Deep Learning on Graphs

- Introduction of Graph, Graph Machine Learning, Graph Neural Network, & GraphStorm

Jian Zhang

AWS AI Research & Education

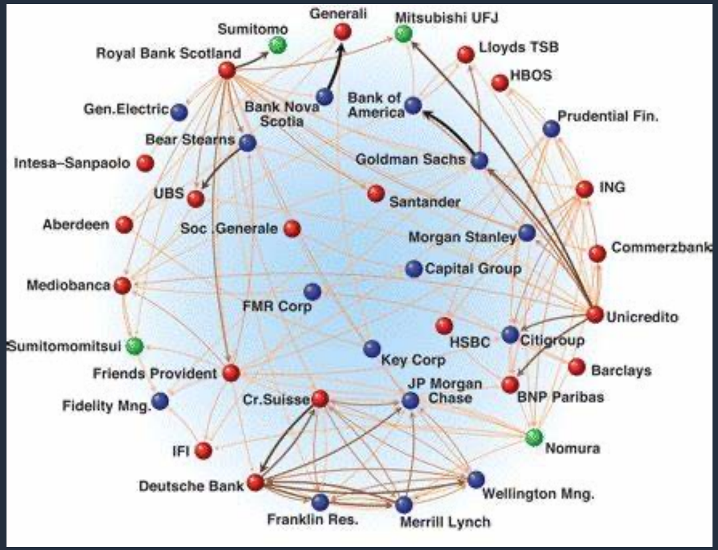
Agenda

- Graph
- Graph Machine Learning (GML)
- Graph Neural Network (GNN)
- Using GNN in Real World
- GraphStorm Framework
- Break
- Hands on GraphStorm

Graph

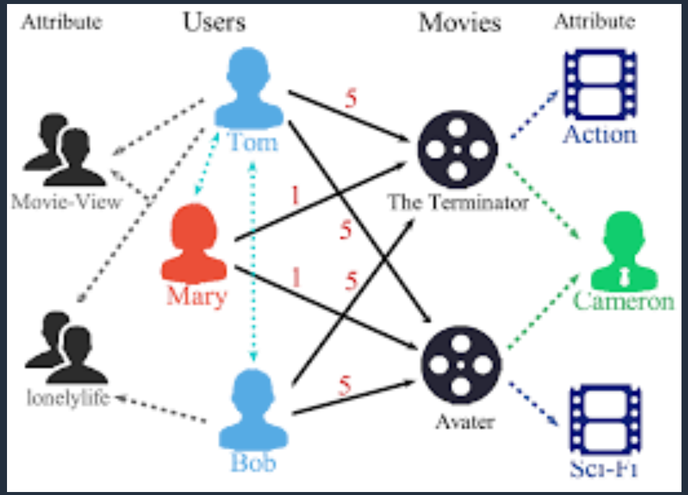
What is Graph?

Graph is the language of complex interactions

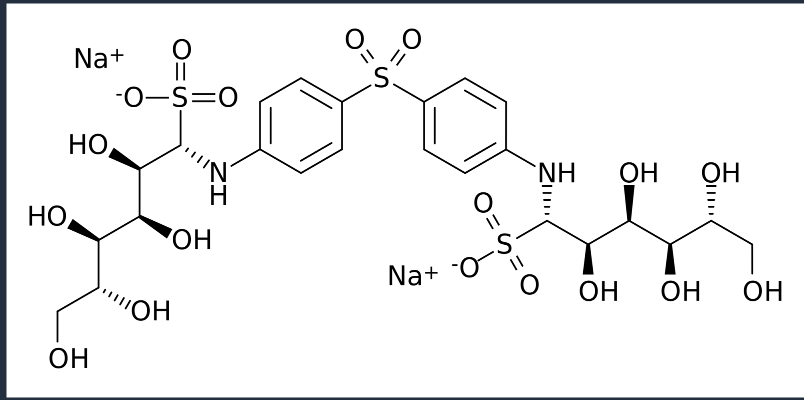


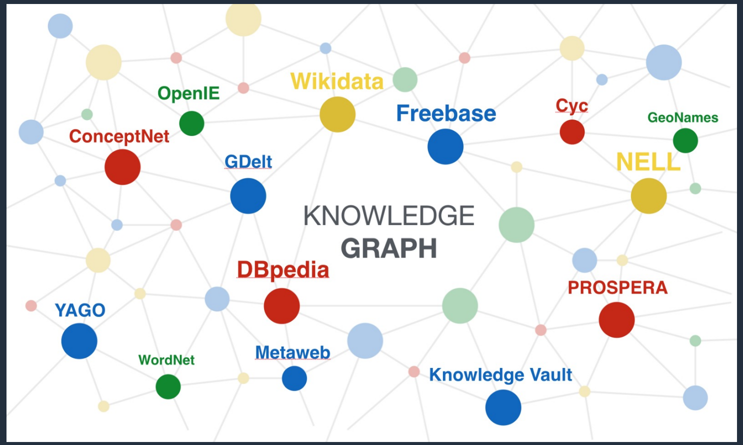
Financial transactions network


Social network



Collaborative knowledge graph


Chemical molecule



Knowledge graph

Concepts and terms

Graph vs Image

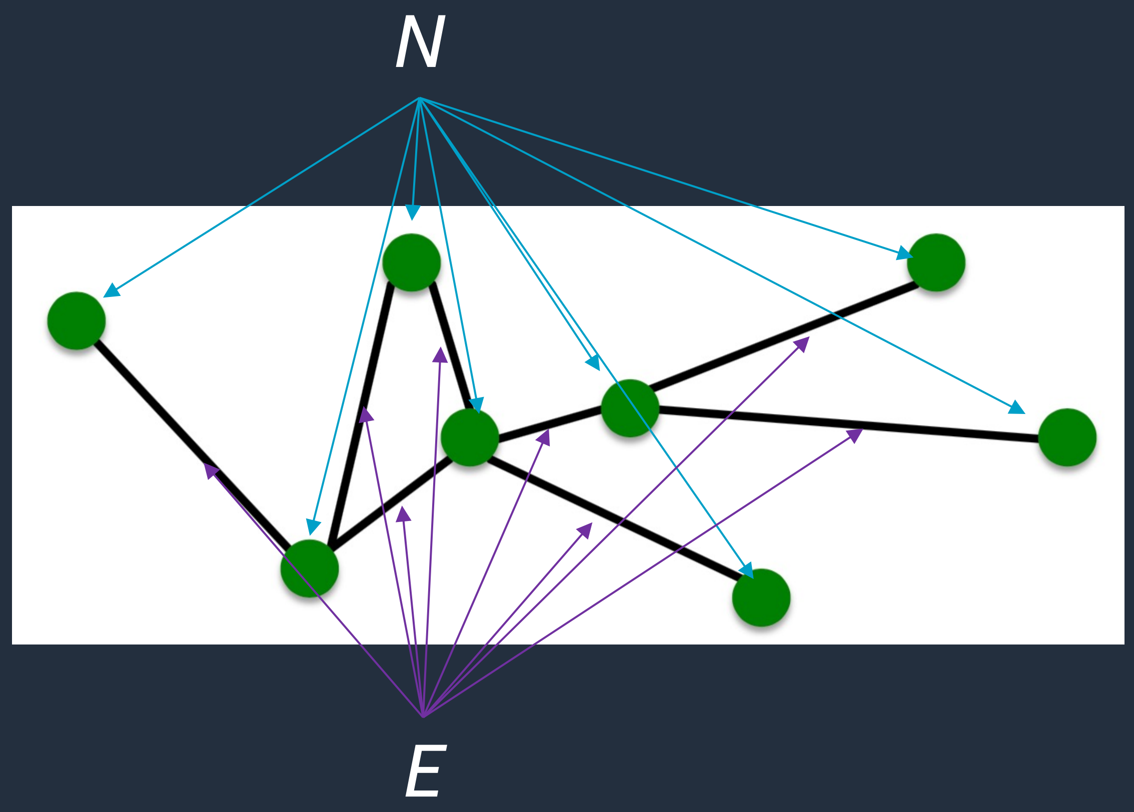
“图”

(a) ImageNet Synset: One sample image from each category
(b) Corel-1000 Dataset: Sample images from each category
(c) Caltech-256 Dataset: One sample image from each category
(d) Caltech-101 Dataset: One sample image from each category

