3} = \begin{pmatrix} \frac{-2}{\sqrt{6}} \\ \frac{1}{\sqrt{6}} \\ \frac{1}{\sqrt{6}} \end{pmatrix}$.

$U = \begin{pmatrix} u_{\lambda_1} & u_{\lambda_2} & u_{\lambda_3} \end{pmatrix}$. Then $L = U \cdot \mathrm{diag}(\lambda_1, \lambda_2, \lambda_3) U^T$

Example of Graph Fourier Transform

$U = \begin{pmatrix} \frac{1}{\sqrt{3}} & 0 & \frac{-2}{\sqrt{6}} \\ \frac{1}{\sqrt{3}} & -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{6}} \\ \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{6}} \end{pmatrix}$. If $\mathbf{f} = \begin{pmatrix} f(1) \\ f(2) \\ f(3) \end{pmatrix} = \begin{pmatrix} 1 \\ -1 \\ 1 \end{pmatrix}$. Then the graph Fourier transform of $\mathbf{f}$ is

$$\mathcal{GF}[\mathbf{f}] = \hat{\mathbf{f}} = U^T\mathbf{f} = \begin{pmatrix} \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} \\ 0 & -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{-2}{\sqrt{6}} & \frac{1}{\sqrt{6}} & \frac{1}{\sqrt{6}} \end{pmatrix} \begin{pmatrix} 1 \\ -1 \\ 1 \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{3}} \\ \frac{2}{\sqrt{2}} \\ \frac{-2}{\sqrt{6}} \end{pmatrix}$$

The inverse graph Fourier transform of $\hat{\mathbf{f}}$ is

$$\mathcal{IGF}[\hat{\mathbf{f}}] = \mathbf{f} = U\hat{\mathbf{f}} = \begin{pmatrix} \frac{1}{\sqrt{3}} & 0 & \frac{-2}{\sqrt{6}} \\ \frac{1}{\sqrt{3}} & -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{6}} \\ \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{6}} \end{pmatrix} \begin{pmatrix} \frac{1}{\sqrt{3}} \\ \frac{2}{\sqrt{2}} \\ \frac{-2}{\sqrt{6}} \end{pmatrix} = \begin{pmatrix} 1 \\ -1 \\ 1 \end{pmatrix}$$

Graph convolution

Then $\mathcal{GF}[\mathbf{f}] = \hat{\mathbf{f}} = U^T\mathbf{f}$. In particular, for graph convolution

$$(\mathbf{f} * \mathbf{g}) = \mathcal{IGF}[\mathcal{GF}[(\mathbf{f} * \mathbf{g})]]$$

That is,

$$(\mathbf{f} * \mathbf{g})(i) = \sum_{k=1}^{n} \hat{\mathbf{f}}(\lambda_k)\hat{\mathbf{g}}(\lambda_k)u_k(i)$$

$$(\mathbf{f} * \mathbf{g})_G = U \begin{pmatrix} \hat{\mathbf{g}}(\lambda_1) & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \hat{\mathbf{g}}(\lambda_n) \end{pmatrix} U^T f$$

GCN

參考論文：SEMI-SUPERVISED CLASSIFICATION WITH GRAPH CONVOLUTIONAL NETWORKS 2017
假設任務是要分類 graph 裡的 node，但只有部分答案（some labels are available）。這個監督問題可以被圖的結構弭平

$$\mathcal{L} = \mathcal{L}_0 + \lambda \cdot \mathcal{L}_{reg},$$

where $\mathcal{L}_0$ 是監督式有標籤的 loss、

$$\mathcal{L}_{reg} = \frac{1}{2}\sum_{i,j} A_{ij}\|f(X_i) - f(X_j)\|^2 = f(X)^T \triangle f(X) = \frac{1}{2}\|\nabla f\|_A^2$$

where $\triangle = D - A$ the unnormailized graph Laplacian of an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with N nodes $v_i \in \mathcal{V}$, edges $(v_i, v_j) \in \mathcal{E}$, an adjacency matrix $A \in \mathcal{R}^{N\times N}$(binary or weighted) and a degree matrix $D_{ii} = \sum_{j=1}^{N} A_{ij}$。此 loss function 是基於一個聯通的圖會分享同一個 label，這個假設有可能會限制 model 的容納量

