Enhancing Recommendations on Uber Eats with Graph Convolutional Networks

Ankit Jain/Piero Molino

ankit.jain/piero@uber.com

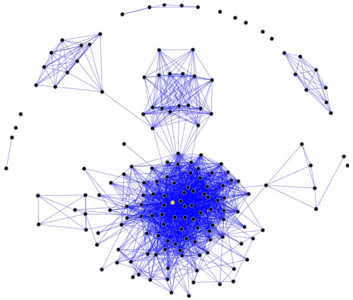
Uber AI

Agenda

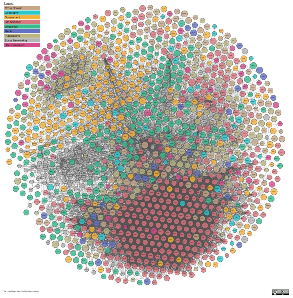
1. **Graph Representation Learning**
2. **Dish Recommendation on Uber Eats**
3. **Graph Learning on Uber Eats**

Graph Representation Learning

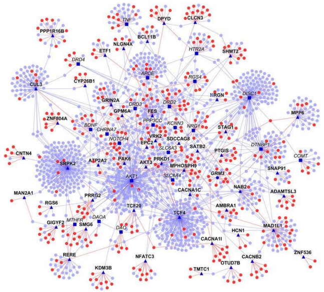
Graph data



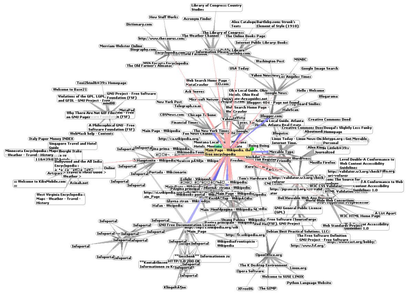
Social networks



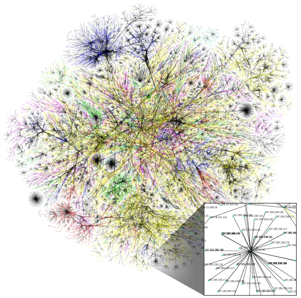
Linked Open Data



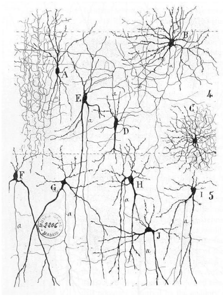
Biomedical networks



Information networks



Internet



Networks of neurons

Tasks on graphs

Node classification

Predict a type of a given node

Link prediction

Predict whether two nodes are linked

Community detection

Identify densely linked clusters of nodes

Network similarity

How similar are two (sub)networks

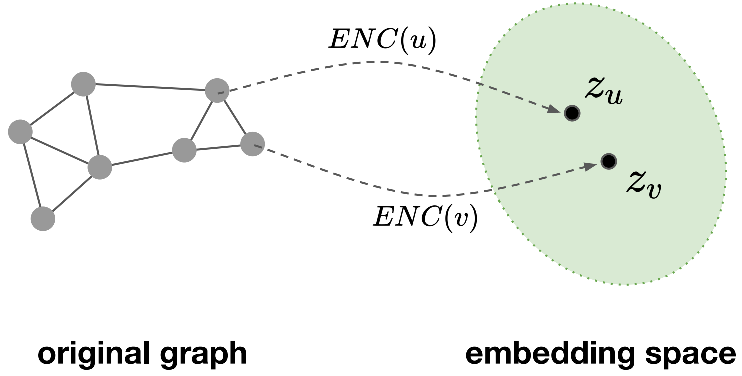
Learning framework

Define an encoder mapping from nodes to embeddings

Define a node similarity function based on the network structure

Optimize the parameters of the encoder so that:

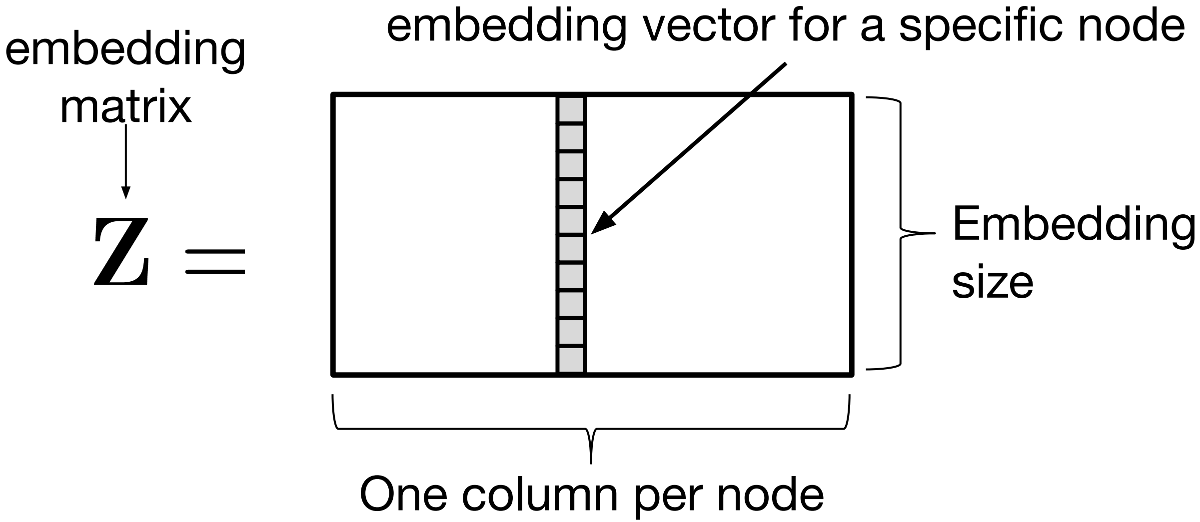
$$similarity(u, v) \approx z_v^\top z_u$$



