Graph Sampling for Link Prediction

CS 430

Algorithms for Data Science

Nilay Thakor

Ojas Joshi

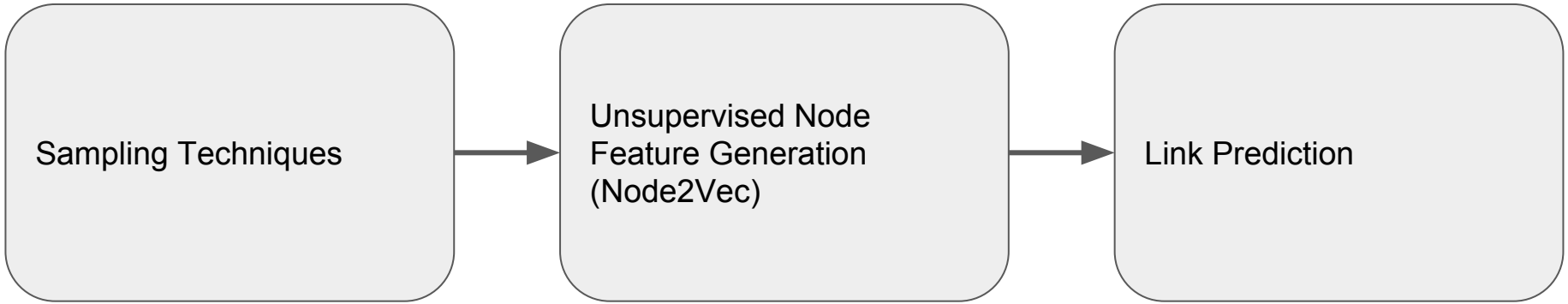
Motivation

- Node feature vector required for supervised machine learning tasks such as link prediction, multilabel classification.
- Node embedding techniques such as node2vec, deep walk requires entire graph to be in memory at the same time
- Use sampling to reduce the size of graph and see its impact on the embeddings and link prediction

Problem Statement

Analysis of Sampling techniques for link prediction using edge features generation.

Approach


Sampling Techniques
Unsupervised Node Feature Generation (Node2Vec)
Link Prediction

Sampling Techniques

- Graph Sample & Hold
  - p=0.001, q=0.1 -- 10% sampling
  - p=0.001, q=0.01 -- 1% sampling

- Graph Priority Sampling
  - Using number of triangles completed by incoming edge as weights
  - Using minimum degree of incoming node as weights

- Reservoir Sampling

Feature Generation

- Node2vec
  - Random Walk around each node to create neighborhood
  - Pass neighborhood, node pair to word2vec for embeddings

Algorithm 1 The *node2vec* algorithm.

LearnFeatures (Graph $G = (V, E, W)$, Dimensions d, Walks per node r, Walk length l, Context size k, Return p, In-out q)
  π = PreprocessModifiedWeights(G, p, q)
  $G' = (V, E, \pi)$
  Initialize $walks$ to Empty
  for $iter = 1$ **to** r **do**
    for all nodes $u \in V$ **do**
      $walk$ = node2vecWalk(G', u, l)
      Append $walk$ to $walks$
  f = StochasticGradientDescent(k, d, $walks$)
  return f

node2vecWalk (Graph $G' = (V, E, \pi)$, Start node u, Length l)
  Inititalize $walk$ to $[u]$
  for $walk\_iter = 1$ **to** l **do**
    $curr = walk[-1]$
    V_{curr} = GetNeighbors($curr, G'$)
    s = AliasSample(V_{curr}, π)
    Append s to $walk$
  return $walk$

Link Prediction

- Node feature vector:

Operator	Symbol	Definition
Average	$\boxplus$	$[f(u) \boxplus f(v)]_i = \frac{f_i(u)+f_i(v)}{2}$
Hadamard	$\boxdot$	$[f(u) \boxdot f(v)]_i = f_i(u) * f_i(v)$
Weighted-L1	$\|\cdot\|_{\bar{1}}$	$\|f(u) \cdot f(v)\|_{\bar{1}i} = \|f_i(u) - f_i(v)\|$
Weighted-L2	$\|\cdot\|_{\bar{2}}$	$\|f(u) \cdot f(v)\|_{\bar{2}i} = \|f_i(u) - f_i(v)\|^2$

- Classifier:
  - Support Vector Machine
    - Kernel : Radial Basis Function
    - Regularisation parameter: 1.0

Testing Method

- Separate 30% edges from original graph as test cases
- Sample remaining 70% edges and create node features
- Create positive and negative train cases from sampled graph
- Create create positive and negative test cases from test cases

Dataset Used

- SNAP ego-Facebook large networks dataset
  - Nodes: 4,039
  - Edges: 88,234
  - No. of triangles: 1,612,010

Experimental Results

Sampling Method	Sampling Percentage (%)	F1	Precision Score	Recall
Reservoir	1	0.44	0.70	0.52
GPS_triangles	1	0.54	0.75	0.55
GPS_nodes	1	0.46	0.70	0.52
SampleAndHold	1	0.55	0.72	0.55
Reservoir	10	0.88	0.89	0.88
GPS_triangles	10	0.79	0.80	0.79
GPS_nodes	10	0.87	0.89	0.86
SampleAndHold	10	0.86	0.84	0.87
Without Sampling	0	0.96	0.96	0.96

References

1. Grover, Aditya, and Jure Leskovec. "node2vec: Scalable feature learning for networks." Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, 2016.
2. Ahmed, Nesreen K., et al. "Graph sample and hold: A framework for big-graph analytics." Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2014.
3. Ahmed, Nesreen K., et al. "On Sampling from Massive Graph Streams." arXiv preprint arXiv:1703.02625 (2017).
4. Al Hasan, Mohammad, et al. "Link prediction using supervised learning." SDM06: workshop on link analysis, counter-terrorism and security. 2006.