Fast Approximate Convolution on Graph

建立一個以圖為基礎的多層圖卷機（multi-layer Graph Convolutional Network: GCN)

$$H^{(\ell+1)} = \sigma\left(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}H^{(\ell)}W^{(\ell)}\right)$$

where $\tilde{A} = A + I_N$: the adjacency matrix of the undirected graph $\mathcal{G}$ with added self-connection,
$\tilde{D}_{ii} = \sum_{j=1}^{N} \tilde{A}_{ij}$
$W^{(\ell)} \in \mathbb{R}^{D\times D}$：第 ℓ 層的權重
σ: activation function 可以是 ReLU 或其他
$H^{(\ell)} \in \mathbb{R}^{N\times D}$：output of the ℓ-th hidden layer
$H^{(0)} = X$

Spectral Graph Convolution

Given a signal $x \in \mathbb{R}^N$，定義 spectral graph convolution with a filter: $g_\theta = \mathrm{diag}(\theta)$, where $\theta \in \mathbb{R}^N$

$$g_\theta \star x = U g_\theta U^T x = U \begin{pmatrix} \theta_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \theta_N \end{pmatrix} U^T x$$

where U is the matrix of eigenvectors of the symmetric normalized graph Laplacian$=\mathcal{L}$

$$\begin{aligned} \mathcal{L} &= D^{-\frac{1}{2}}(\triangle)D^{-\frac{1}{2}} = D^{-\frac{1}{2}}(D-A)D^{-\frac{1}{2}} \\ &= I_N - D^{-\frac{1}{2}}AD^{-\frac{1}{2}} = U\Lambda U^T, \end{aligned}$$

Λ 是 $\mathcal{L}$ 的對角化矩陣（特徵值對角矩陣，U 的 column vector $\mathcal{L}$ 的特徵向量

Approximation of Chebyshev polynomials

$g_\theta(\Lambda)$ is a function of eigenvalues of $\mathcal{L}$・但計算成本太大，這可被 truncated expansion in terms of Chebyshev polynomial $T_k(x)$ up to K-th order

$$g_{\theta'}(\Lambda) \approx \sum_{k=0}^{K} \theta'_k T_k(\tilde{\Lambda})$$

with rescale $\tilde{\Lambda} = \frac{2}{\lambda_{max}}\Lambda - I_N$, λ_{max} denote the largest eigenvalue of $\mathcal{L}$. (因為 T_k 定義範圍在 [-1,1] 之間，所以需要 rescale 輸入) $\theta' \in \mathbb{R}^K$is a vector of Chebyshev coefficients $T_k(x) = 2xT_{k-1}(x) - T_{k-1}(x)$ with $T_0(x) = 1$ and $T_1(x) = x$

重新定義 a convolution of a signal x with filter $g_{\theta'}$

$$g_{\theta'} \star x \approx \sum_{k=0}^{K} \theta'_k T_k(\tilde{L})x,$$

where $\tilde{L} = \frac{2}{\lambda_{max}}L - I_N$・因為 $(U\Lambda U^T)^k = U\Lambda^k U^T$

Layer-Wise Linear Model

$K = 1$, $\lambda_{max} \approx 2$

$$g_{\theta'} \star x \approx \theta'_0 x + \theta'_1 (L - I_N) x = \theta'_0 - \theta'_1 D^{-\frac{1}{2}} A D^{-\frac{1}{2}} x$$