Concepts and terms

Node vs Vertex

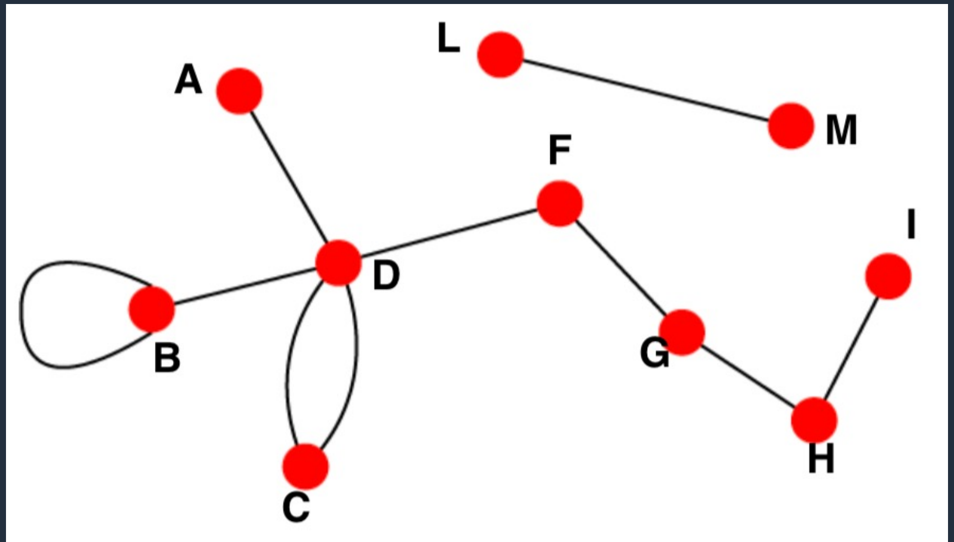
Edge vs Link

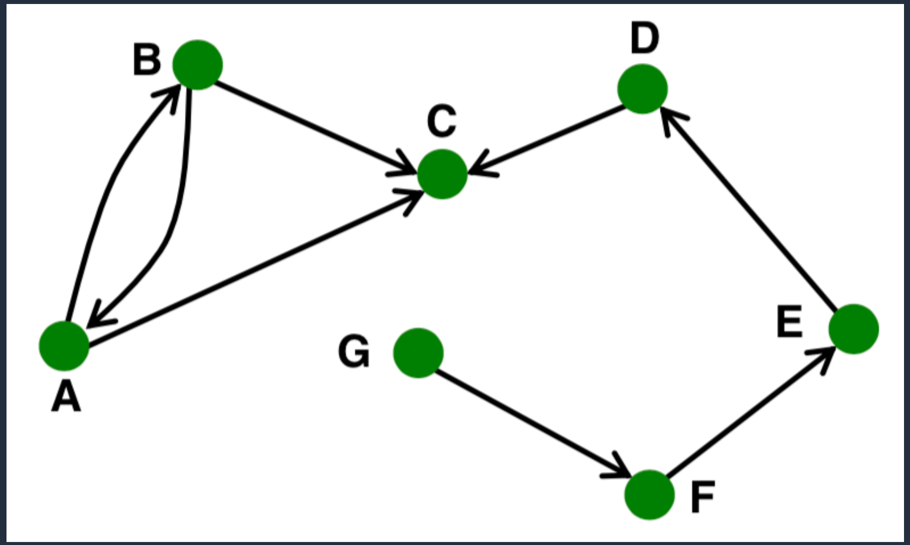


$\rightarrow G(N, E)$

Concepts and terms

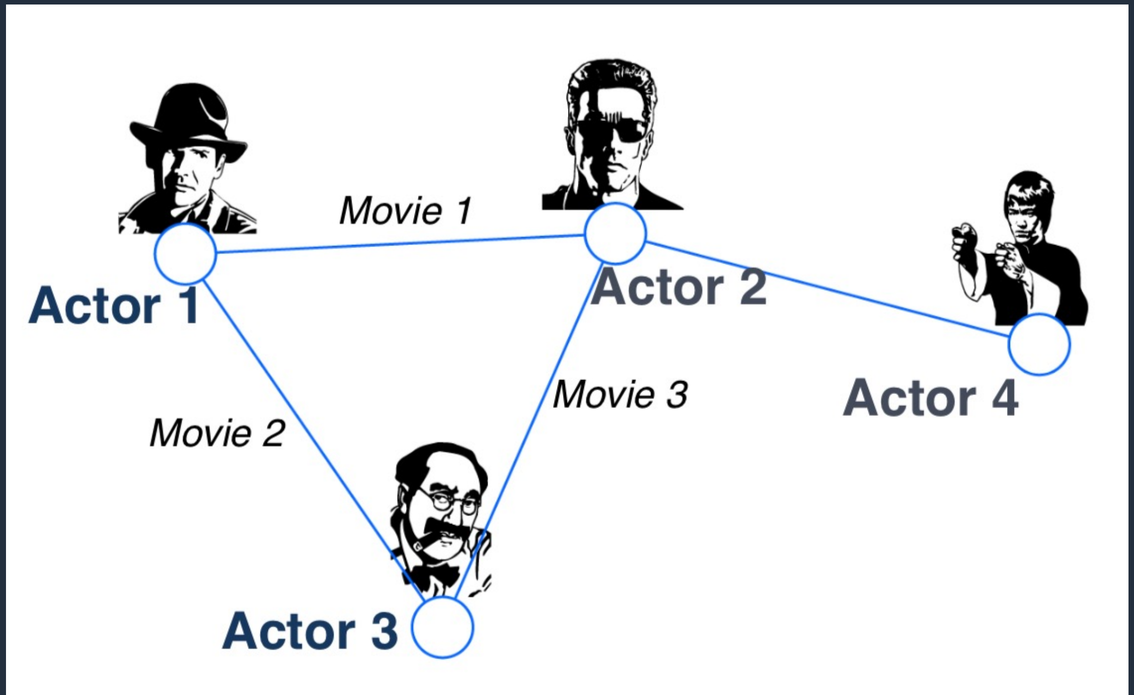
Undirected vs Directed

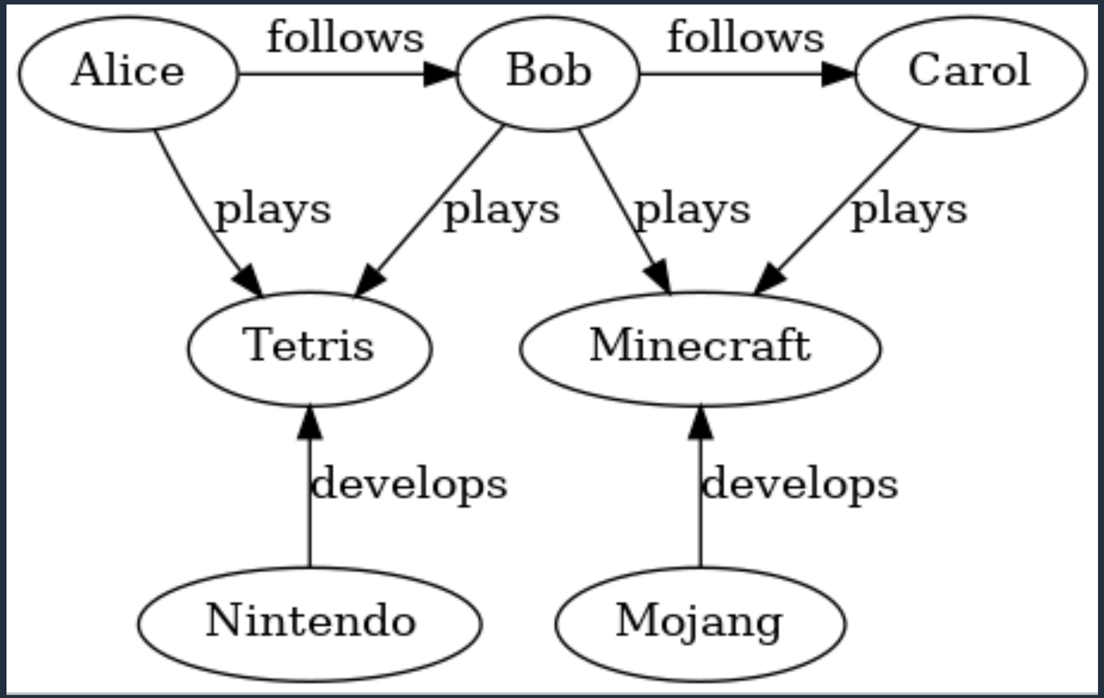




Concepts and terms

Homogeneous vs Heterogeneous





Concepts and terms

Adjacency Matrix vs Edge List

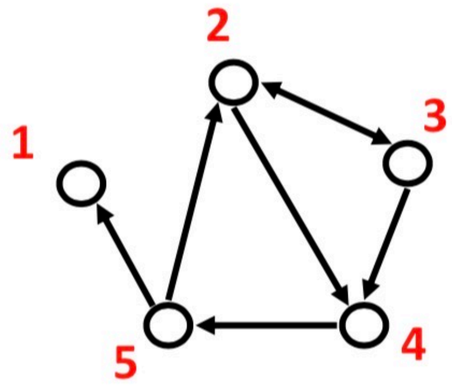
$A_{ij} = 1$ if there is a link from node i to node j

$A_{ij} = 0$ otherwise

$$A = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix} \qquad A = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

Represent graph as a set of edges:

- (2, 3)
- (2, 4)
- (3, 2)
- (3, 4)
- (4, 5)
- (5, 2)
- (5, 1)



Concepts and terms

Features: Nodes Edges

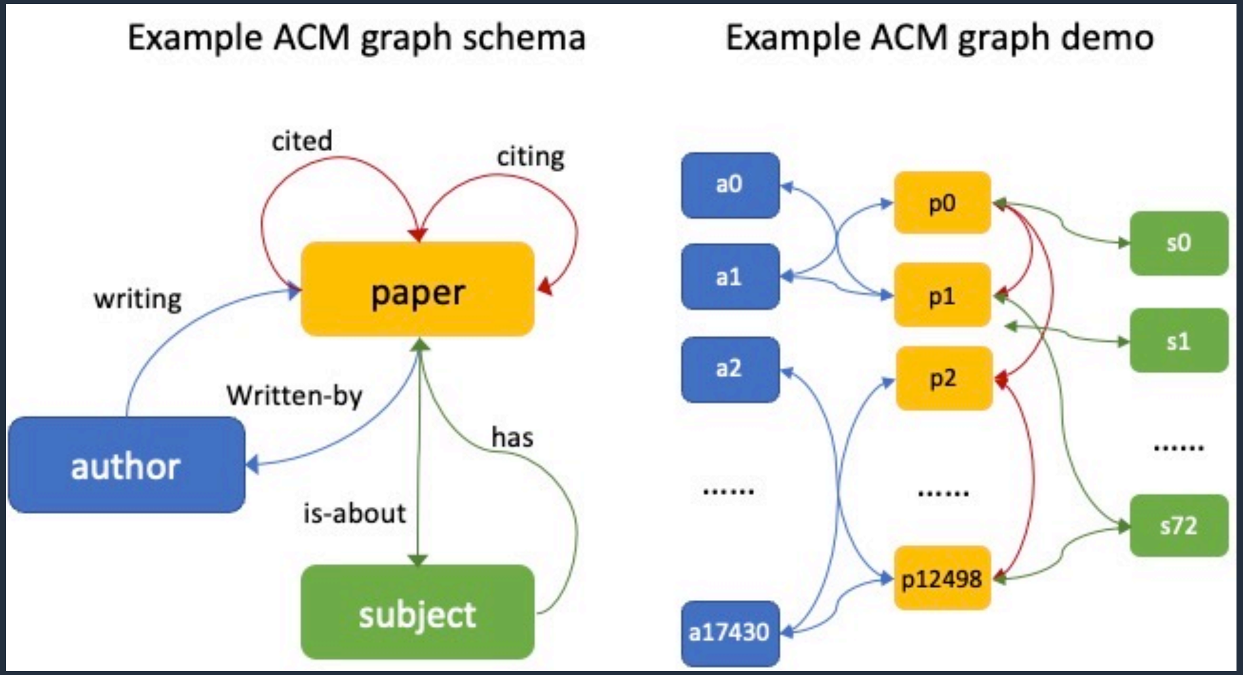
Graph Edge List	
src	dst
U1	P1
U1	P2
U1	U4
…	…
U3	P11
U5	P11
U5	P12

Node Features				
Node	Label	nf1	nf2	…
U1	0	10	0.5	
P1				
…		…	…	
U5	1	50	0.9	
P12				

Edge Features				
src	dst	ef1	ef2	…
U1	P1	0.1	L	
U1	P2	0.2	C	
U1	U4	1.4	C	
…	…	…	…	
U3	P11	…	…	
U5	P11	…	…	
U5	P12	…	…	

Concepts and terms

Heterogeneous Graph & Features



Node Parquets

author node parquet

node_id	feat		
a0	...	...	...
a1	...	...	...
a2	...	256D	...
…	...	...	...
a17430	...	...	...

paper node parquet

node_id	feat			label
p0	...	...	...	11
p1	...	...	...	2
p2	...	256D	...	0
…	...	...	...	...
p12497	...	...	...	11

subject node parquet

node_id	feat		
s0	...	...	...
s1	...	...	...
s2	...	256D	...
…	...	...	...
s71	...	...	...

Edge Parquets

author,writing,paper edge parquet

source_id	dest_id
a148	p0
a148	p1
…	...
a4653	p11567

paper, citing, paper edge parquet

source_id	dest_id	label
p74	p25	1
p8	p1	1
…	...	...
p963	p6607	1

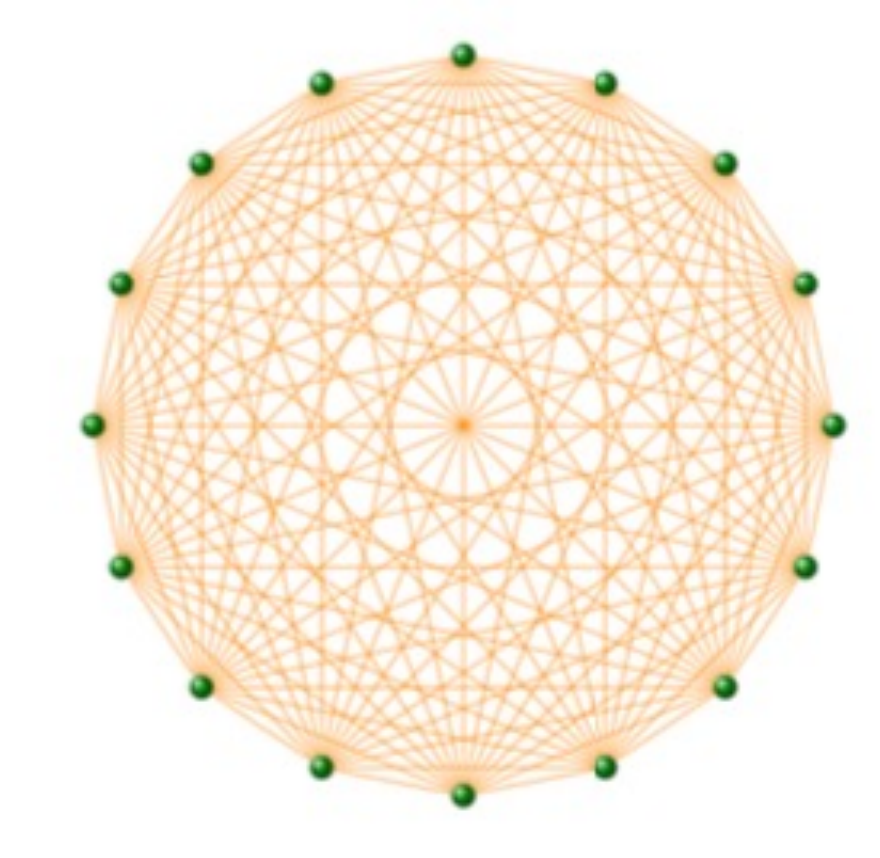
……

subject,has,paper edge parquet

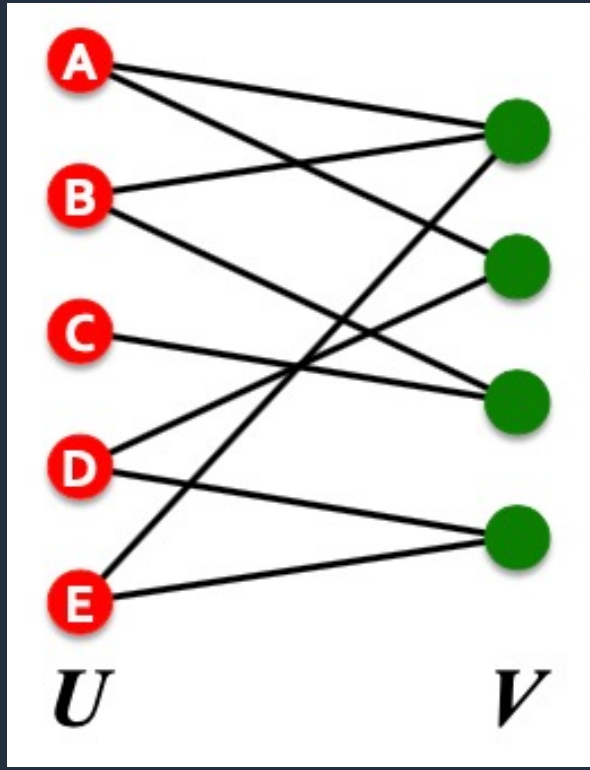
source_id	dest_id
s0	p0
s3	p443
…	...
s56	p11567