Shallow encoding

Simplest encoding approach: encoder is just an embedding-lookup

Algorithms like Matrix Factorization, Node2Vec, Deepwalk fall in this category



Shallow encoding limitations

O(|*V*|) parameters are needed, every node has its own embedding vector

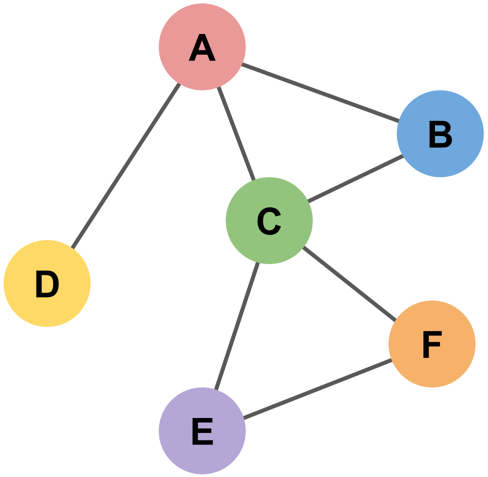
Either not possible or very time consuming to generate embeddings for nodes **not seen during training**

Does not incorporate **node features**

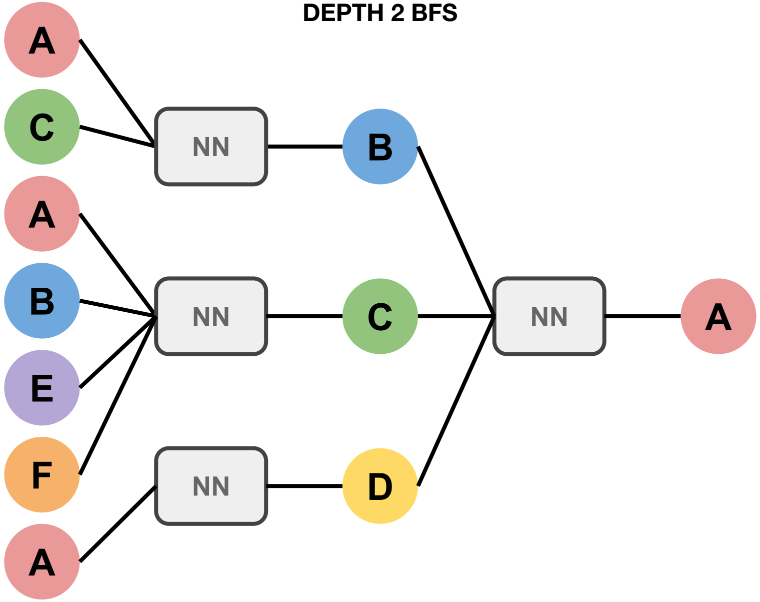
Graph Neural Network

Key Idea: To obtain node representations, use a neural network to aggregate information from neighbors recursively by limited Breadth-FIrst Search (BFS)

