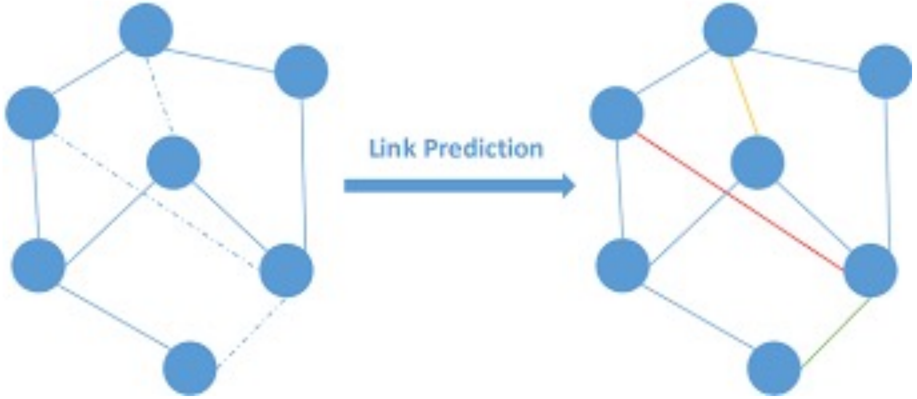
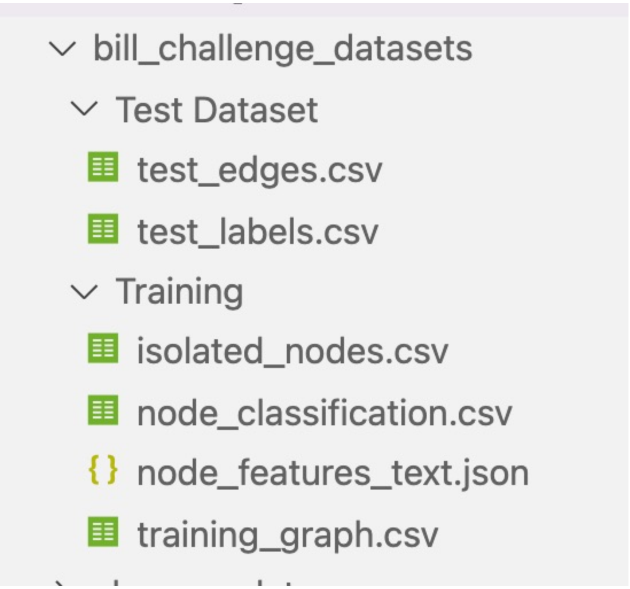


Mini-GCN for Link Prediction

Peikun Guo, Computer Science

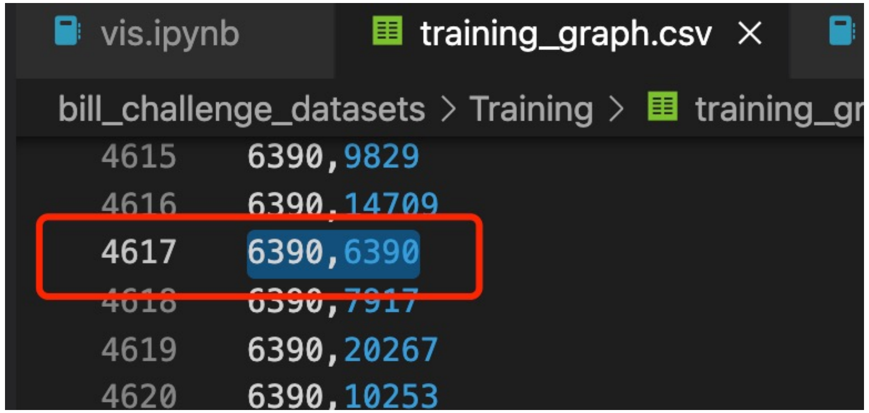
(Bill.com Track)