Concepts and terms

Completed Graph



Bipartite



Concepts and terms

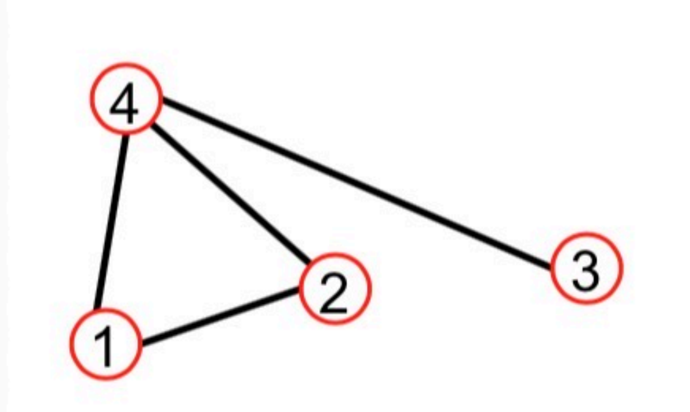
Node Degree

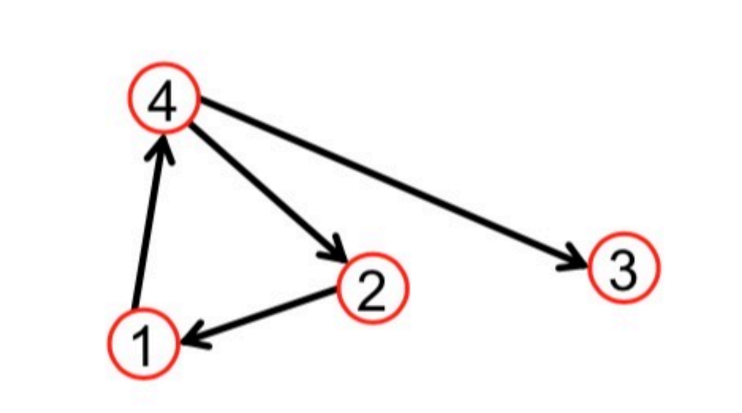
The number of edges adjacent to a node, e.g.

$$k_4 = 3$$

In directed graphs, a node has an in-degree and out-degree. And total degree is the sum of in- and out-degrees. e.g.

$$k_4^{\mathrm{in}} = 1,\ k_4^{out} = 2,$$
$$k_4^{total} = 3$$

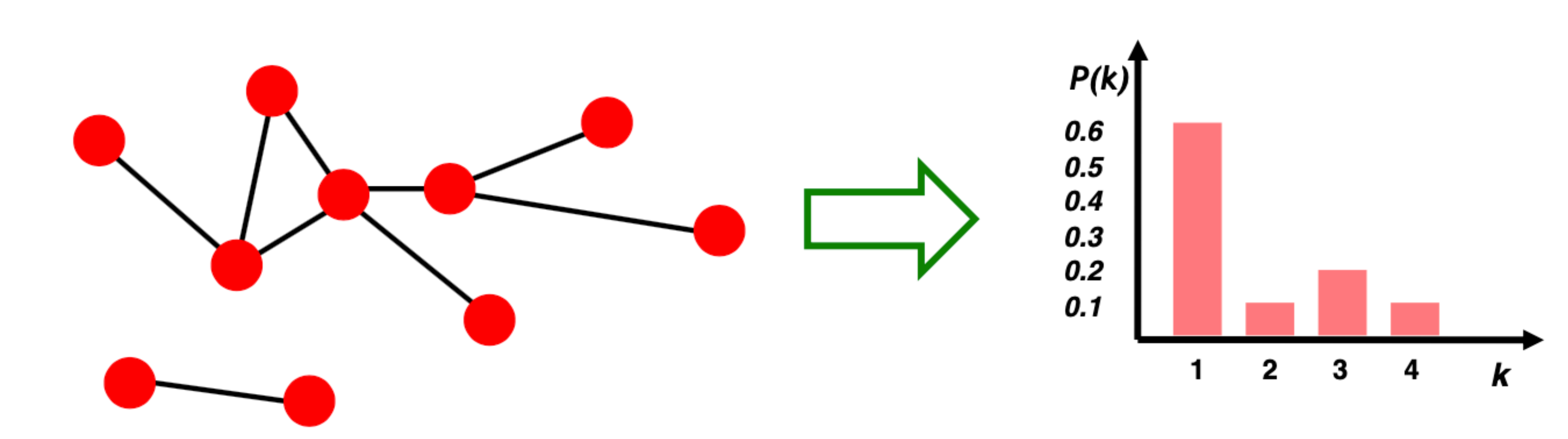




Concepts and terms

Degree Distribution $P(k)$

Probability that a randomly chosen node has degree k, $N_k = \# \, nodes \; with \; degree \; k$. Normalized $P(k) = N_k/N$



Concepts and terms

Path P

A path is a sequence of nodes in which each node is linked to the next one. $P_n = \{i_0, i_1, i_2, \ldots i_n\}$. Path length $h = \#\ of\ edges\ in\ a\ path$. In directed graph, paths need to follow the direction of edges.

The distance between two nodes is the length of the shortest path.

The diameter of a graph is the maximum distance between any pair of nodes.
Average path length of connected graph

The average path length for a connected graph (need to be connected).

$$\bar{h} = \frac{1}{2E_{max}} \sum_{i,j \neq i} h_{ij}\ ,where\ h_{ij}\ is\ the\ distance\ of\ node\ i\ and\ j, and\ E_{max} = n(n-1)/2$$

Concepts and terms

Cluster Coefficient C

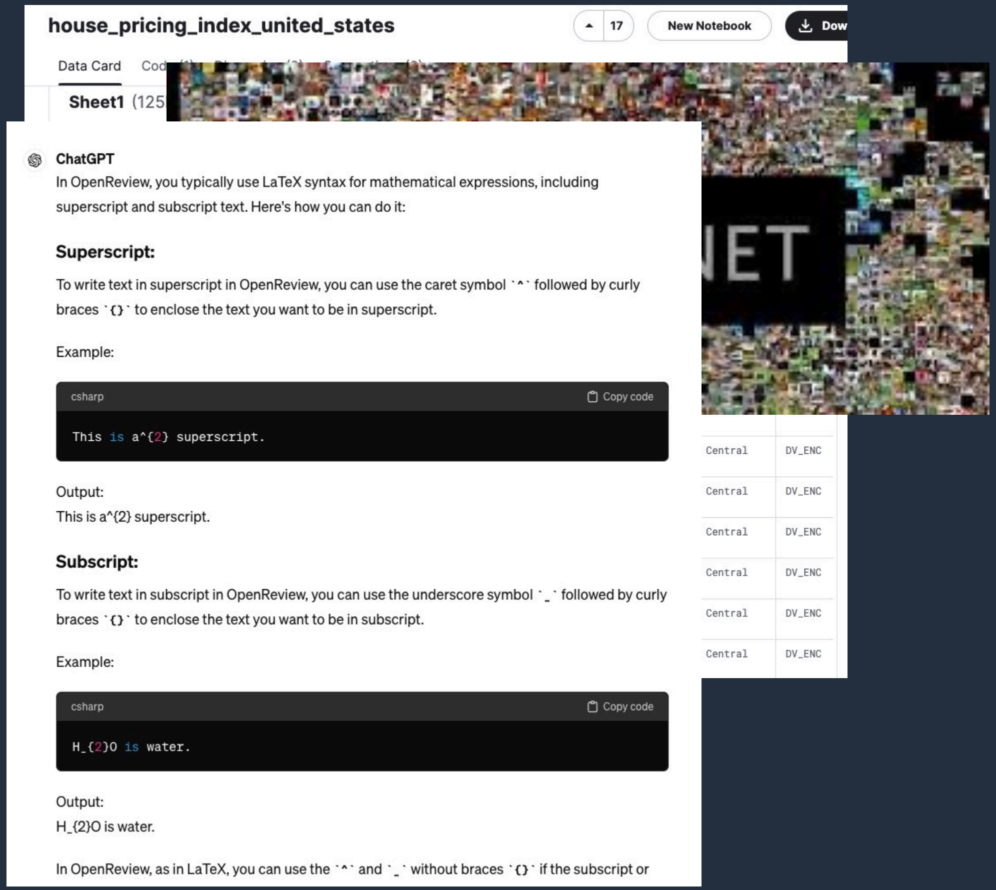
A Cluster Coefficient is the ratio of the number of edges among neighbors of node i and the max number of edges between all neighbors: $C_i = \frac{2e_i}{k_i(k_i - 1)}$.

Average Clustering Coefficient is the mean of all nodes' Clustering Coefficient.

Graph Machine Learning (GML)

Recall Machine Learning (ML)

$$f(X, \theta) \Rightarrow Y$$

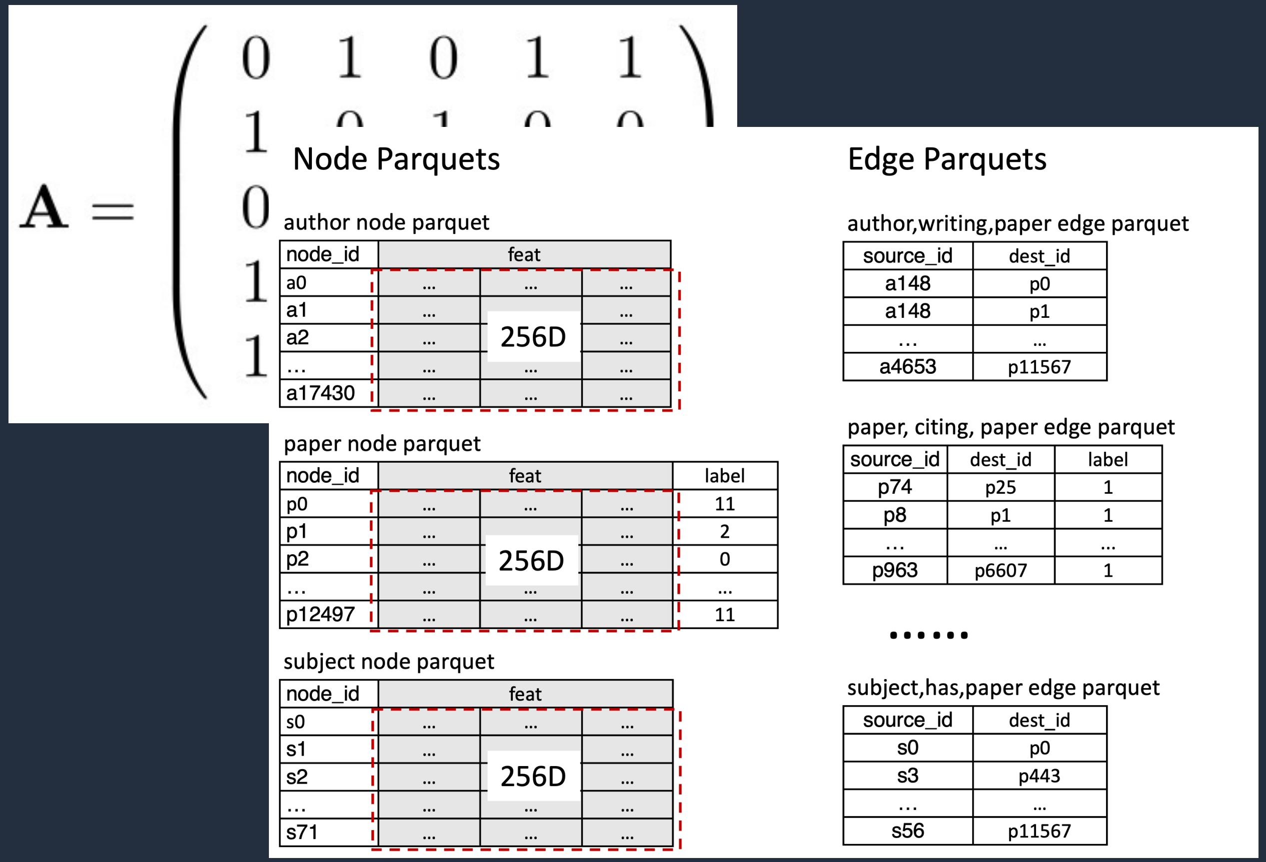


- Logistic Regression
- Support Vector Machine
- Multilayer Perceptron
- Convolutional Neural Network
- Recurrent Neural Network
- Transformer
- …