INPUT GRAPH



DEPTH 2 BFS



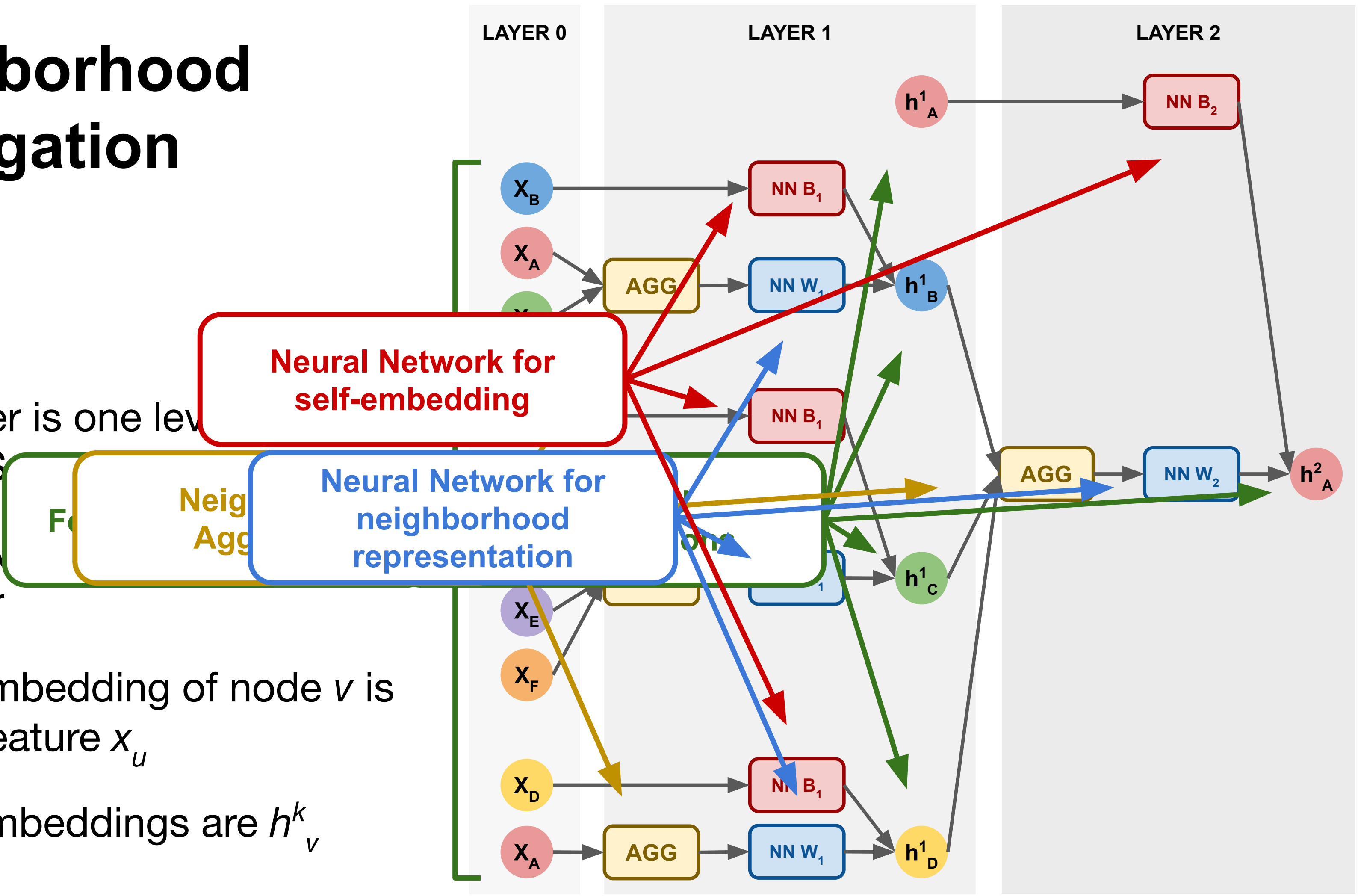
Neighborhood aggregation

Each Layer is one lev
in the BFS

Nodes hav
each layer

Layer 0 embedding of node v is its input feature x_u

Layer k embeddings are h^k_v

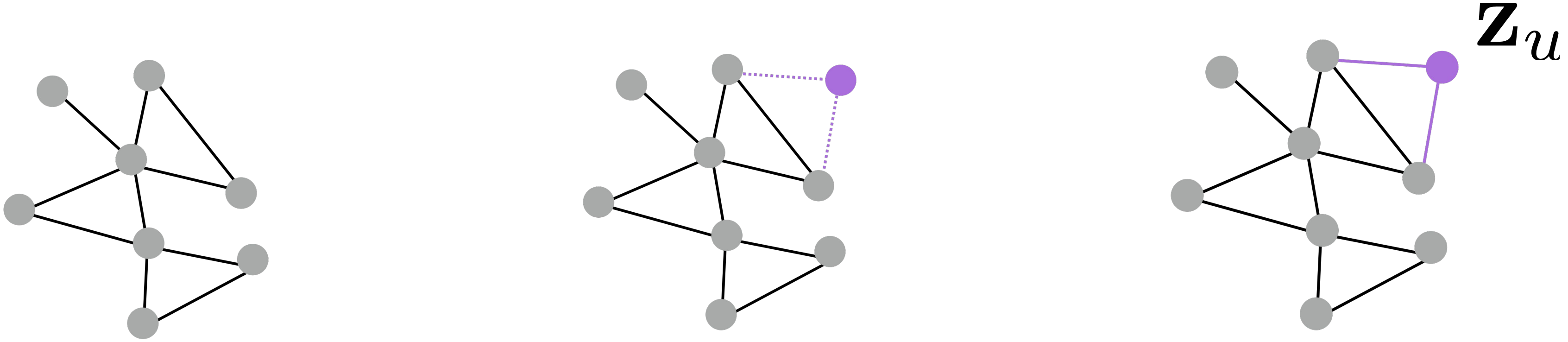


Inductive capability

In many real applications new nodes are often added to the graph

Need to generate embeddings for new nodes without retraining

