Keywording

- Inverse Page Rank
- Investor Rank

Power Problem

$$r_j^{(t+1)} = \sum \frac{r_i^{(t)}}{d_i}$$

- does it converge
- " " to what we want
- does it have reasonable result

- spider_trap doesn't matter
- Dead_end should be ok

↳ use tele_port (teleport, randomly jump somewhere else, for connection

↳ Maybe need to consider spider_trap

↳ preprocess to remove dead_ends in page rank

random jumper: all nodes have some importance

Personalized pagerank

↳ teleport to subset of nodes (not any node)
= Random walk with restart.

→ consider both: shortest path
, common neighbour

For similarity.

- Multiple connection
- Multi path
- direct and indirect connection
- degree of nodes

$S = [0.1, 0.1, 0.1, 0.1]$. Page Rank.

$S = [0.2, 0, 0.8, 0]$. Topic specific.

$S_2 = [0, 0, 1, 0]$. Restart

does it work with weighted graph.

Matrix Factorization

$z^T z = A$

$\min \| z^T z - A \|_2$ L2 used soft max

↳ solve this instead of gradient desc

limitation

- can't predict for new commer
- deepwalk, node2vec can't predict structral similarity
  - ↳ can't capture local similarity
- Features can't be utlize between node, edge and graph features

Msg passing - Node classification.

semi-suppervised : half, partialy label

↳ Node share label are connected.

- Relational Classification → node label
- Iterative → label + node attribute
- Belief propogation → LOOPY BP Algorithm

Social Network

Correlation: node close belong to have same label

- Homophily: tendency of individual to bond
- Influence: social interaction affect individual character

Markov assumption: only node neighbors

↳ consider weights

GNN

- stop at convergence of validation set
- weight matrix (Jacobian Matrix)

$F(x) = W_1 x$

Back Propogation: chain rule to propogate gradient of intermediate steps and finally find the gradient

GNN: Learn how to propogate information across the graph to compute node features.

↳ every node comes with its own computational network

Neigborhood Aggrigation

Design Space

1 Layer
- Aggregation (1)
- Msg (2) → add nonlinearity
- layer connection (3)

(4) raw input graph ? computational graph
Graph Manipulation

(5) learning objective

① MSG computation $m_u^{(l)} = MSG^{(l)}(h_u^{(l-1)})$ $m_v^{(l)}$ B

② Aggregation $h_v^{(l)} = AGG^{(l)}(\{m_u^{(l)}, u \in N(v)\})$ $N(v)$

Sum(·), Max(·), Mean(·)

• concatenation, sumation

$$h_v^{(l)} = \text{CONCAT}\left(\text{AGG}\left(\{m_u^{(l)}, u \in N_{w_i}\}\right), m_v^{(l)}\right)$$

- GCN — $\frac{1}{|N(v)|}$ implicit
- GraphSAGE
- GAT attention — can also learn
  α_{uv} → How important is a msg

(3) Layer Connection

- Stacking many layers → over smoothing
- ~~layer~~ depth in GNN is not same as NN
- Receptive Field - set of the node of interest
- if you collect so much, you include all

Shallow GNN More expresive

↳ add more depth to a single layer

↳ add layer without msg sending/pre-post processing

↳ skip connection (mixture of models)

Framework

Raw input Graph ≠ computational graph
- graph Feature augmentation
- " Structur

- too sparse/Dense/Larg
- Lack Feature — const Feature → any FTs in lec 2
  ↳ one-hot node

- sparse ① add virtual edge
  ↳ biparttite graph — $A + A^2$ (2 hop adj)

② add virtual node
make msg sending eff, for that one nodes

- Dense
sample random

(4) Train. Graph — Learning Objectives

① Different prediction head: y_v node
y_{uv} edge - dotprod, contacte
y_G graph [Global pooling
hirerarchical

can train paralel to ← Diffpool
- how to group (community detection)
- how to make supper node

② Predictions
- suppervised
- unsuppervised: link prediction (self-sup)
  use graph structure

③ Loss Function

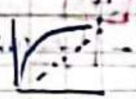
	Loss Function
- Classification, discrete	cross Entropy (CE)
- Regression, continues	MSE, L^2
- node ordering	triplet loss, Maximum Margin loss

④ Evaluation

RSME: root mean square error
MAE: mean absolute error
(sklearn)

accuracy
Precision
Recall
F1. score
ROC curve (recieving operating characteristic curve)

TPR



FPR

TPR = recall = $\frac{TP}{TP+FN}$

FPR = $\frac{FP}{FP+TN}$

- ROC AUC
  - area under ROC curve
  - higher than 0.5 is good

⑤ How to split the data

- Fixed — test, valid, train
- Random

solution 1 — split only node lable / Transductive
solution 2 — remove edges to split / Inductive

→ Graph classification

Link prediction

alt1 inductive

① devide edge
- suppervision → Train / V / Te
- Msg passing → Tr / V / Test

alt2 Transductive

- train: train msg —predict→ train supp
- valid: train msg + train supp → valid edges
- test: train msg + valid e + train sup + → test edge

- DeepSNAP — PyG.org → pytorch geometric
- GraphGym — twitter
  - DGL → small comunity

Front

- GCNs, GraphSAG not powerful / not injective in their aggregation (Max, mean pooling) function

