Course Project: Check in by Nov 20th

Instructions for this by tomorrow.

Regrades - a few?
Scalable Storage Systems

Datacenter Architecture

Resource Management

Computational Engines

Applications

Machine Learning

SQL

Streaming

Graph

Graph X

Powergraph

Machine Learning

SQL

Streaming

Graph

Graph X

Analytics

Graphs + ML
GRAPH EMBEDDINGS

Knowledge graph → different kinds of edges → multiple edges

Social graph = friend relation
• on an edge
• can learn 100 dim embedding

Knowledge graph

→ prediction
Friend recommendation

 LINK

embedding

PBC

embedding

2d embedding
• very close
• very close

(≈ 100 dim is common)
**TRAINING GRAPH EMBEDDINGS**

<table>
<thead>
<tr>
<th>Vertex</th>
<th>Embedding</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>&lt;0.0, 0.25&gt;</td>
</tr>
<tr>
<td>14</td>
<td>&lt;0.33, 0.5&gt;</td>
</tr>
<tr>
<td>16</td>
<td>&lt;0.45, 0.6&gt;</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Source</th>
<th>Dest</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>Id</td>
</tr>
<tr>
<td>14</td>
<td>Id</td>
</tr>
<tr>
<td>12</td>
<td>18</td>
</tr>
<tr>
<td>16</td>
<td>14</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

**Score function**

\[ f(\theta_s, \theta_r, \theta_d) = \text{sim} \left( g_s(\theta_s, \theta_r), g_d(\theta_d, \theta_r) \right) \]

**Similarity function**

\[ \text{sim}(\mathbf{a}, \mathbf{b}) = \mathbf{a} \cdot \mathbf{b} \]

**Example:**

\[ \text{sim}(12, 14) = \langle 0.0, 0.25 \rangle \cdot \langle 0.33, 0.5 \rangle = 0.075 \]

Score should be higher for vertices that are "more" connected or closer.
NEGATIVE SAMPLING

Sample from edges not in the graph!

→ Corrupt positive edges with a new src or dest

Two options
1. According to data distribution
   → Very rarely see negative edges between rare nodes

2. Uniformly
   → Model will tend to predict high degree vertices
   → Link prediction: always recommend very popular users

loss function

\[
\mathcal{L} = \sum_{e \in G} \sum_{e' \in S^e} \max(f(e) - f(e') + \lambda, 0)
\]

→ maximize score for true edges positive
→ minimize score for negative edges

Regularizer

Hybrid between rare nodes.
SCALING CHALLENGES

Fast enough to embed graphs with $10^{11} - 10^{12}$ edges in a reasonable time

~100 embedding parameters per node $\rightarrow$ require 800GB of memory!
GRAPH PARTITIONING

Nodes (destination entity types)

Nodes (source entity types)

Edges (3,1)
Edges (3,2)
Edges (3,3)
Edges (3,4)

Edges (2,1)
Edges (2,2)
Edges (2,3)
Edges (2,4)

Edges (1,2)
Edges (1,3)
Edges (1,4)

Edges (4,1)
Edges (4,2)
Edges (4,3)
Edges (4,4)

N vertices

To compute score, vertex embeddings of vertex (3) and vertex (1) only these two non-overlapping edge buckets be processed in parallel

Consider Edges (3,1) bucket

→ sample some edges from this bucket
→ To compute score, vertex embeddings of vertex (3) and vertex (1)

partitions of vertex embeddings

non-overlapping edge buckets be processed in parallel

edges per bucket may not be equal!
SYSTEM DESIGN

Assign partitions to workers

Parallel workers

all edges of graph

Assign partitions to workers

Parameter Client Thread

1. Request bucket

2. Swap partitions

3. Load Edges

N. Write Checkpoint

Shared Filesystem

Sharded Parameter Server

Sharded Partition Server

Lock Server

other relation embeddings on parameter server

vert ex

emb

~ HFS where each partition is on diff server
DISTRIBUTED EXECUTION

At a worker:

1. Get a edge bucket from lock server
2. Fetch vertex embedding partitions
3. Load positive edges from FS
   - Do TRAINING
   - Checkpoints to FS

- Load checkpoints to FS
BATCH NEGATIVE SAMPLING

-> Also need to fetch emb. for negative samples!

-> Only going to corrupt src/dest with vertices from same partitions

-> Re-use negative edges by batching them.

4 Positive Edges

Edges separated into two chunks

3 nodes sampled for each chunk

Each positive edge now has 3 corresponding negative edges

1

3

(3, 3)

Also need to fetch emb. for negative samples!

Thanks Jason!

Sample few get a lot of negatives
Graph Embeddings: Learn embeddings from graph data for ML

Partition graph into buckets for scalability
Distributed execution with shared partition server
Batched negative sampling
DISCUSSION

https://forms.gle/YEtECXCUtksoL6Sr8
Accuracy degrades for 16 partitions.

Running for a fixed epochs may not be ideal?

Time per epoch much lower with more machines.
How does the partitioning scheme used in this paper differ from partitioning used in PowerGraph and why? (from review)

Power Graph
- update vertex/edge state
  - here we update embeddings!
→ Power graph: not all vertices might get updated
  - order of traversal didn't matter (barrier)
→ Power graph
  - vertex had to look at all the neighbors
PBG → edge only look at emb. for src, dst
Next class: New module!
Project check-ins by Nov 20th