Hard to do with shallow methods

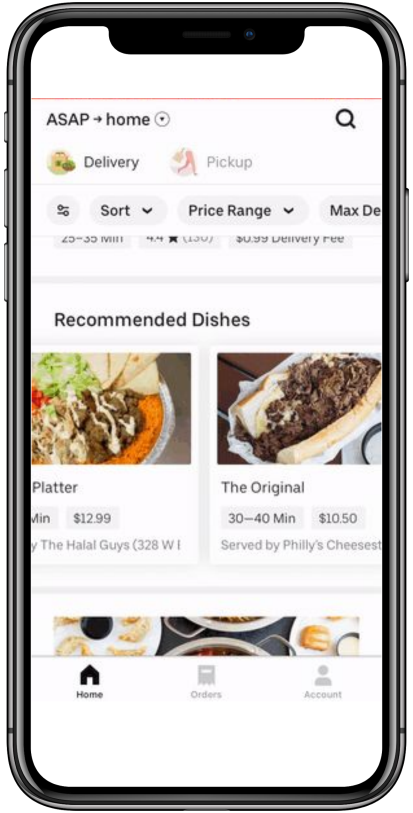


train with snapshot **new node arrives** **generate embedding for new node**

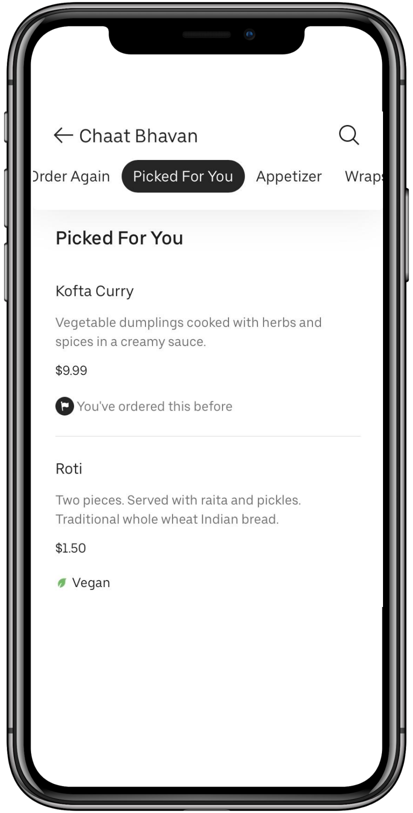
Dish Recommendation on Uber Eats

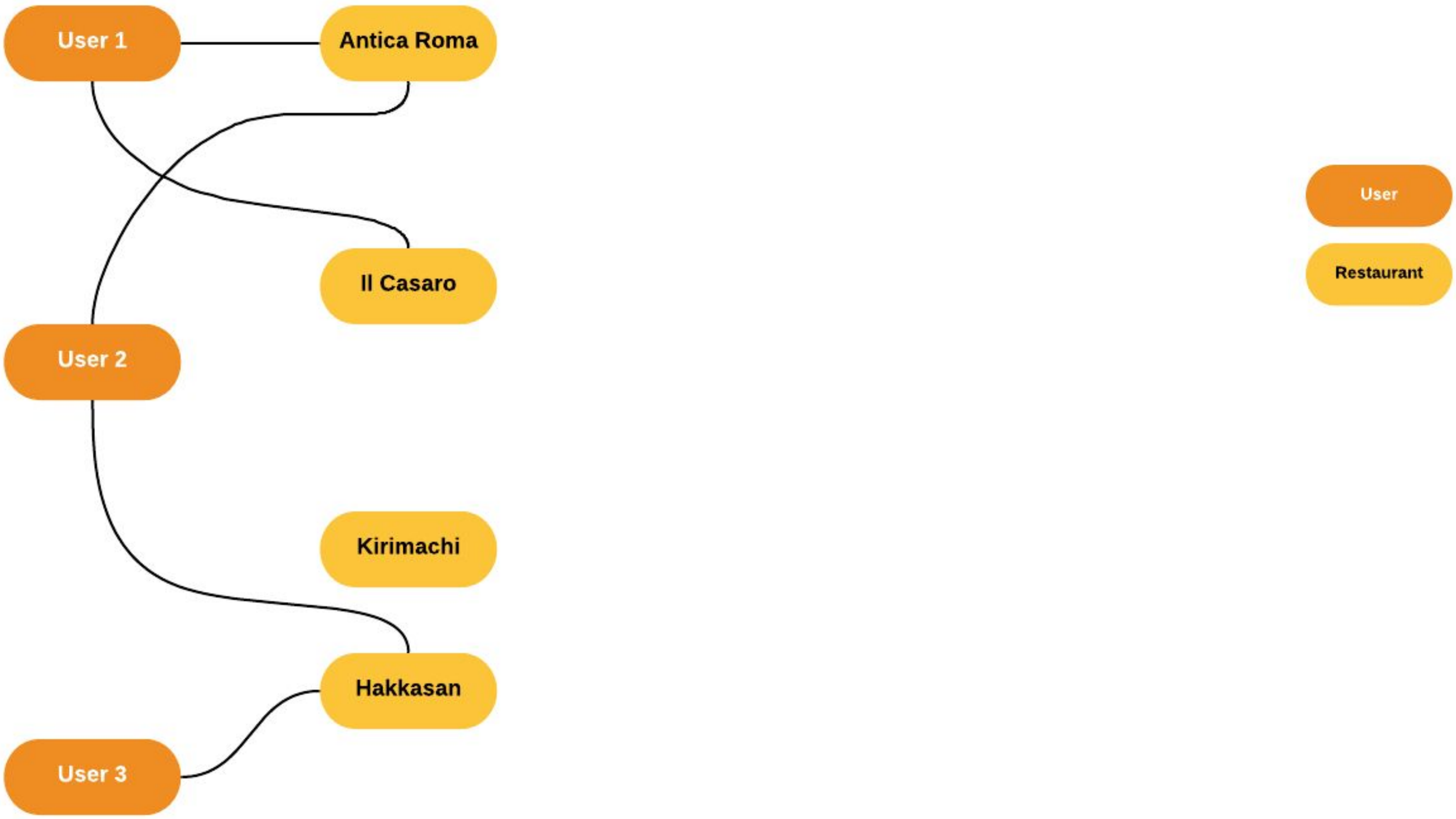
Suggested Dishes

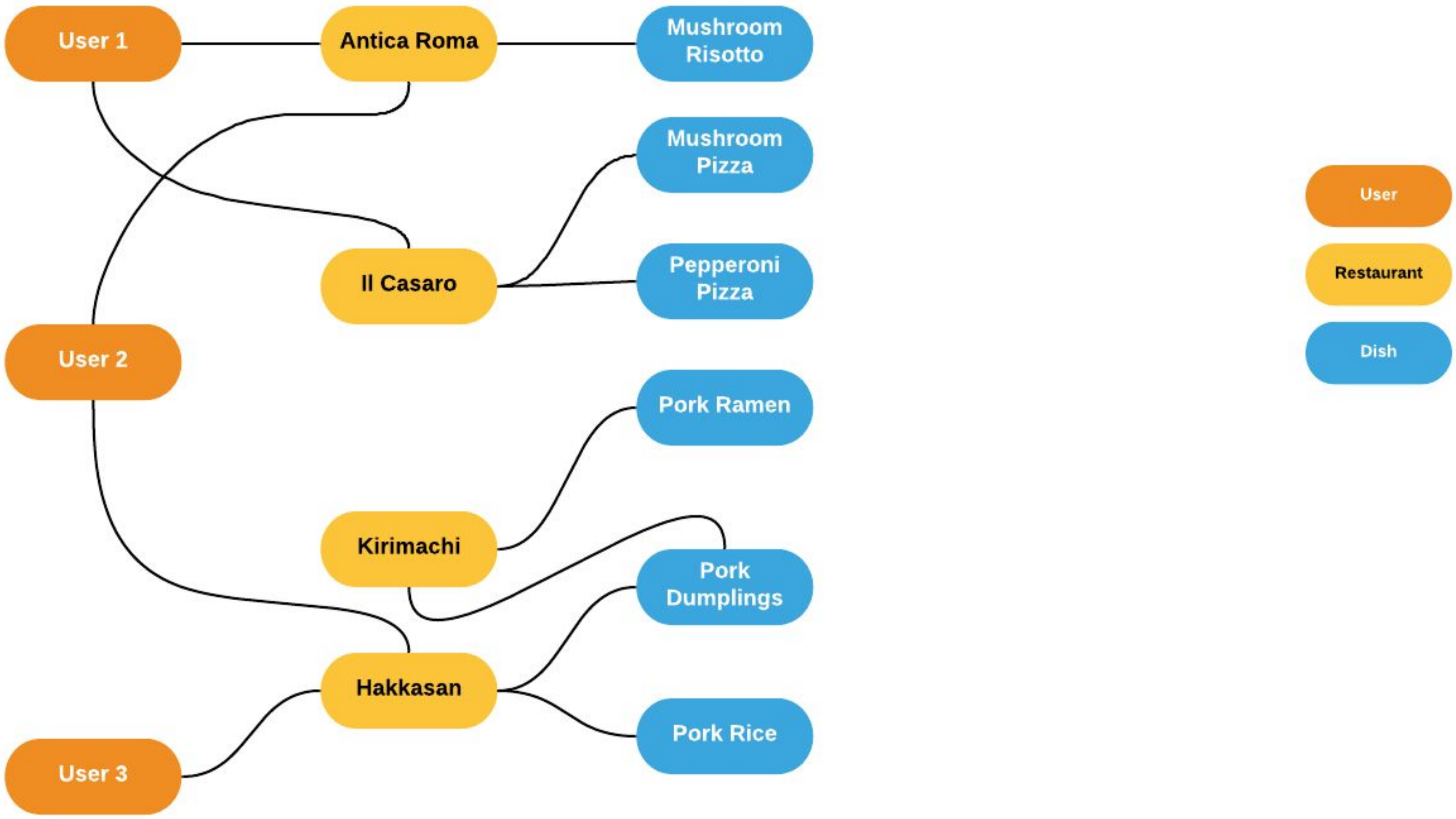
Recommended Dishes Carousel

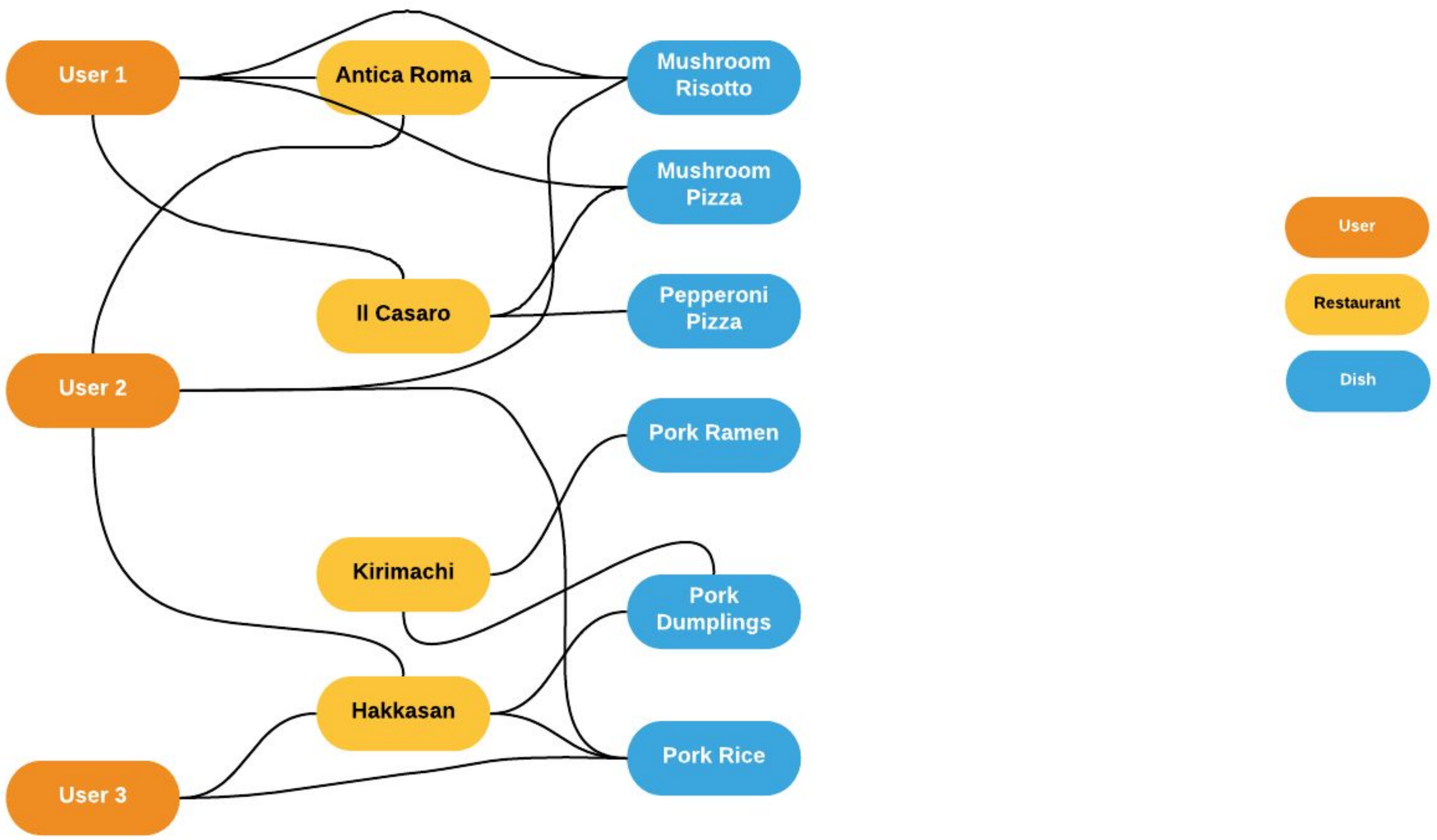