Injective Function — sum over multiset

- MLP (multi layer perceptor)
  - 100 - 500 hidden dimension

base info

Most Expressive GNN: GIN — Graph isomorphis Network

$MLP_{\Phi} \sum_{x \in S} MLP_{f}(x)$

→ same as WL kernel Method

$c^{(k+1)}(v) = GINConv(c^{(k)}_{(v)}, \{c^{(k)}_{(u)}\}_{u \in N})$

differintaible Hash Function

- Have differencive, but equally expressive

Injective Multi Set Function

Hybrid Model : Langugе Model + KG

- StanFord 2020 kG seminar
- PiNSAGE → large scale
  - GDB → computation Graph batche → too slow
    - ↳ on going work for mini batch sampler
- time series data, GNN
  - ↳ Dynamic graph

KNowledge graph

- Hetrogeneous graphs: multi edge/node types
  - ↳ $G = (V, E, T, R)$ ↳ Relation Type
    - ↳ node Type
- RGCN: relational GNN → too many parameter over fitting issue
- ↳ Regularize $W_r^{(l)}$
  - block diag matrice
  - Basis/Dictionary learning

KG capktion: What is the place of birth

Completetion Alg: transE, transR

DistMult, complEx

		sym	Asym	Inv	compos	1-to-N
transE	$-\|\|h+r-t\|\|$	✗	✓	✓	✓	✗
transR	$-\|\|W_r h + r - W_r t\|\|$	✓	✓	✓	✗	✓
DistMult	$<h, r, t>$	✓	✗	✗	✗	✓
complEX	$Re(<h, r, \bar{t}>)$	✓	✓	✓	✗	✓

Kernel

- low dim → high dim to simplify task
- smothniess → over Fit / under Fit

Hilbert Space kernel: inner product (dot product) dot product of Features

- ↳ sum, product of kernel is kernel → polynomyals → kernel
- ↳ Is it a kernel or not? → kernel is definite positive

ML → Make Function of Features to predict

GNN WorkShop

ROLAND

- learn combing feature
- learn structure
- forcast central bank transaction

- Graph ML Tutorials
  - PYG - NIvida
    - ↳ accelerated GNN for heterogenous graph

OGB.stanford.edu

- store fact in head — station — tail

SMORE: scalable Framework for Multihope

80M node, 300M edges

PyG_lib = [redacted] + torch-scatter + torch-sparse

.acceler

MaRDi 2022.26.

Formal varification - HOL light

NFDI > MARDI | European Open Science Cloud

↳ Fair Research Data

BMWF, DFG money

↳ 5 years

Findable
Accessable
Interpretable
Re-useable

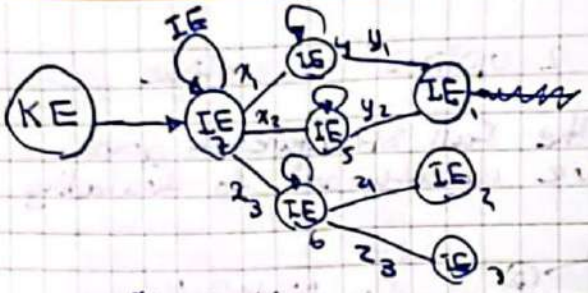
KG For Simulation
T_{14}

Tim-Berners-Lee - Data Quality

*** CRC-1456 - Mathematic of experiments
C. Lehrenfeld

- Lab book / snip, sci. photos / QR / SNI22 — Binder - Jupyter
- LiveDocs / → interaction
  ↳ Math Preprint ↳ ipywidget / voila

- MLFlow
- weights biases
- correlation Matrix
- optimal



Find x_1 x_2 x_3 y_1 y_2 z_1 z_2 z_3 $x_3 = z_1 + z_2 + z_3$

↳ It can work as a recommendation for a load path

solve PageRank reverse

$r = Mr$

$r_2 = Mr$

= page rank

→ use (DFS (vs BFS)) to compare similarity

→ what is the load path

↳ where is the surfer

$p(t+1) = M\,p(t) = p(t)$

stablize over graph

$p(t)$ stationary distribution of the random walk

Problem: IF IE is on edge what is the pagerank

$\lambda c = \underline{A}c$

↳ adjacency Matrix

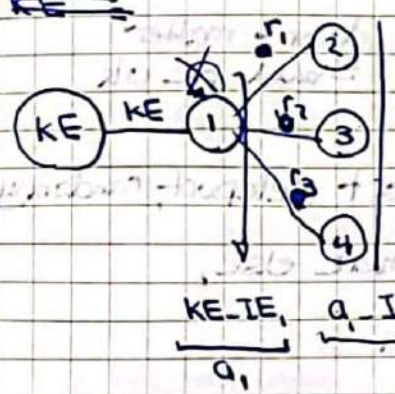
$1 \cdot r = M \cdot r$ → flow equation

reverse flow summation

$x_3 = IE_2 + IE_3 + z_1 + z_3$

$x_2 + x_1 = IE_1 + IE_4 + IE_5 + y_1 + y_2$

$x_1 + x_2 + x_3 + IE_7 = KE$



$\underbrace{KE\_IE_1}_{a_1}$ $\underbrace{a_1\_IE_2\_IE_3\_IE_4}_{a_2}$

$a_1(r_4 + r_2 + r_3) = a_2 + IE_2 + IE_3 + IE$

What if we ignore the self loop and have the IE as the Page Rank

↳ $\frac{IE_i}{KE}$