減低參數 $\theta = \theta'_0 = -\theta'_1$，上式變成

$$g_\theta \star x \approx \theta (I_N + D^{-\frac{1}{2}} A D^{-\frac{1}{2}}) x$$

renormalization trick: $I_N + D^{-\frac{1}{2}} A D^{-\frac{1}{2}} \rightarrow \tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}}$with $\tilde{A} = A + I_N$and $\tilde{D}_{ii} = \sum_j \tilde{A}_{ij}$
若推廣這個定義到 a signal $X \in \mathbb{R}^{N \times C}$with C input channels and F filters ($\Theta \in \mathbb{R}^{C \times F}$ matrix of filer parameters)

$$Z = \tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}} X \Theta \in \mathbb{R}^{N \times F}$$

is the convolved signal

Semi-Supervised Node classification

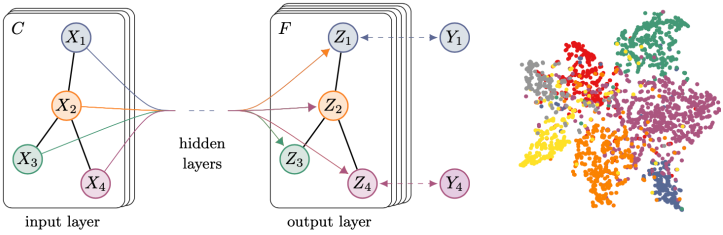
考慮 two-layer GCN for semi-supervised node classification: let $\hat{A} = \tilde{D}^{-\frac{1}{2}}\tilde{A}D^{-\frac{1}{2}}$

$$Z = f(X, A) = \mathrm{softmax}(\hat{A}(\mathrm{ReLU}(\hat{A}XW^{(0)}))W^{(1)})$$

監督式部分的 loss：cross-entropy

$$\mathcal{L}_0 = -\sum_{\ell \in Y_L}\sum_{f=1}^{F} Y_{\ell f}\log Z_{\ell f}$$

where Y_L is the set of node indices that have labels．



(a) Graph Convolutional Network (b) Hidden layer activations

GCMC

論文：Graph Convolutional Matrix Completion 2017

基於 user-item 的 Bipartite graph，利用 Graph Auto-Encoder 來預測使用者的評分

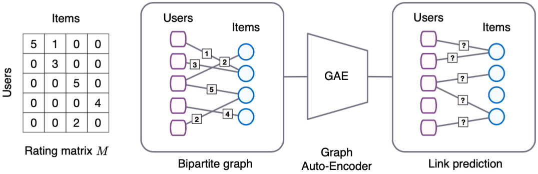
Let a rating matrix $M \in \mathbb{N}^{N_u \times N_v}$, where N_u is the number is users and N_v is the number of items

常用方式：協同過濾

$$M_{ij} = \begin{cases} r & \text{if user } i \text{ rated item } j \\ 0 & \text{otherwise} \end{cases}$$

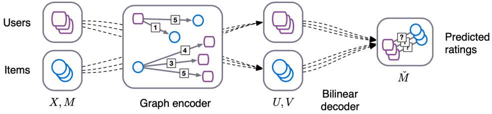
for $r \in \{1, \cdots, R\}$

整體模型



- Bipartite graph $G = (\mathcal{W}, \mathcal{E}, \mathcal{R})$, where $\mathcal{U} = \{u_i\}_{i=1}^{N_u}$: users nodes, $\mathcal{V} = \{v_j\}_{j=1}^{N_v}$: item nodes, $\mathcal{W} = \mathcal{U} \cap \mathcal{V}$, edge (u_i, r, v_j) for $r \in \{1, \cdots, R\} = \mathcal{R}$
- Graph encoders: $Z = f(X, A)$, where $X \in \mathbb{R}^{N \times D}$ for $N = N_u + N_v$ 所有 nodes, A: adjacency matrix
  output: $Z = [z_1^T, \cdots, z_N^T]^T \in \mathbb{R}^{N \times E}$
- decoder $g(Z) = \check{A}$, $g(z_i, z_j) = \check{A}_{ij}$

Graph auto-encoders



- Encoder $f(X, M_1, \cdots, M_R) = [U, V]$, where $M_r \in \{0, 1\}^{N_u \times N_v}$, for $r \in \mathcal{R}$, $U \in \mathbb{R}^{N_u \times E}$, $V \in \mathbb{R}^{N_v \times E}$
- Decoder $\check{M} = g(U, V)$ rating matrix of shape $N_u \times N_v$