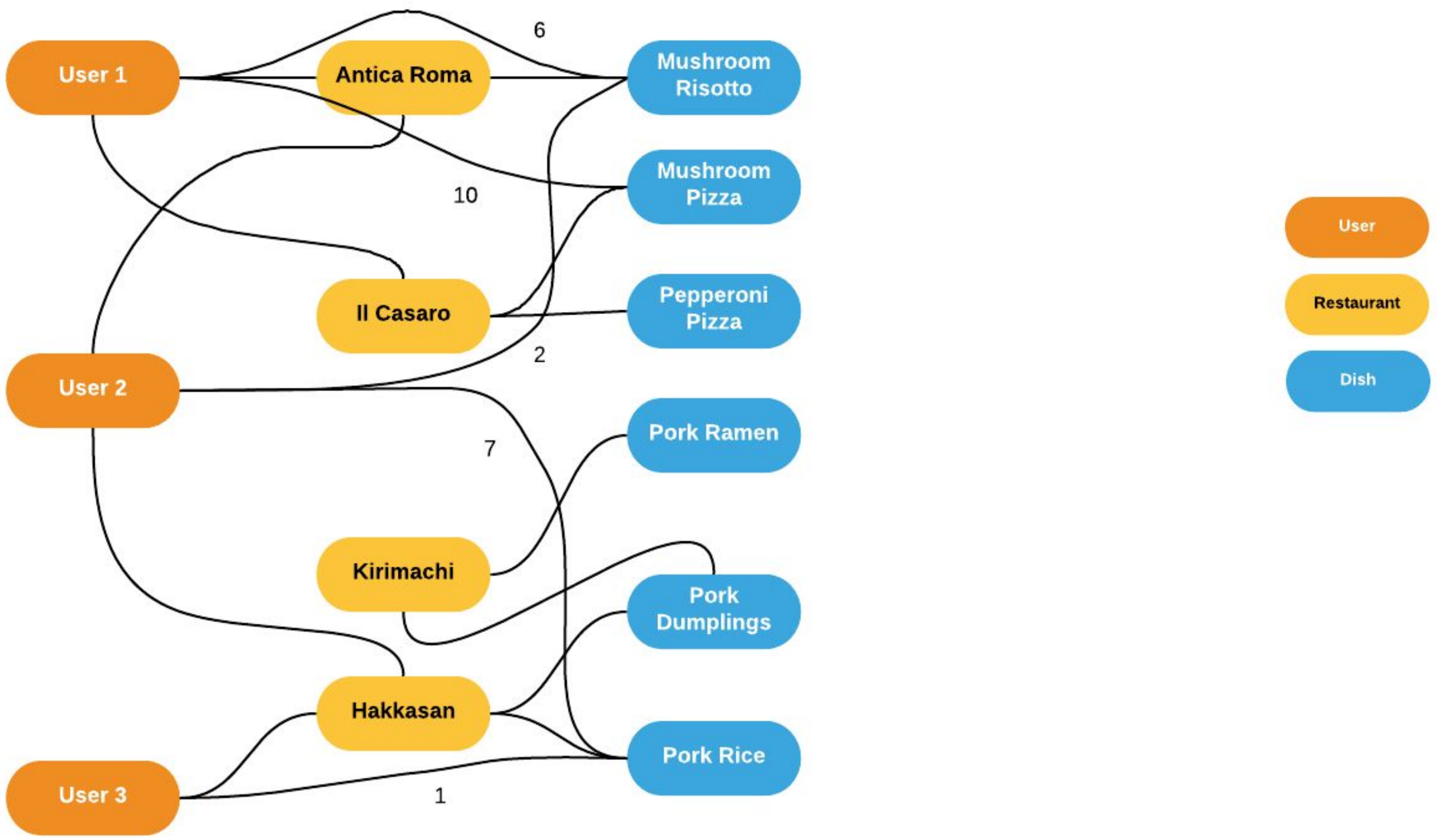
Picked for You




User 1
Antica Roma
Il Casaro
User 2
Kirimachi
Hakkasan
User 3
User
Restaurant


User 1
Antica Roma
Mushroom Risotto
Mushroom Pizza
Il Casaro
Pepperoni Pizza
User 2
Pork Ramen
Kirimachi
Pork Dumplings
Hakkasan
Pork Rice
User 3
User
Restaurant
Dish


User 1
Antica Roma
Mushroom Risotto
Mushroom Pizza
Il Casaro
Pepperoni Pizza
User 2
Pork Ramen
Kirimachi
