Course Project: Check in by Nov 20th
## 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>14</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>

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

Sample from edges not in the graph!

Two options
1. According to data distribution
\[ \mathcal{L} = \sum_{e \in G} \sum_{e' \in S'_e} \max(f(e) - f(e') + \lambda, 0) \]
2. Uniformly
SCALING CHALLENGES

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

~100 embedding parameters per node → require 800GB of memory!
## Graph Partitioning

<table>
<thead>
<tr>
<th>Nodes (destination entity types)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Edges (1,1)</td>
</tr>
<tr>
<td>Edges (1,2)</td>
</tr>
<tr>
<td>Edges (1,3)</td>
</tr>
<tr>
<td>Edges (1,4)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Nodes (source entity types)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Edges (2,1)</td>
</tr>
<tr>
<td>Edges (2,2)</td>
</tr>
<tr>
<td>Edges (2,3)</td>
</tr>
<tr>
<td>Edges (2,4)</td>
</tr>
</tbody>
</table>

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

| Edges (4,1)                     |
| Edges (4,2)                     |
| Edges (4,3)                     |
| Edges (4,4)                     |
SYSTEM DESIGN

Shared Filesystem

Rank 1

Trainer
Thread 1
Thread 2
Thread 3

Parameter Client Thread

Lock Server

1. Request bucket

Sharded Parameter Server

2. Swap partitions

Async

Rank 2

Trainer
Thread 1
Thread 2
Thread 3

Parameter Client Thread

N. Write Checkpoint

3. Load Edges

Rank 3

Trainer
Thread 1
Thread 2
Thread 3

Parameter Client Thread

Sharded Partition Server

...
DISTRIBUTED EXECUTION

- Rank 1:
  - Trainer
  - Thread 1
  - Thread 2
  - Thread 3
  - Parameter Client Thread
  - Lock Server

- Rank 2:
  - Trainer
  - Thread 1
  - Thread 2
  - Thread 3
  - Parameter Client Thread

- Rank 3:
  - Trainer
  - Thread 1
  - Thread 2
  - Thread 3
  - Parameter Client Thread

- Shared Filesystem
- Shared Parameter Server
- Sharded Parameter Server
BATCH NEGATIVE SAMPLING

4 Positive Edges

Edges separated into two chunks

3 nodes sampled for each chunk

Each positive edge now has 3 corresponding negative edges
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/YEtECXCUtks0L6Sr8
How does the partitioning scheme used in this paper differ from partitioning used in PowerGraph and why? (from review)
NEXT STEPS

Next class: New module!
Project check-ins by Nov 20th