Graph convolution encoder for users

- The hidden vector for each user i

$$h_i = \sigma[\sum_{j \in \mathcal{N}_{i,1}} \mu_{j \to i,1}, \cdots, \sum_{j \in \mathcal{N}_{i,R}} \mu_{j \to i,R}],$$

  where $\mu_{j\to i,r} = \frac{1}{c_{ij}} W_r x_j$：edge-type rating result from item j to user i，$c_{ij}$is a normalization constant $c_{ij} = |\mathcal{N}_i|$ or $c_{ij} = \sqrt{|\mathcal{N}_i||\mathcal{N}_j|}$ with $\mathcal{N}_i$ denoting the set of neighbors of node i, and $\sigma(x) = \mathrm{ReLU}(x) = \max(0, x)$.
- $\sum_{j \in \mathcal{N}_{i,r}} \mu_{j \to i,r} = \sum_{j=1}^{N_v} M_r \mu_{j \to i,r}$
- The output of GAE for the user vector $u_i = \sigma(W h_i)$

Graph convolution encoder for users

Similarly, item vector 也可透過同樣的方式得到：

- $$\mu_{i \to j, r} = \frac{1}{c_{ij}} W_r x_i$$
- $$h_j = \sigma\left[\sum_{i \in N_{j,1}} \mu_{i \to j, 1}, \cdots, \sum_{i \in N_{j,R}} \mu_{i \to j, R}\right]$$
- The output of GAE for the item vector $v_j = \sigma(Wh_j)$

Decoder

再來經過一個 bilinear decoder $\check{M} = g(U, V)$：

- The predicted rating

$$\check{M}ij = g(u_i, v_j) = \mathbb{E}_{p(\check{M}ij=r)}[r] = \sum_{r \in R} rp(\check{M}_{ij} = r)$$

- The decoder produces a probability distribution of the rating between user i and item j with M_{ij} ,

$$p(\check{M}ij = r) = \frac{e^{u_i^T Q_r v_j}}{\sum s \in Re^{u_i^T Q_r v_j}},$$

$Q_r \in \mathbb{R}^{E \times E}$ 是學習參數。

Training

Loss function:

$$L = - \sum_{i,j \in \Omega : \Omega_{ij}=1} \sum_{r=1}^{R} \mathbf{1}[r = M_{ij}] \log p(\check{M}_{ij} = r),$$

where $\Omega \in \{0,1\}^{N_u \times N_v}$ serves as a mask for unobserved ratings, such that ones occur for elements corresponding to observed ratings in M, and zeros for unobserved ratings.

Matrix representation

- $$Z = \begin{bmatrix} U \\ V \end{bmatrix} = f(X, M_1, \cdots, M_R) = \sigma\left(\begin{bmatrix} H_u \\ H_v \end{bmatrix} W^T\right),$$
  where $X = \begin{bmatrix} X_u \\ X_v \end{bmatrix}$, $\begin{bmatrix} H_u \\ H_v \end{bmatrix} = \sigma\left(\sum_{r=1}^{R} D^{-1}\mathcal{M}_r X W_r^T\right)$,
  $\mathcal{M}_r = \begin{pmatrix} 0 & M_r \\ M_r^T & 0 \end{pmatrix}$, $U \in \mathbb{R}^{N_u \times E}$, $V \in \mathbb{R}^{N_v \times E}$.
- Feature representation
  為了區分 user node 與 item node，可加入 user side information 與 item side information：
  user: $u_i = \sigma(Wh_i + W_2^f f_i)$, $f_i = \sigma(W_1^f x_i^f + b)$,
  item: $v_j = \sigma(Wh_j + W_2^f f_j)$, $f_j = \sigma(W_1^f x_j^f + b)$
  在 GCN 輸入時的 $X = [X_u, X_v]$ 使用 one-hot，所以 X 是一個 identity matrix。

Training

- Mini-batch
- 每次只採固定數量的 user-item 對
- Node Dropout
  為了使模型泛化到未觀測到的評分，在訓練中使用 dropout，以一定機率 p_{dropout} 刪除特定節點所有的傳出值．
- Weight sharing