- Classification
- Regression
- Segmentation
- Detection
- Reconstruction
- …

Graph Machine Learning (GML)

$$f(X, \theta) \Rightarrow Y$$

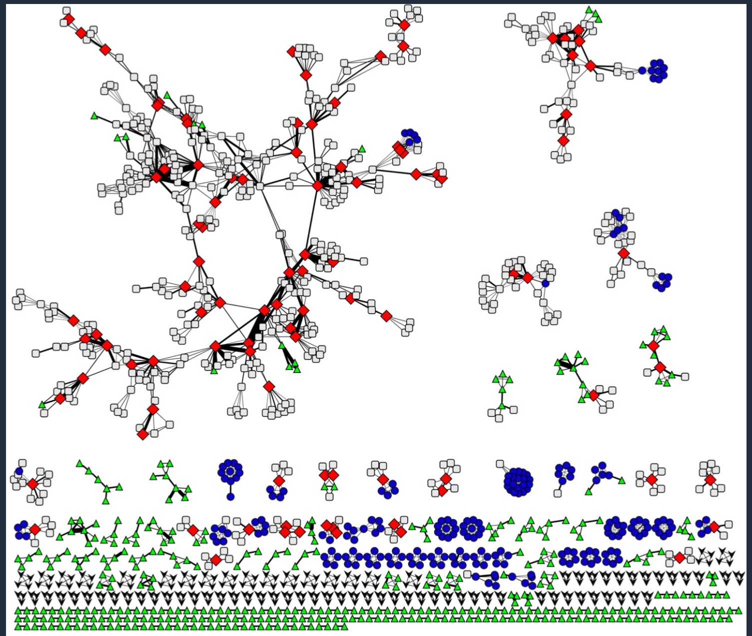


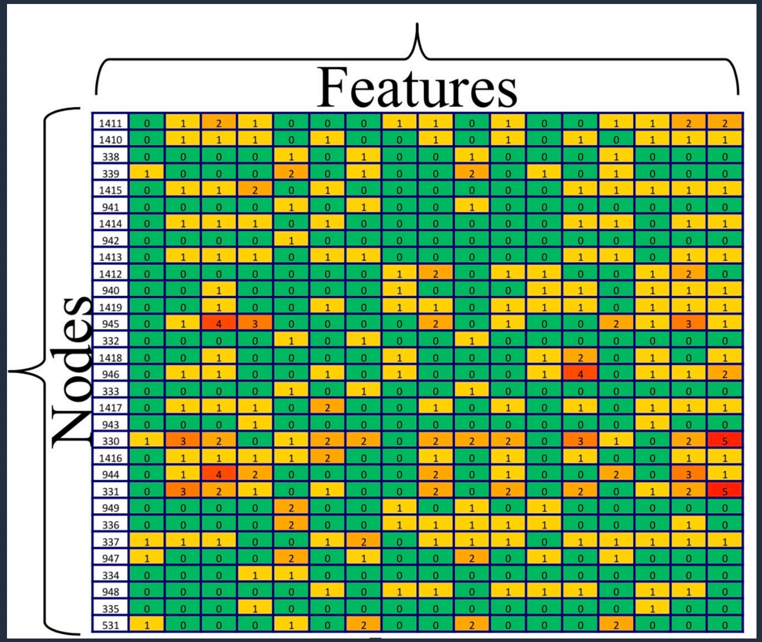
- Node
  - Classification, e.g., detecting malicious accounts
  - Regression, e.g., predicting customer rating
- Edge
  - Classification, e.g., detecting suspicious transactions
  - Regression, e.g., predicting when will traffic jam start
  - Link Prediction e.g., recommending friends
- Graph
  - Classification, e.g., predicting if a new compound is toxic
  - Regression e.g., predicting medicine molecular solubility

GML before

- Generate embeddings by manual feature engineering

- Automatically generate embeddings using unsupervised dimensionality reduction approaches



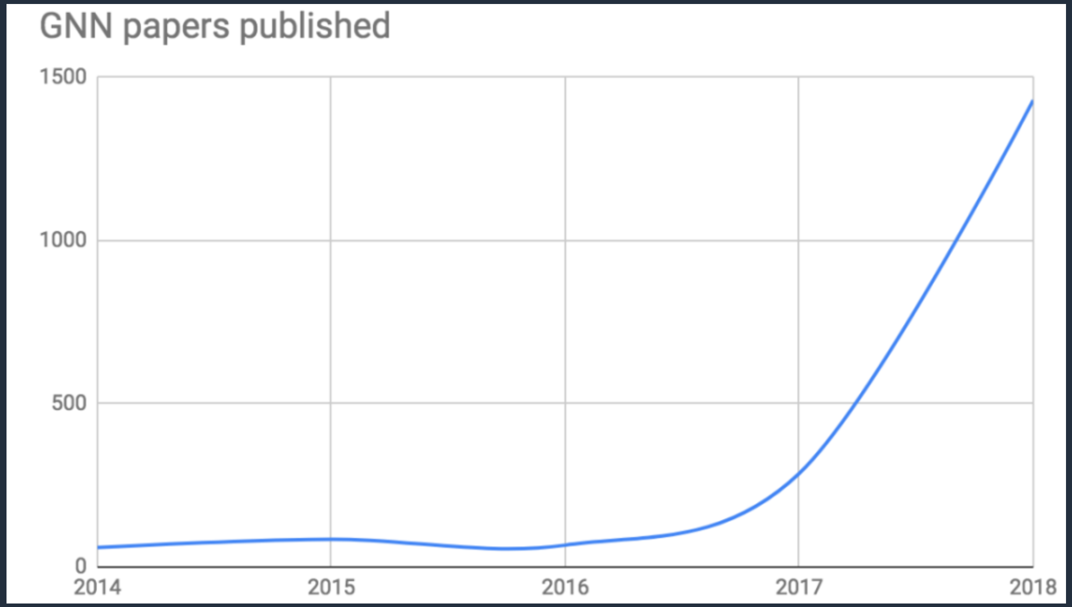


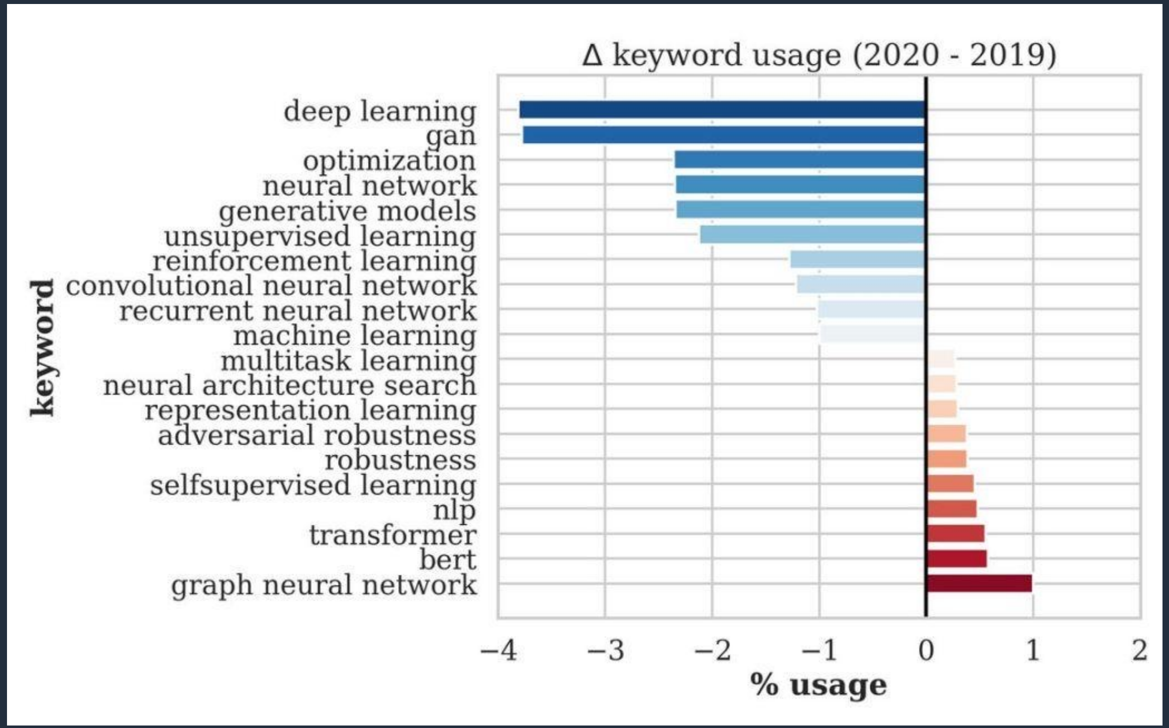
Graph Neural Networks (GNNs)

Graph Neural Network (GNN)

DL(NN) + Graph => GNN

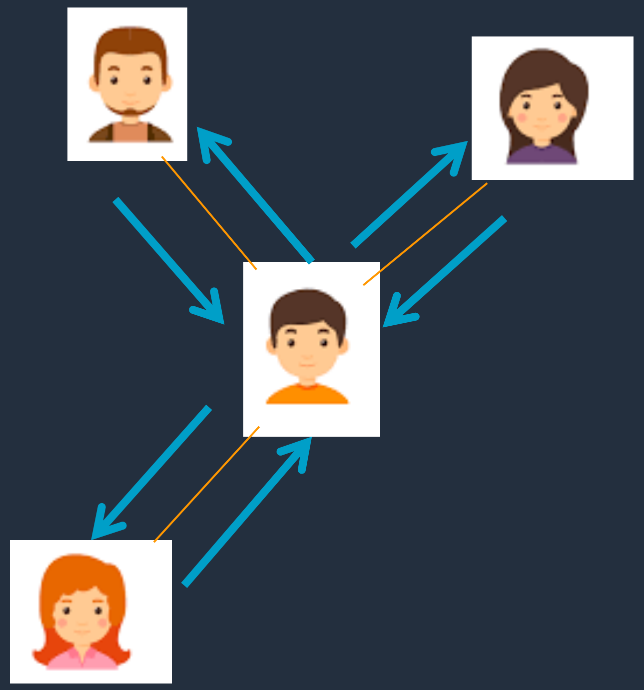
A family of (deep) neural networks that learn node, edge, and graph features





How does GNN work

Message-passing & Aggregation



Reduce function

Message function

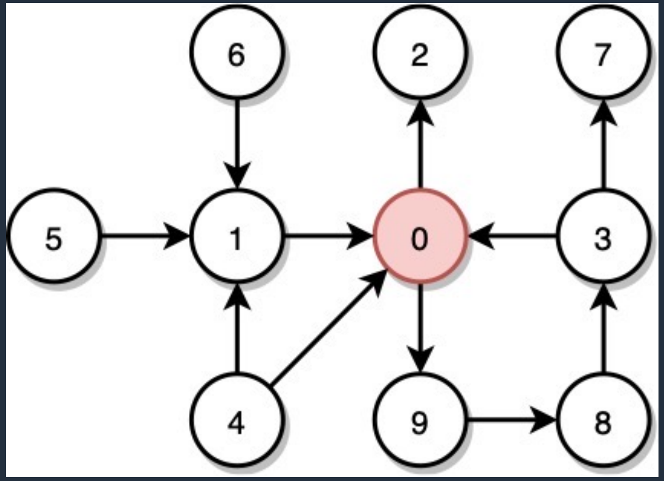
$$m_v^{t+1} = \sum_{w \in N(v)} M_t(h_v^t, h_w^t, e_{vw}) \qquad (1)$$