Exploratory Data Analysis



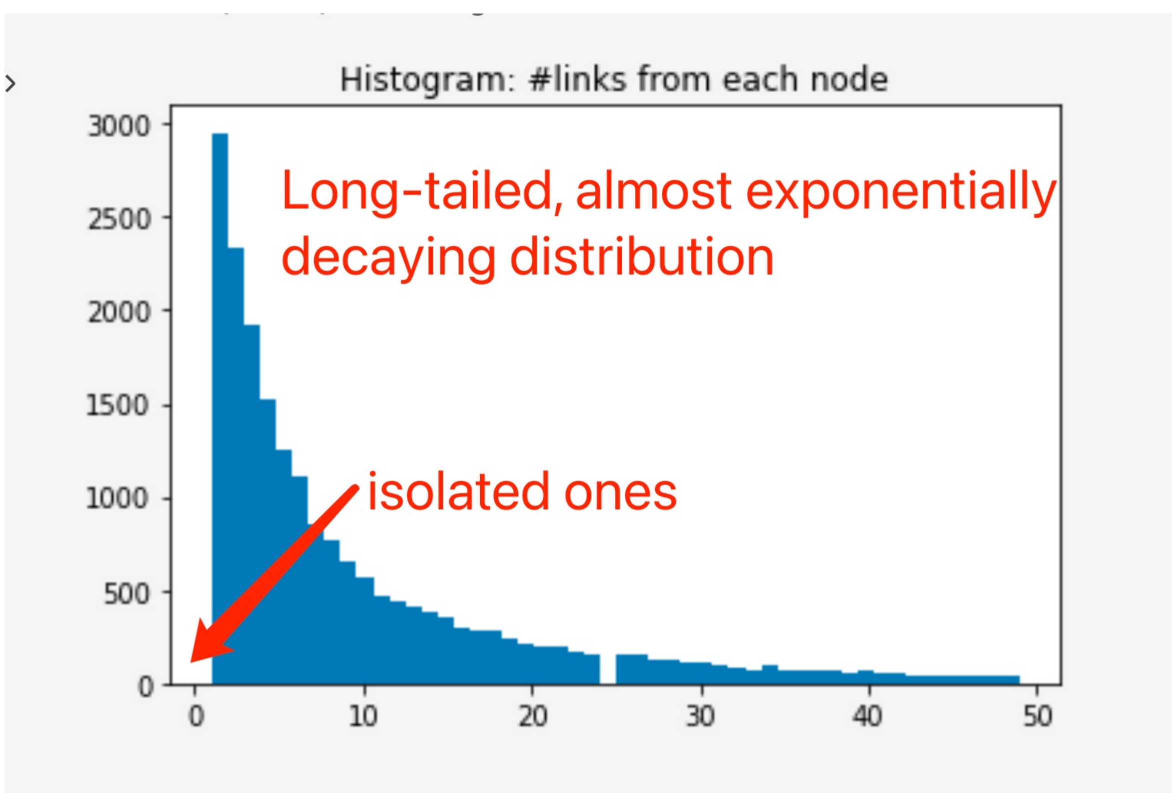
```
bill_challenge_datasets
  Test Dataset
    test_edges.csv
    test_labels.csv
  Training
    isolated_nodes.csv
    node_classification.csv
    node_features_text.json
    training_graph.csv
```

- Nodes: webpages
  - (22470 linked, 1655 isolated)
- Edges: exist if two pages are linked(132039)
- Page's text description (vector of one-hot indices, not text)
- Page type (label {1,2,3,4})