Pork Dumplings
Hakkasan
Pork Rice
User 3
User
Restaurant
Dish


User 1
Antica Roma
6
Mushroom Risotto
Mushroom Pizza
10
Il Casaro
Pepperoni Pizza
2
User 2
7
Pork Ramen
Kirimachi
Pork Dumplings
Hakkasan
User 3
1
Pork Rice
User
Restaurant
Dish

Graph Learning in Uber Eats

Bipartite graph for dish recommendation

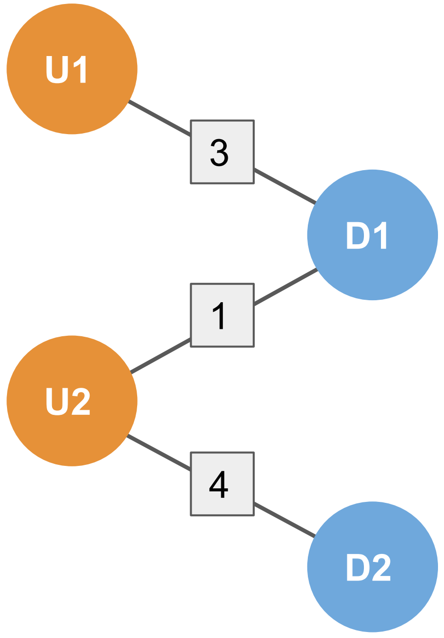
Users connected to dishes they have ordered in the last M days