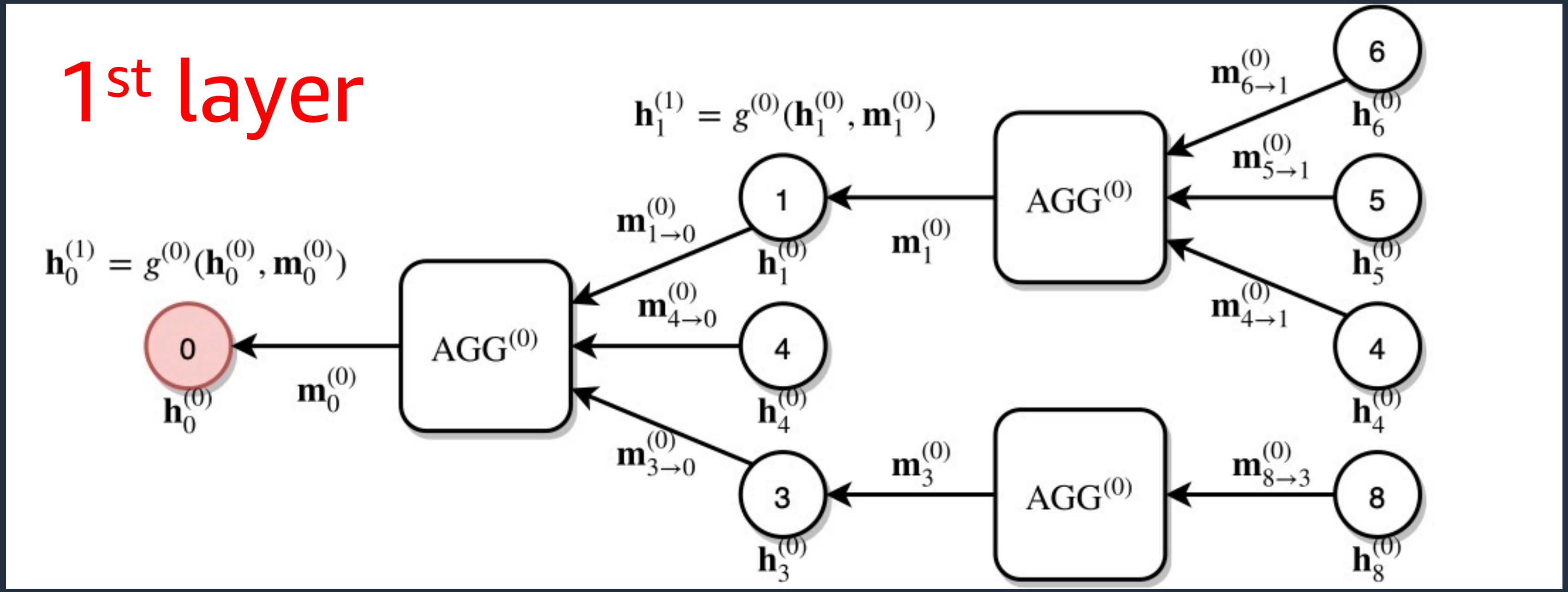
$$h_v^{t+1} = U_t(h_v^t, m_v^{t+1}) \qquad (2)$$

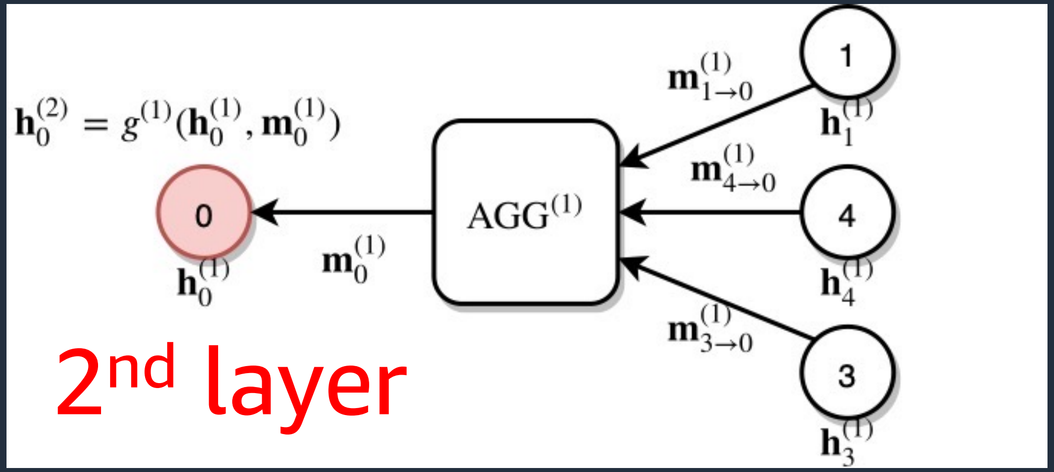
Update function

Stacked Multiple GNN layers

GNNs can *integrate* topologically distant information in a non-linear fashion.







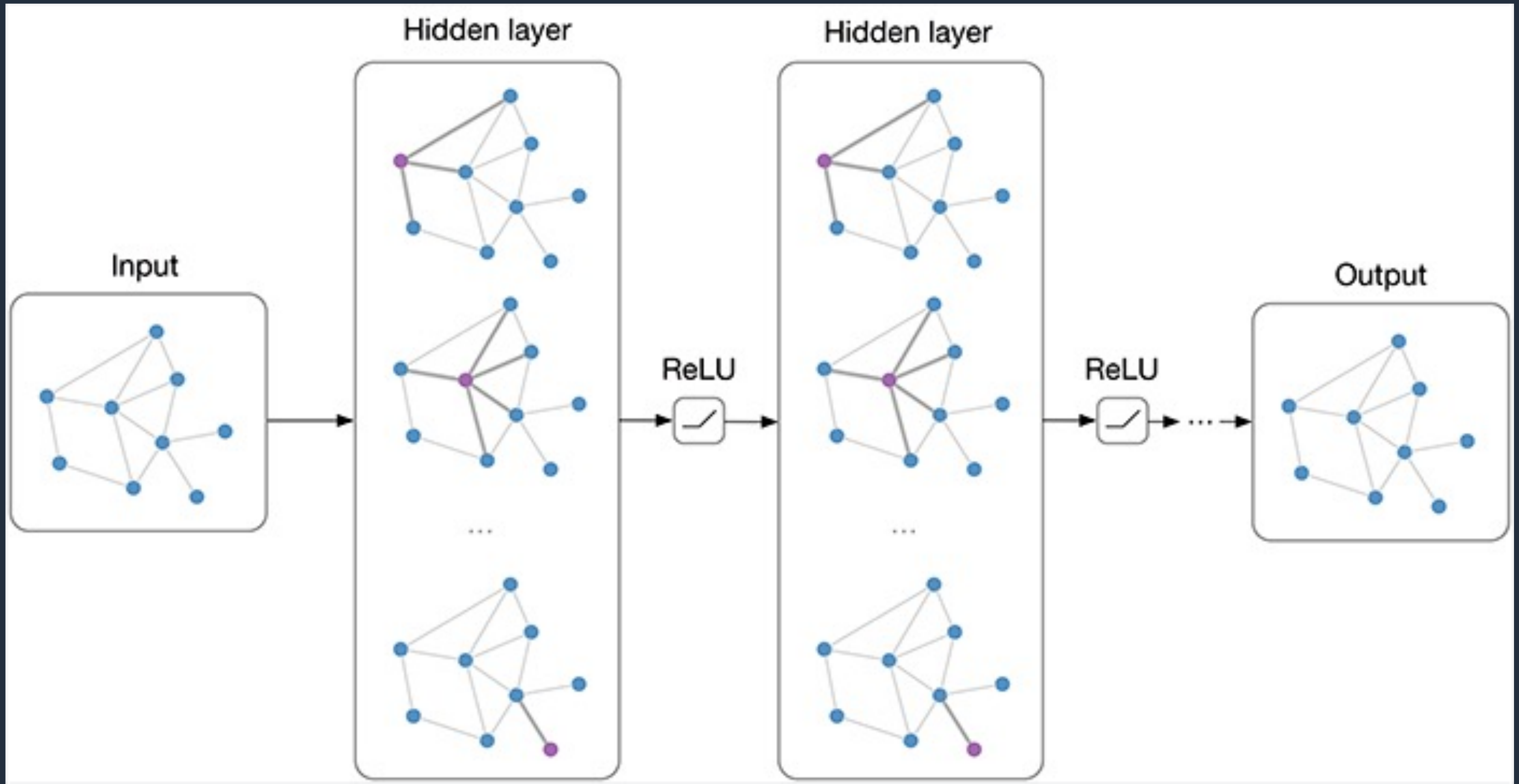
Common GNN Models — GCN

GCN (Graph Convolutional Network) for homogeneous graphs

$$M_{vw}^{(l)} = \frac{h_w^{(l-1)}}{d_v+1}$$

$$m_v^{(l)} = \sum_{w \in N(v) \cup \{v\}} M_{vw}^{(l)}$$

$$h_v^{(l)} = \phi(m_v^{(l)} W^{(l)})$$



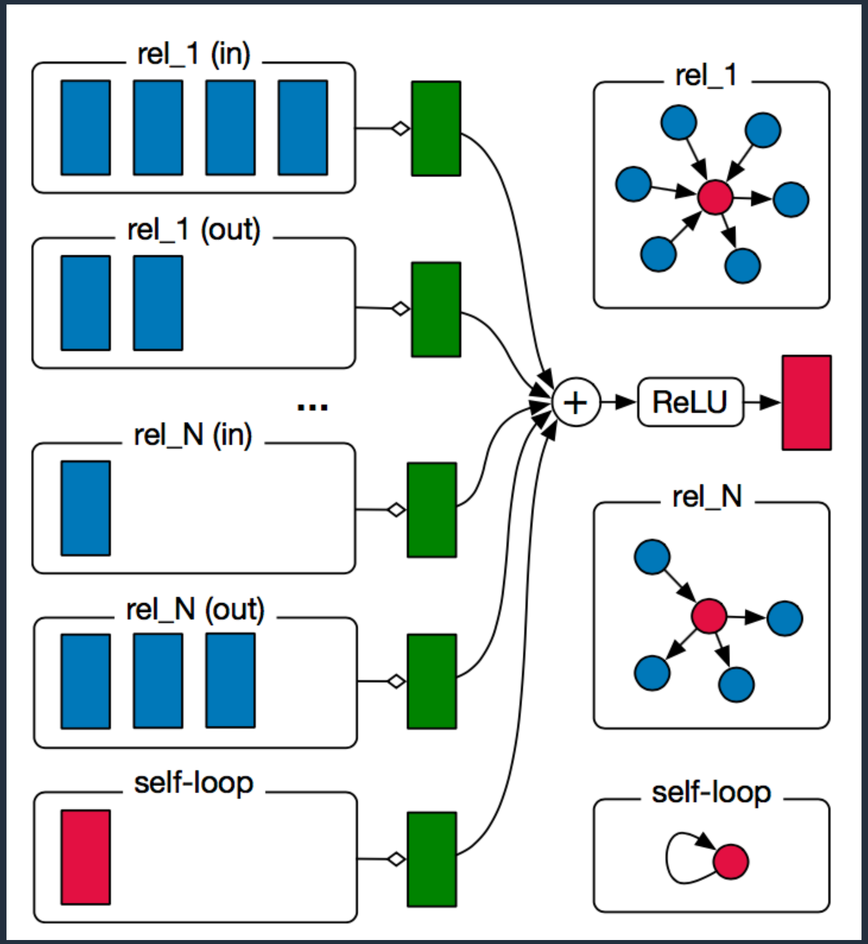
Common GNN Models — RGCN

Relational graph convolution networks (RGCN) handles graphs whose nodes are connected with different relations.