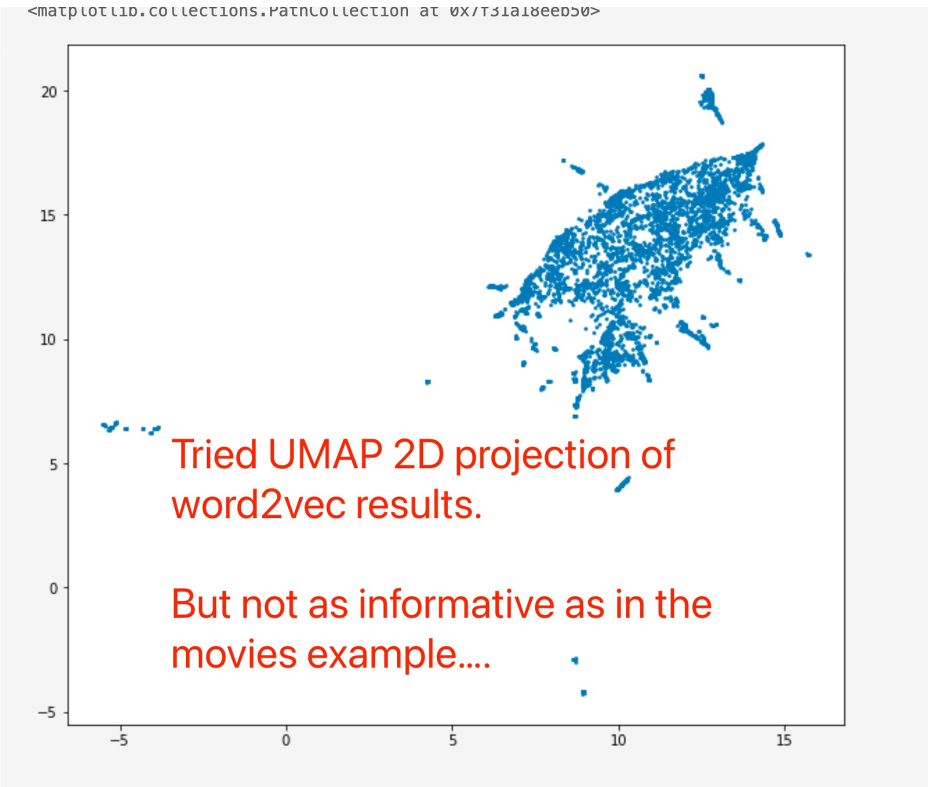
```
6390,9829
6390,14709
6390,6390
6390,7917
6390,20267
6390,10253
```

Exploratory Data Analysis



Long-tailed, almost exponentially decaying distribution

isolated ones



```
<matplotlib.collections.PathCollection at 0x7f31a18eeb50>
```

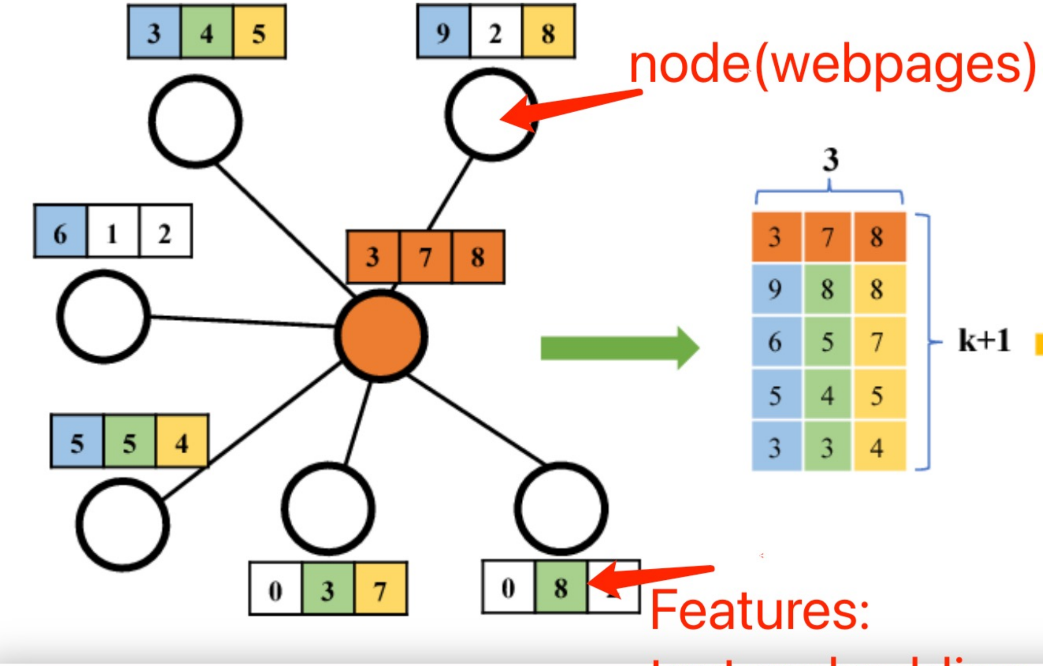
Tried UMAP 2D projection of word2vec results.

But not as informative as in the movies example….

- Some nodes have 500+ links, making them hard to be fit on the plot
- 90%+ of the nodes have <50 links. ➔SPARSE GRAPH!

