CS 744: DISTRIBUTED DGL

Shivaram Venkataraman

Fall 2022
- Midterm grades out!
- Regrade requests (check question numbers)
  - Thu: After class, Roger’s OH
  - Mon: Shivaram’s OH, Roger’s OH
  - Tue: After class

- Course Project: Check in by Nov 23rd
  -> Canvas / Piazza today
EXAMPLE: LINK PREDICTION

Task: Predict potential connections in a social network

Find K-nearest neighbors

[0.25, 0.45, 0.30]

[0.15, 0.85, 0.92]

...
EXAMPLE: NODE CLASSIFICATION

Task: Classify papers in a citation graph by subject area

From: https://github.com/lanstonchu/citation-graph
Loss function

Maximize score for edges in graph
Minimize for others (negative edges)

\[ 
\mathcal{L} = \sum_{e \in G} \sum_{e' \in S_e} \max(f(e) - f(e') + \lambda, 0) 
\]
Graph Neural Networks: Use neural network to capture neighborhood structure.

Input: \( h_i^0 \) (base node representations)

Model: \( h_i^k = AGG(h_i^{k-1}, \{h_u^{k-1} : u \in N_i\}) \)

\( N_i \) one-hop neighborhood of \( i \)

\( AGG \) parameterized aggregation func
**BACKGROUND: GRAPH SAGE**

weights need to be learned

$$h_v^k \leftarrow \sigma(W \cdot \text{MEAN}(\{h_v^{k-1}\} \cup \{h_u^{k-1}, \forall u \in N(v)\}))$$

Gather $h_v^k$ for all $u \in N(v)$

Mean of all nbr embeddings and $h_v^0$

Weight matrix in fully connected layer

Learn weights and embeddings

Example of CNN

only learn weights

$0 \leq h^0_I \leq 7$
DISTDGL: DEEP GRAPH LIBRARY

Distributed system for training GNNs
- KVStore
- Mini-batch sampler
- Trainer

System Design
Key techniques
- Partitioning heterogeneous graphs
- Async mini-batch sampling
**DISTDGL SYSTEM SETUP**

1. **Partition embeddings**
2. **Sampler & Trainer**

**Sampler:**
- Given the graph, generate samples
- List of vertices

**Interact:**
- With each other

**Machine-0**
- Trainer Process
- Sampler Process

**Machine-1**
- Trainer Process
- Sampler Process

**Shared-memory**

**RPC Request**
Hierarchical METIS

Apply METIS to partition graph across machines

Re-apply METIS to partition within a machine such that it fits in GPU memory

A fast and high quality multilevel scheme for partitioning irregular graphs

George Karypis and Vipin Kumar
HETEROGENEOUS GRAPH PARTITIONING

Knowledge graph includes all edge types

author -> affiliated with

cite -> write

adjacency matrix

ghost vertices

METIS
GNN MINI-BATCH PREPARATION

for batch in training_examples:
    sample_neighbors(batch)
    load_representations(batch)
    transfer_to_GPU(batch)
    loss = model(batch)
    transfer_to_CPU(batch)
    update_parameters(batch)

figure out which representations to load

mini-batch preparation
MINIBATCH SAMPLING FOR GNNS

\( b = 8 \implies \text{start with 8 target vertices} \)

up to

3 nbrs for each vertex ≤ 24 vertices

I remove common nbrs before expanding nbr hood

Access all of these for training
Async mini-batch sampling with sync training

Create nbrhood and feature for future mini batches

Sample layers with affecting sync training
Graph NN: capture the structure of graphs in creating embeddings

DistDGL: Distributed GNN training
- Partition graphs using METIS, hierarchical
- Pipelining to use CPU, GPU for sampling
DISCUSSION

https://forms.gle/Dp8qtqdpWuoVeys67
Speedup is sub-linear → CPU saturation of mini-batches

**Graph Attention** → more compute intensive → easier to scale

**Speedup**: GraphSage vs. GAT

- Log scale

**Speedup**: RGCN

- Seems to be linear
If you wished to extend the design of Marius to support GNNs, how would you do that? What would be some challenges?

**Encoder**
- emb table
- emb
- Agg:
  - GAT | Conv.
- vertex emb

**Decoder**
- src
- dist
- emb
- loss function
- link prediction
- decoder only

**Encoder + Decoder models**

If Marius is distributed → BETA ordering
Avoid Staleness → Change Pipelining

Need to sample
- vbrs of given vertex
- sample vbrs which are in cache
- METIS or smarter partitioning
Next class: Serverless computing
Project check-ins by Nov 23th