$$M_{vw}^{(l)} = \frac{1}{c_{v,r}} W_r^{(l)} h_w^{(l-1)}$$, *r is the relation of* e_{vw}

$$m_v^{(l)} = \sum_{w \in N(v) \cup \{v\}} M_{vw}^{(l)}$$

$$h_v^{(l)} = \sigma(m_v^{(l)} W^{(l)})$$



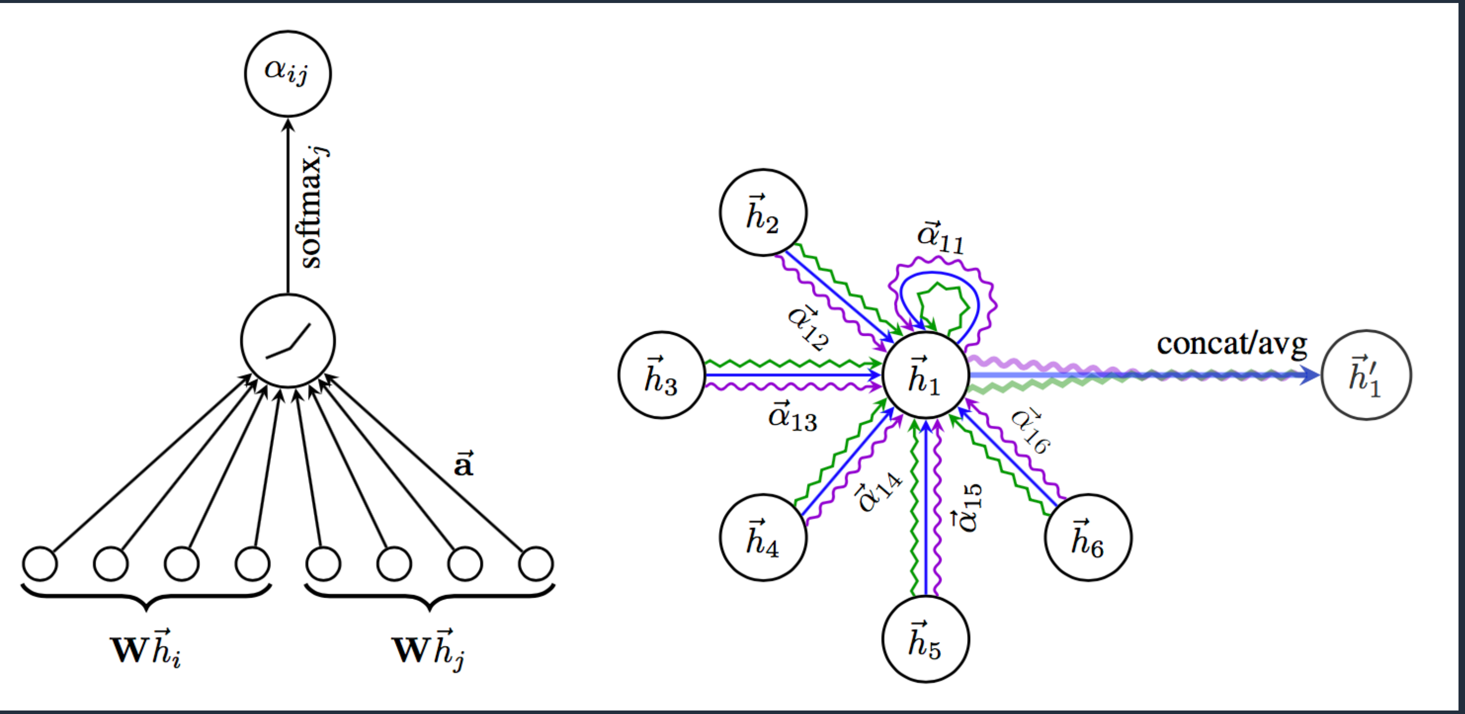
Common GNN Models — GAT

Graph AttenTion Network(GAT) provides weighted sum over the neighborhood—
Enables to selectively integrate information.

$$M_{vw}^{(l)} = \alpha_{vw} h_w^{(l-1)}$$

$$m_v^{(l)} = \sum_{w \in N(v) \cup \{v\}} M_{vw}^{(l)}$$

$$h_v^{(l)} = \phi(m_v^{(l)} W^{(l)})$$



$$\alpha_{vw} = \frac{\exp(\text{LeakyReLU}(\vec{a}^T[W\vec{h}_v || W\vec{h}_w]))}{\sum_{k \in N_v} \exp(\text{LeakyReLU}(\vec{a}^T[W\vec{h}_v || W\vec{h}_k]))}$$

GNN References

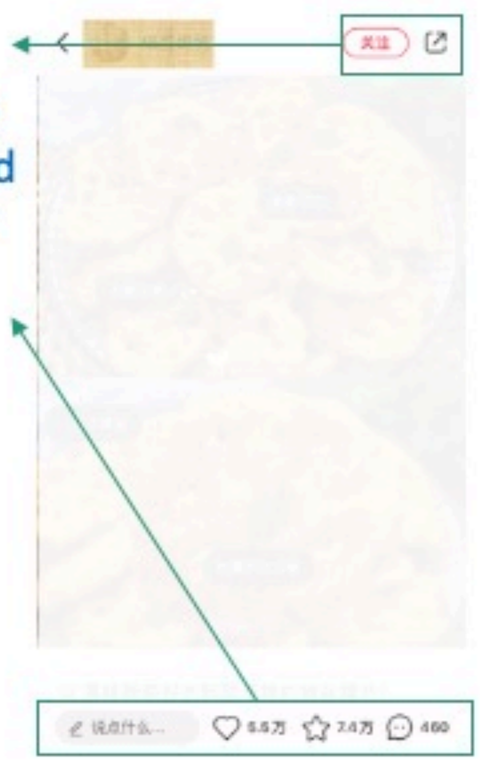
- GNN Libraries
    - DGL (Deep Graph Library): https://www.dgl.ai
    - PyG (Pytorch Geometric): https://pyg.org/
    - TF_GNN (TensorFlow GNN): https://github.com/tensorflow/gnn
- Online Books
    - Deep Learning on GraphS (https://yaoma24.github.io/dlg_book/)
    - Graph Neural Networks (https://graph-neural-networks.github.io/)
- Online Courses
    - Stanford CS224W: ML with Graphs(https://web.stanford.edu/class/cs224w/)

Real Cases of Using GNN

GNN Helps to Detect ‘Suspicious’ Interaction

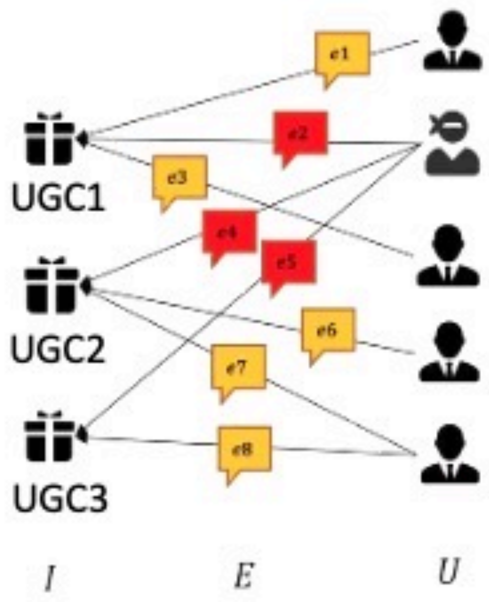
Problem:

Interactions between users and UGCs could be ‘suspicious’ with hidden agendas.

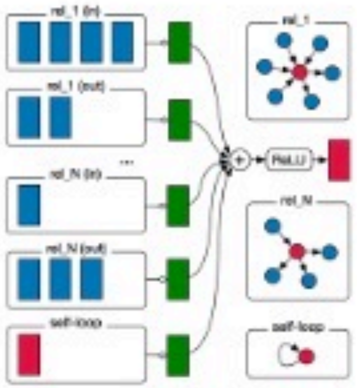


Bipartite:

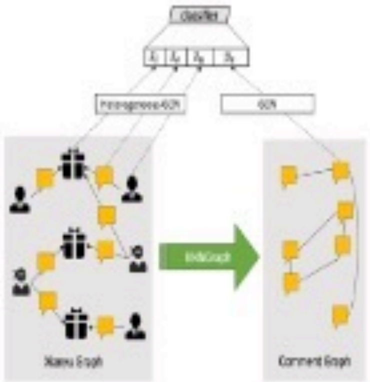
User <-> UGCs



Improvements:



Focus on User:
As a node classification task, RGNN model out-performs baseline models in all settings



Focus on Behavior:
SOTA model out-perform baseline models in some settings, and help find **new patterns** of ‘suspicious’ that previous rule-based methods can not touch.

GNN Helps to Detect 'Bot' Accounts

Problem:

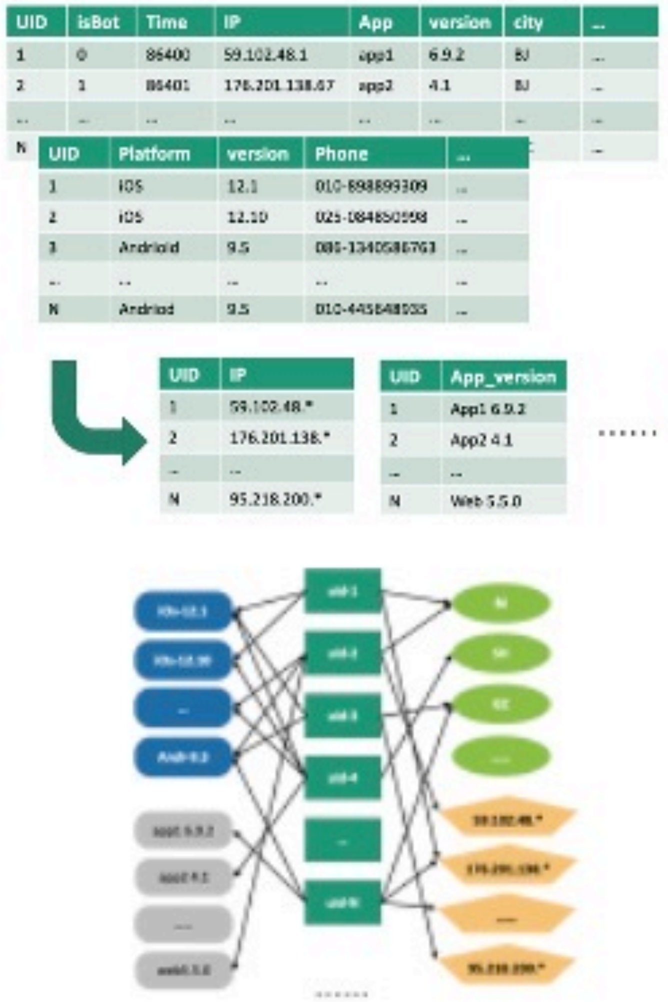
Bots are increasingly impacting the e-comm platform's customer in:

- promo code abuse.
- inventory encumbrance.
- return logistics costs.
- fraud response operations cost.

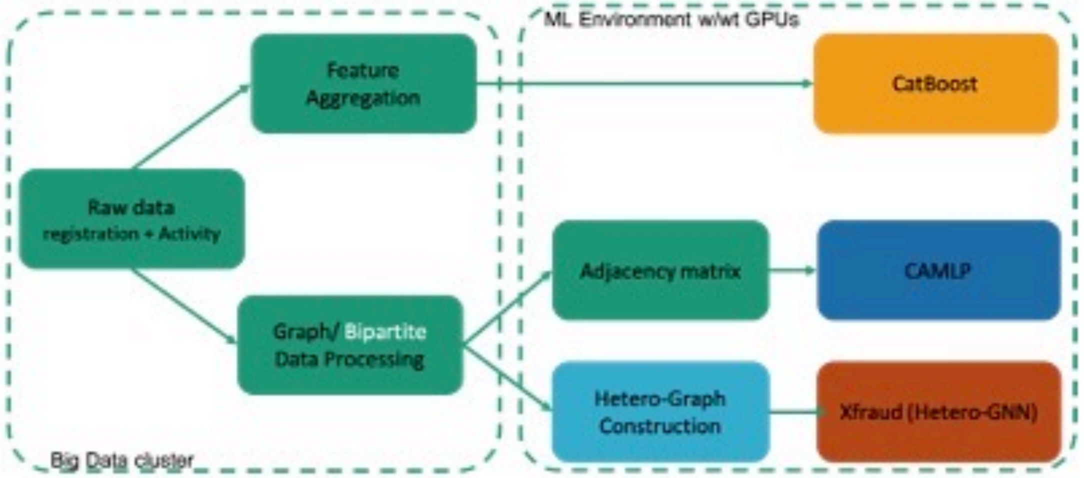
Challenges:

- Account features differ in different life stages.
- New/Inactive accounts having nearly no features.
- Lack of solid labels, particularly newly registered.
- Huge amounts.