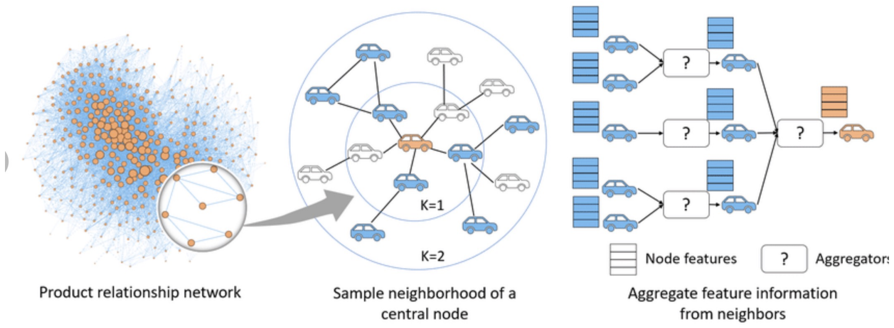
Pre-processing and Feature Engineering

- Node features
  - labels: provided, 4 types
    - Can be fed into graph package APIs like DGL and PyG
  - Embedding text one-hot vectors
  - Use Doc2Vec, decide the output feature dimension based on the raw sentence length
- **Graph**
  - Nodes: pages
  - Edges: connectivity of pages
  - Node feature: label + (embedded) text
  - GCN is built for the job



node(webpages)