Weights are frequency of orders

Graph properties

Graph is dynamic: new users and dishes are added every day

Each node has features, e.g. word2vec of dish names



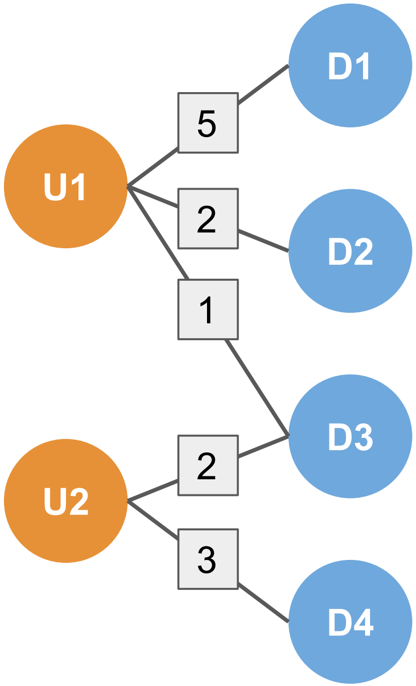
Max Margin Loss

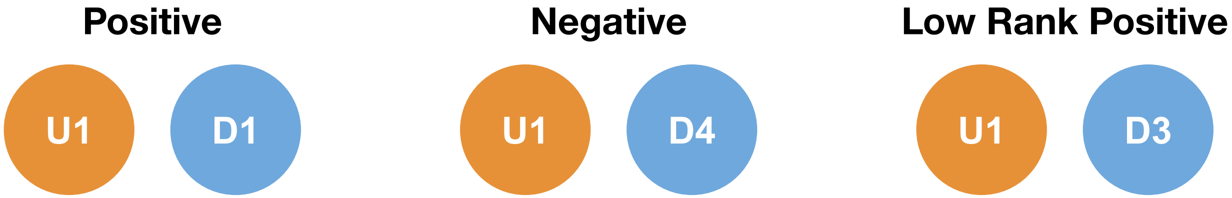
For dish recommendation we care about **ranking**, not actual similarity score

Max Margin Loss:

$$L = \sum_{(u,v)\in E} \max(0, -\underbrace{z_u z_v}_{\text{positive pair}} + \underbrace{z_u z_n}_{\text{negative sample}} + \underbrace{\Delta}_{\text{margin}})$$

New loss with Low Rank Positives



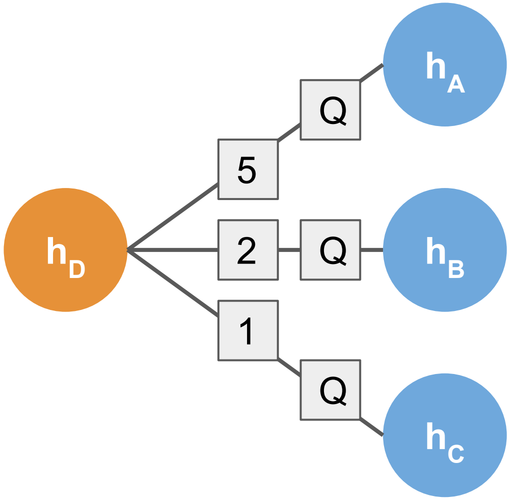


$$L = \sum_{(u,v)\in E} \alpha_n \max(0, -z_u z_v + z_u z_n + \Delta_n) + \alpha_l \max(0, -z_u z_v + z_u z_l + \Delta_l)$$