Tables -> Bipartite:



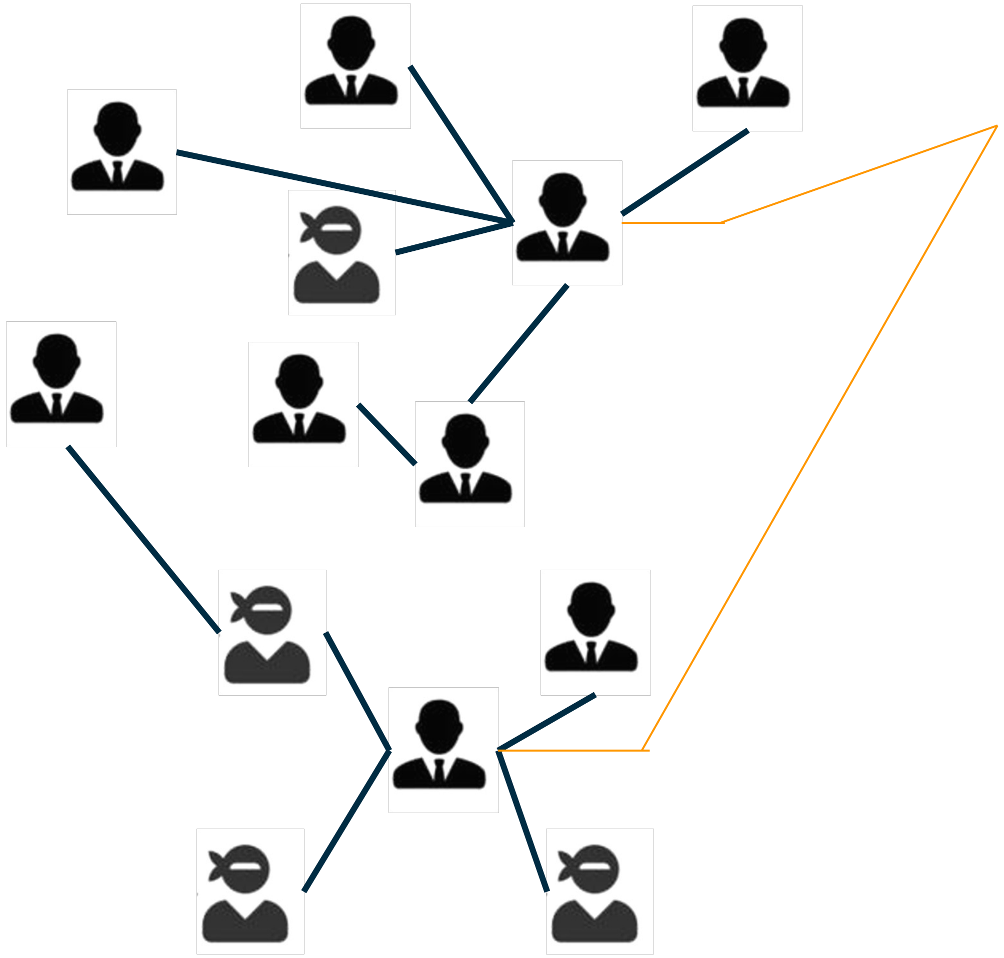
Improvements:



Multiple models all have 11x performance boost than customer's existing models!
GNN and tree-based models work together, helping to handle accounts in their whole-life.

Case 3： A FinTech – Predict Loan Overdue

Problem: Personal loan overdue prediction

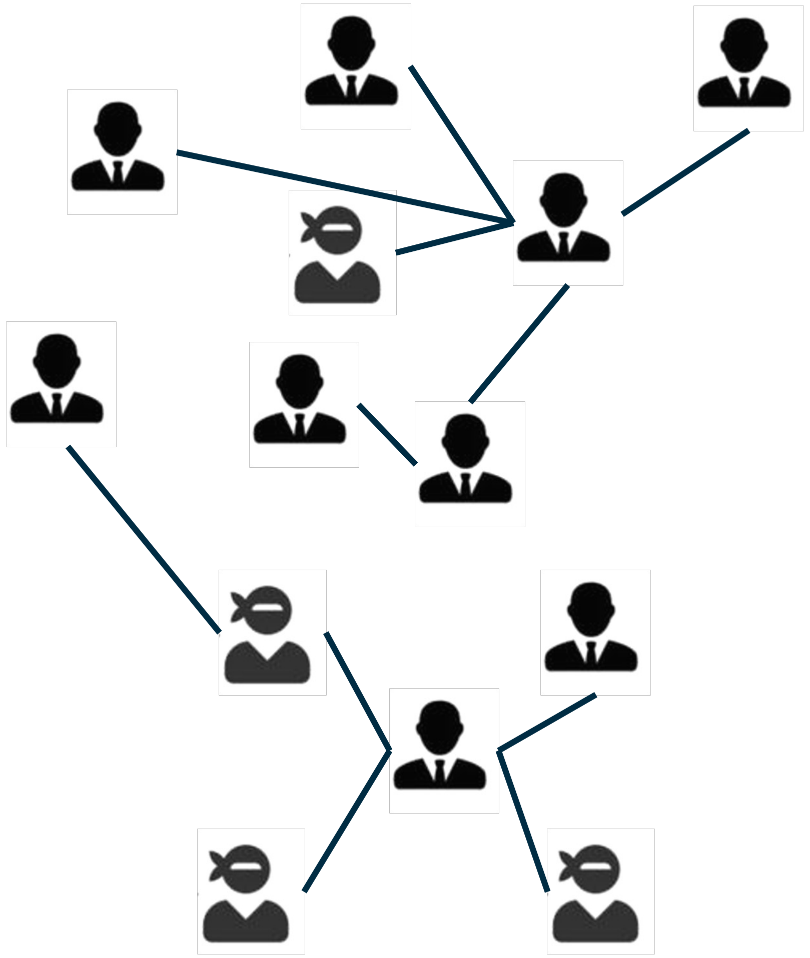


Will the personal load application be overdue?

- Has an xGboost model as baseline, using person’s info, loan history, and social relation as input features
- Hope to fully leverage the social relation about customers to boost prediction performance

The Data - Homogenous graph

Item	Performance Verification
Data range*	7 years
No. Nodes – phone*	26M
No. Edges – contact*	208M
Node features	353 features
Negative nodes	2.3M
Positive nodes	16K

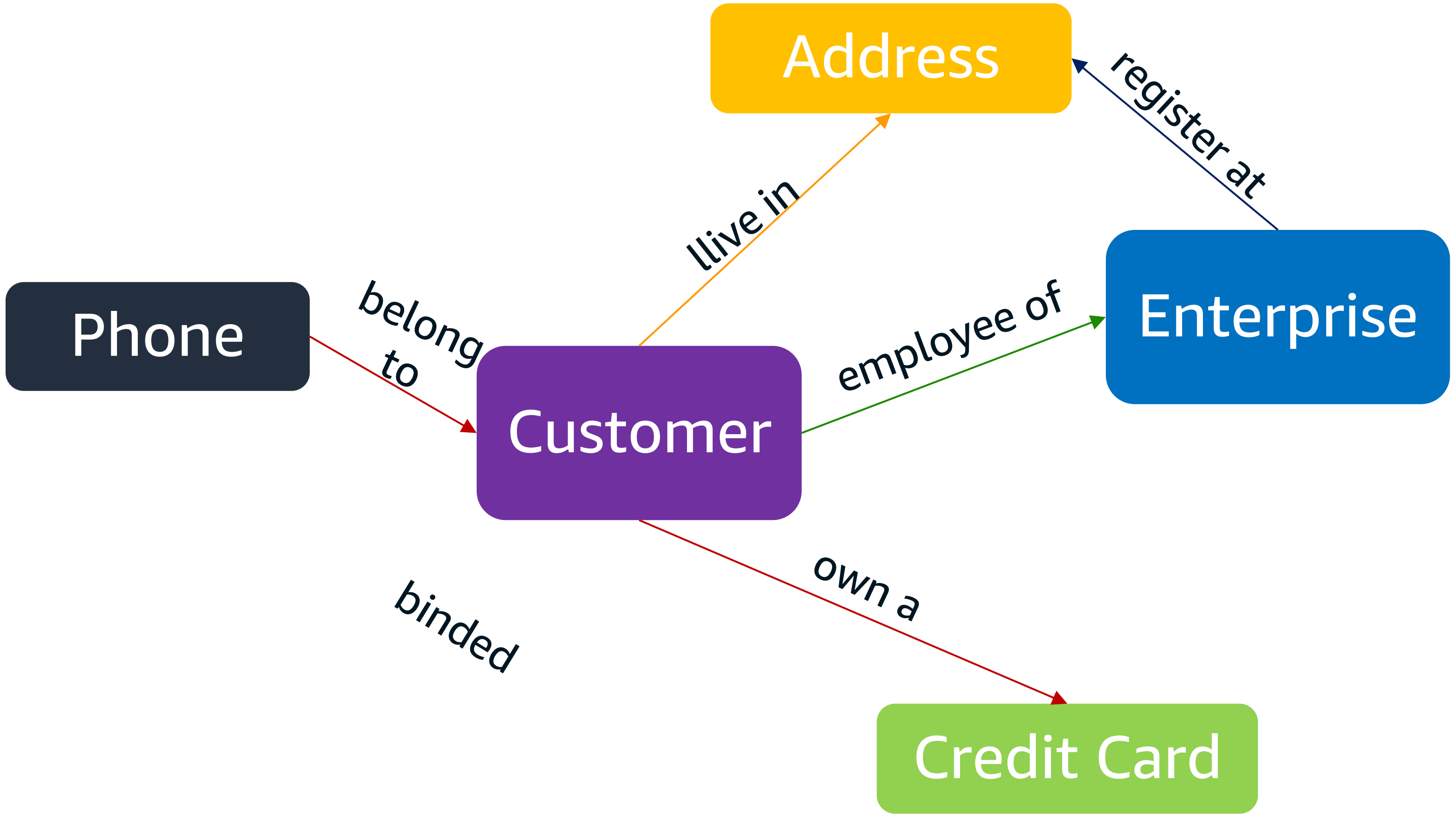


Model Selection — Node Classification

- Most of GNN Models do node classification
  - GCN (Graph Convolutional Network)
  - GraphSage (Simplified GCN)
  - Graph AttenTion Network (GAT)
- Pre-extract higher-order local structure feature
  - Refex, GDV (Graphlet Degree Vector), etc.
- GraphSage model out-performance GCN and GAT, achieving +4% AUC than the xGboost baseline.