Features: text embedding



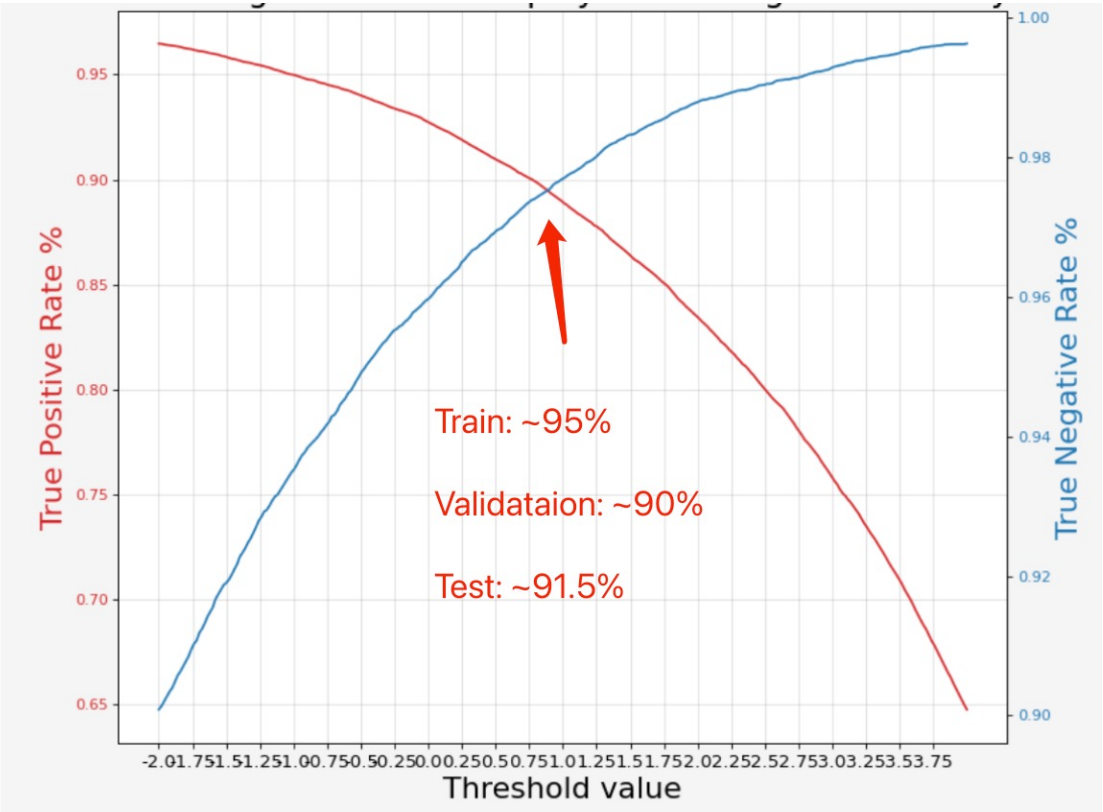
Product relationship network

Sample neighborhood of a central node

Aggregate feature information from neighbors

Results

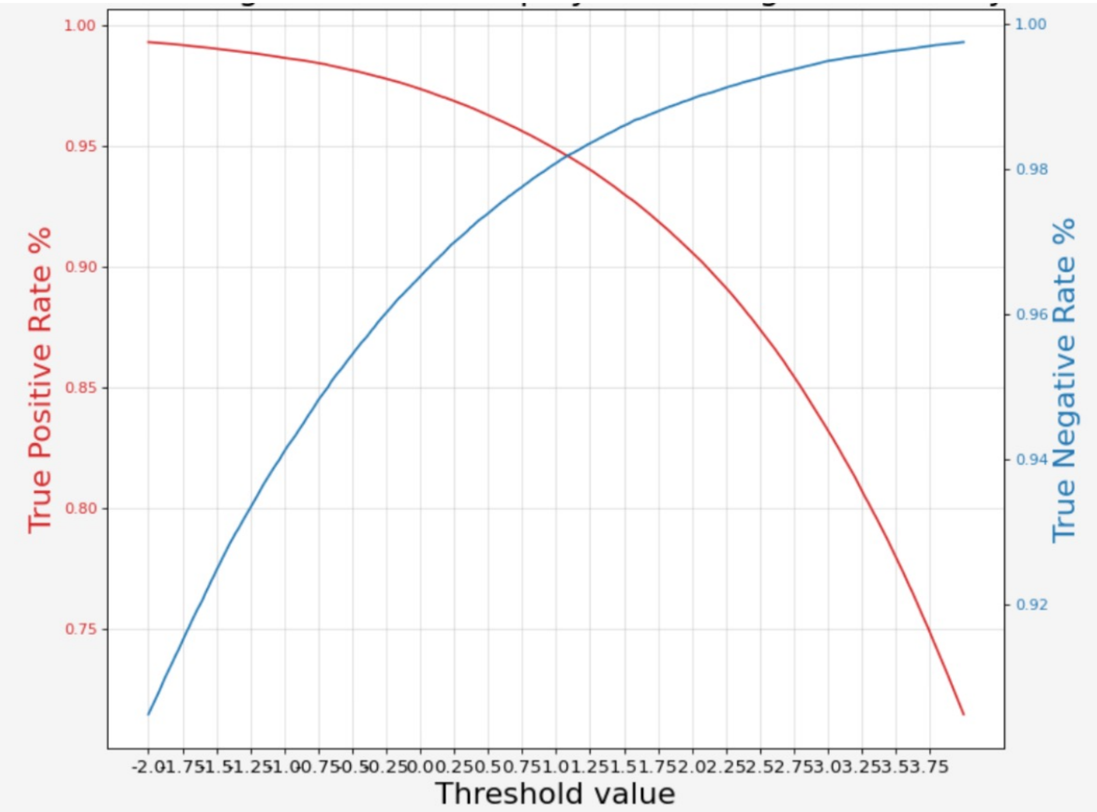
- 91.3% Classification Accuracy in test set edges



Train: ~95%

Validataion: ~90%

Test: ~91.5%



Future Directions

- **Problem Abstraction: Link Prediction in Graph**
- Small model — room for increasing complexity
  - Deeper GraphSAGE
  - GAE, HeteroGraphConv to be tried……
  - Expand the current model, e.g. higher number of channels
  - More complex text embedding, e.g. BERT

References

1. SEAL: https://towardsdatascience.com/seal-link-prediction-explained-6237919fe575
2. Graph construction: https://github.com/raunakkmr/GraphSAGE-and-GAT-for-link-prediction
3. VGAE: https://github.com/jiangnanboy/gnn4lp/
4. Link prediction: https://www.youtube.com/watch?v=EA4sK5t3wf8
5. DGL tutorial: https://docs.dgl.ai/en/0.6.x/new-tutorial/4_link_predict.html
6. A review of graph learning: https://leovan.me/cn/2020/04/graph-embedding-and-gnn/
7. Picture credits in the slides:
    1. https://www.google.com/url?sa=i&url=https%3A%2F%2Fwww.mdpi.com%2703
    2. https://www.researchgate.net/figure/Illustration-of-sampling-and-aggregation-in-GraphSAGE-method-A-sample-of-neighboring_fig1_351575091
    3. https://www.semanticscholar.org/paper/Large-Scale-Learnable-Graph-Convolutional-Networks-Gao-Wang/d5aefe86b1ba8c773a6bd0e84812ace161b8c0db