$$\Delta_l < \Delta_n$$

Weighted pool aggregation

Aggregate neighborhood embeddings based on edge weight



$$\mathbf{AGG} = \sum_{u \in N(v)} w(u,v) Q h_u^{k-1}$$

Q denotes a fully connected layer

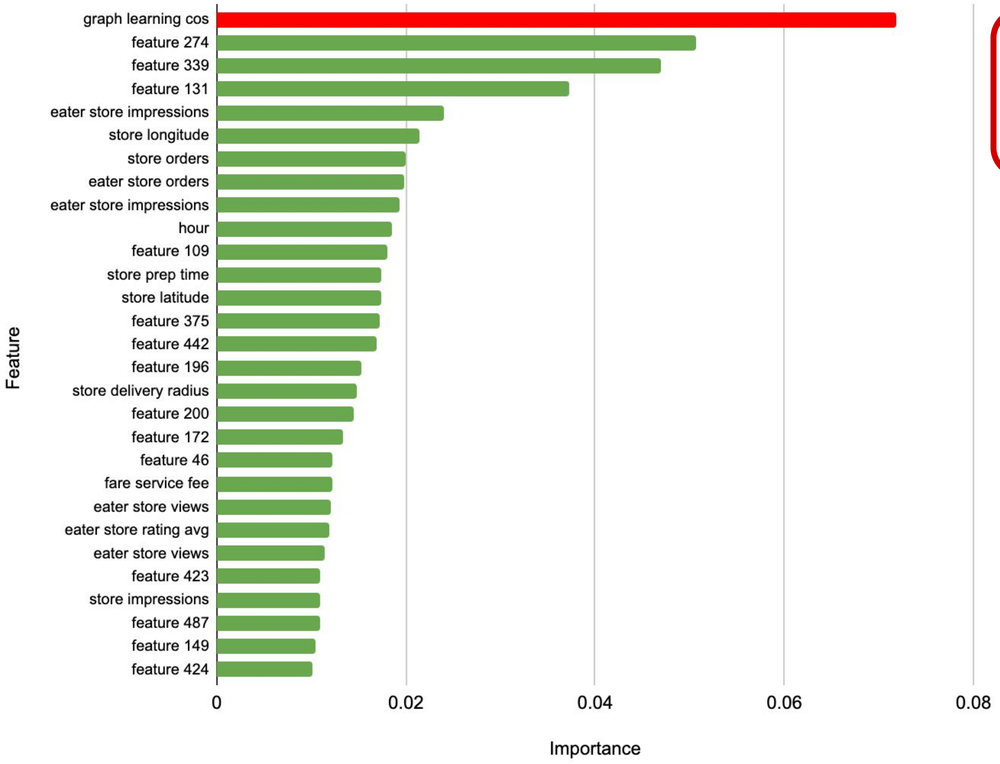
Offline evaluation

Trained the downstream Personalized Ranking Model using graph node embeddings

~12% improvement in test AUC over previous production model

Model	Test AUC
Previous production model	0.784
With graph embeddings	**0.877**

Feature Importance



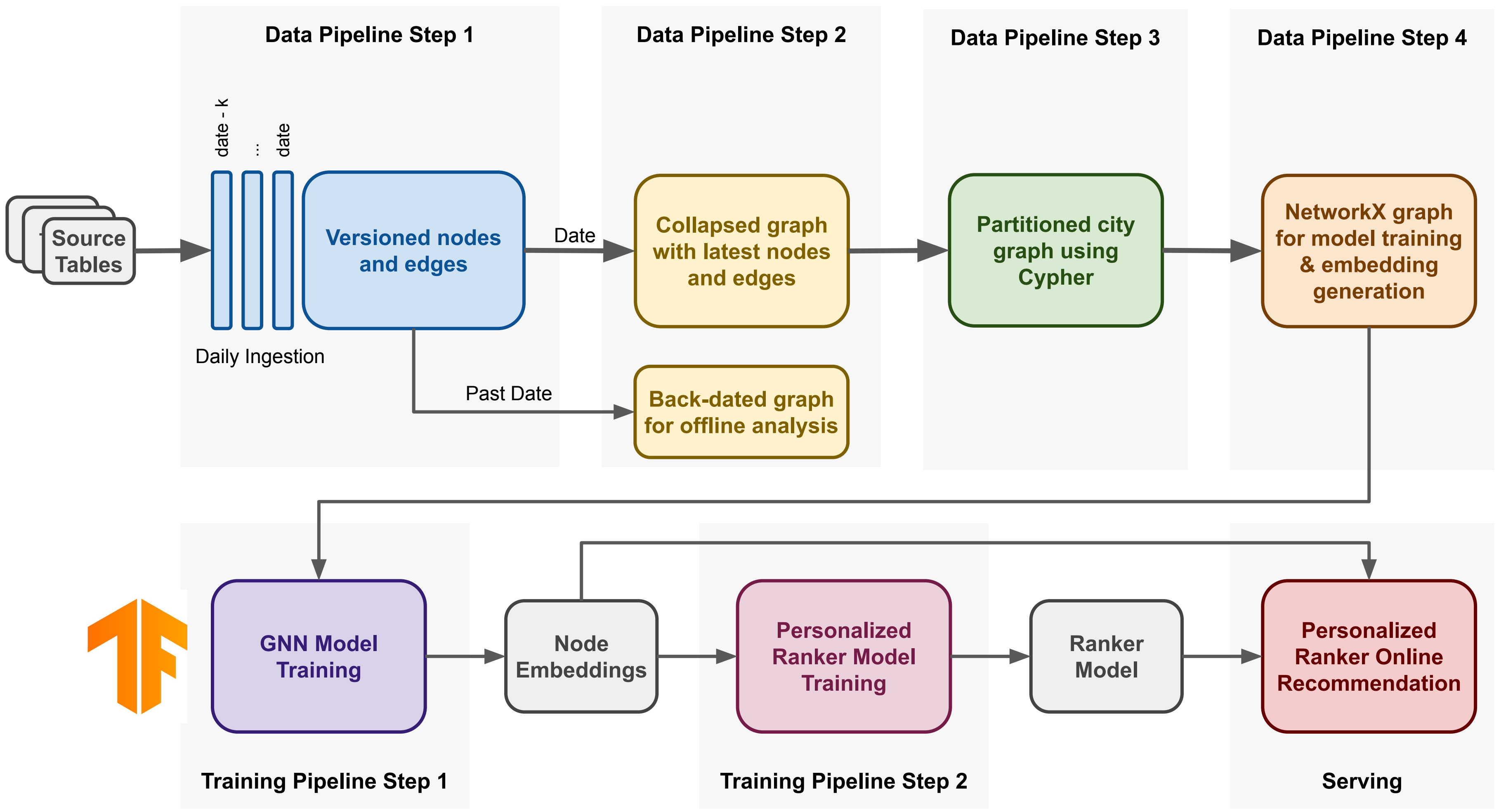
Graph learning cosine similarity is the top feature in the model

Online evaluation

Ran a A/B test of the Recommended Dishes Carousel in San Francisco

Significant uplift in Click-Through Rate with respect to the previous production model

Conclusion: Dish Recommendations with graph learning features are live in San Francisco, soon everywhere else


Data Pipeline Step 1
Data Pipeline Step 2
Data Pipeline Step 3
Data Pipeline Step 4
Source Tables
date - k
...
date
Versioned nodes and edges
Daily Ingestion
Date
Past Date
Collapsed graph with latest nodes and edges
Back-dated graph for offline analysis
Partitioned city graph using Cypher
NetworkX graph for model training & embedding generation
GNN Model Training
Node Embeddings
Personalized Ranker Model Training
Ranker Model
Personalized Ranker Online Recommendation
Training Pipeline Step 1
Training Pipeline Step 2
Serving

More Resources

Uber Eng Blog Post

Learn better representation in data scarcity regimes like small/new cities through meta-learning [NeurIPS Graph Representation Learning Workshop 2019]

Team

Ankit Jain

Piero Molino

Jan Pedersen

Ramit Hora

Isaac Liu

Long Tao

Nathan Berrebbi

Alex Danilychev

Ankur Sarda

Jimin Jia

Santosh Golecha

Thank you for your attention

Enhance Recommendations in Uber Eats with Graph Convolutional Networks

Ankit Jain/Piero Molino

ankit.jain/piero@uber.com

Uber AI

Uber

References

[1] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L. Hamilton and Jure Leskovec: Graph Convolutional Neural Networks for Web-Scale Recommender Systems, KDD 2018

[2] Bryan Perozzi, Rami Al-Rfou' and Steven Skiena: DeepWalk: online learning of social representations. KDD 2014

[3]Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan and Qiaozhu Mei: LINE: Large-scale Information Network Embedding. WWW 2015

[4] Aditya Grover, Jure Leskovec: node2vec: Scalable Feature Learning for Networks. KDD 2016

[5] Marco Gori, Gabriele Monfardini and Franco Scarselli: A new model for learning in graph domains. IJCNN 2005

[6] William L. Hamilton, Rex Ying and Jure Leskovec: Inductive Representation Learning on Large Graphs. NIPS 2017

[7] Joey Bose, Ankit Jain, Piero Molino and Will Hamilton: Meta-Graph: Few shot Link Prediction via Meta-Learning. Graph Representation Learning Workshop @ NeurIPS 2019