Case 4： A Bank – Loan Application Approval

Problem: Loan application approval



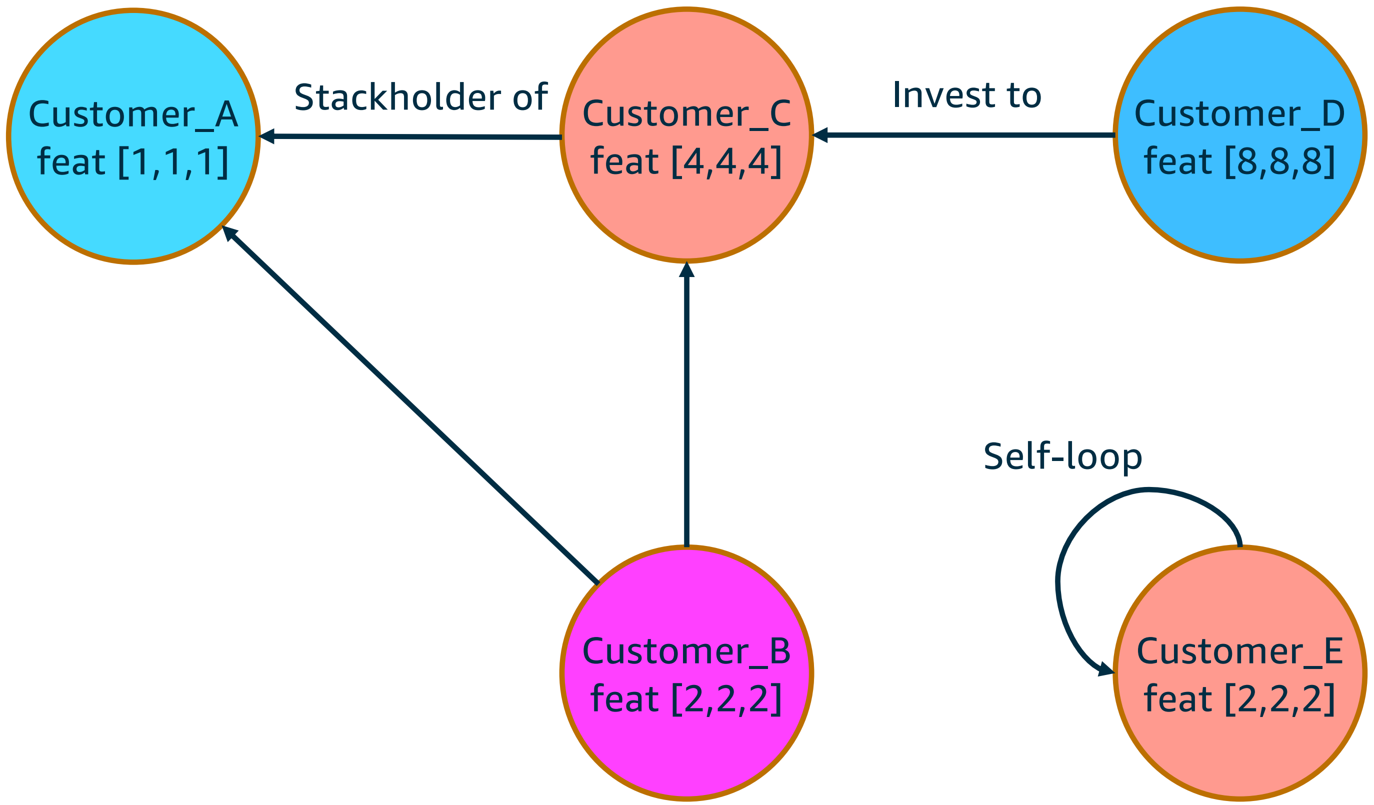
Knowledge Graph in Banking (demo)

The bank has built a knowledge graph about many entities and relations, which could help to predict risk scores for various business scenarios

- Have man-made rules for loan approval decision
- Hope to leverage GNN models to complete the approval in real time

The Data – Knowledge Graph (KG)

- The Graph
  - 2 types of entities
  - ~20 types of relationships
  - contents coming from various sources, e.g. banks, governments, and etc.
- Labels
  - Credit card users' payment history
  - Credit scores from government agencies.

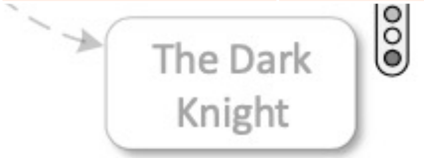


Model Selection —— KG-oriented

- RGCN
  - KG is a specious case of heterogenous graph, where RGCN can work on too
  - Prone to overfitting if the no. of relations is large.
- ICLR 2020 — CompGCN
  - Specially designed for KG
  - Has less parameters, reported SOTA performance

[redacted]

Models	Metric	Train	Test
baseline	ks	0.75	0.7514
	Auc	0.94	0.9226
RGCN	ks	0.48	**0.7984**
	Auc	0.75	**0.9245**



GraphStorm

Challenges of adopting GNN

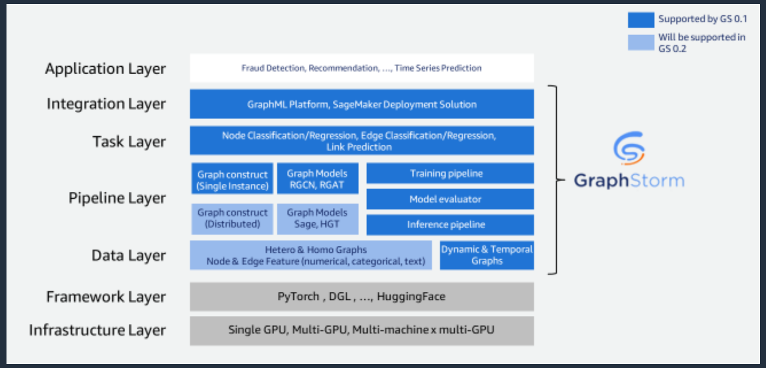
Steep Learning Curve

Complex graph data processing

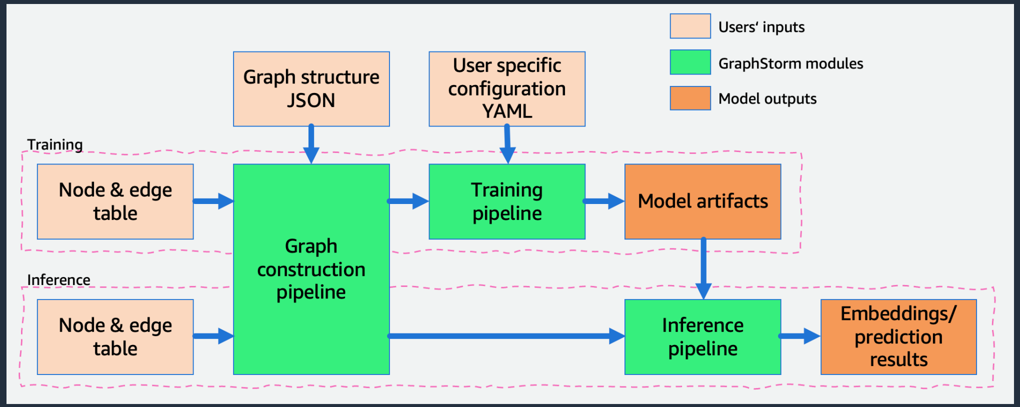
Hard to Scale to extreme large graphs

Fast-track graph ML with GraphStorm: A new way to solve problems on enterprise-scale graphs

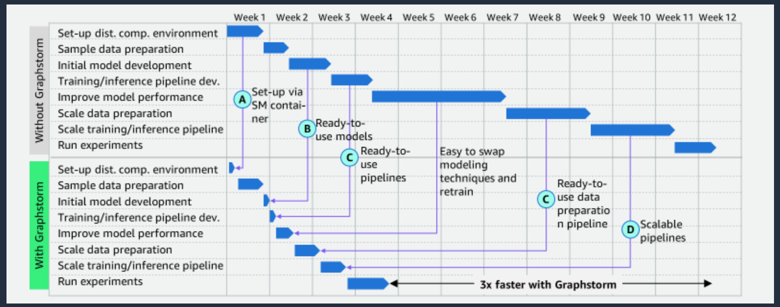
Comprehensive toolbox



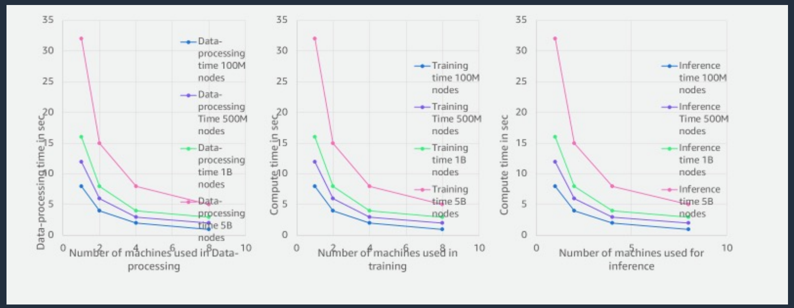
Easy-to-use interface



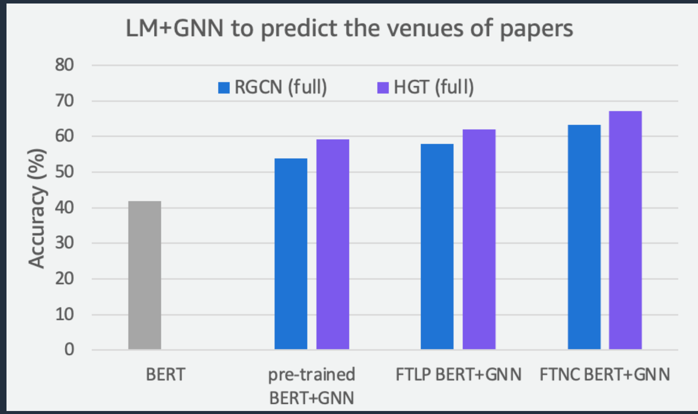
Speed up model development & deployment



Scale to billion-node graphs



Superb model performance

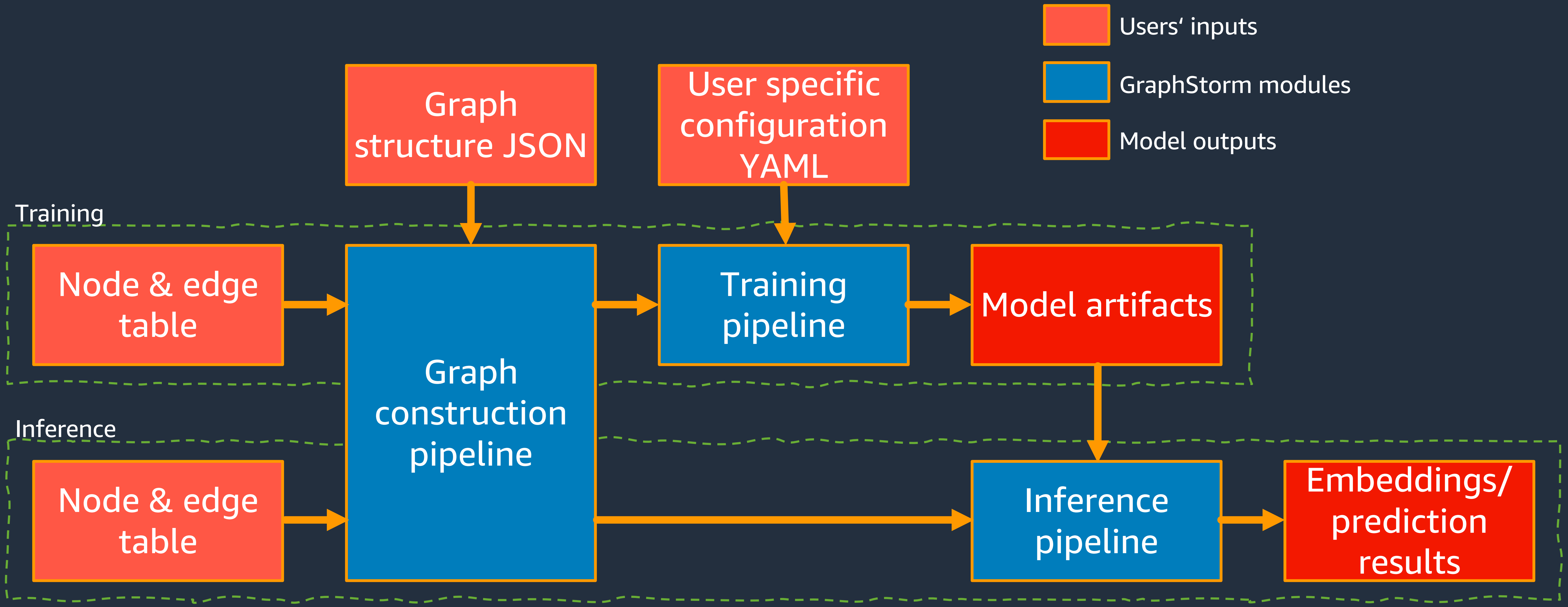


Comprehensive: many functionalities out-of-the-box

Application Layer	Fraud Detection, Recommendation, …, Time Series Prediction
Integration Layer	GraphML Platform, SageMaker Deployment Solution
Task Layer	Node Classification/Regression, Edge Classification/Regression, Link Prediction
Pipeline Layer	Graph construct (Single Instance); Graph Models RGCN, RGAT; Graph construct (Distributed); Graph Models Sage, HGT; Training pipeline; Model evaluator; Inference pipeline
Data Layer	Hetero & Homo Graphs Node & Edge Feature (numerical, categorical, text); Dynamic & Temporal Graphs
Framework Layer	PyTorch , DGL , …, HuggingFace
Infrastructure Layer	Single CPU/GPU, Multi-CPU/GPU, Multi-machine & multi-GPU

GraphStorm

Easy-to-use: ready-to-use pipelines



15min Break

GraphStorm hands-on

Hands-on Environment

Each one will have a temporary account to create an AWS EC2 instance via:

https://catalog.us-east-1.prod.workshops.aws/join?access-code=e94d-04a4a4-d3

Q&A