node2vec: Scalable Feature Learning for Networks
Aditya Grover, Jure Leskovec

Farzaneh Heidari
Outline

- word2vec (Background)
- Random Walk (Background)
- node2vec
- Evaluation Results
- Deficiencies
End-to-End Graph Analytics

Data → Graph analytics → New knowledge and insights
Machine Learning Lifecycle

(Supervised) Machine Learning Lifecycle: This feature, that feature. Every single time!

- Raw Data ➔ Structured Data ➔ Learning Algorithm ➔ Model

- Feature Engineering ✗
- Automatically learn the features
- Downstream prediction task
Random Walk

node2vec

word2vec
word2vec

Input:
Cat fridays are the best
Cats in boxes
Cats on heads
In every small compartment
Entangled in threads

Output:

<table>
<thead>
<tr>
<th>Cats</th>
<th>0.434</th>
<th>0.239</th>
<th>0.123</th>
<th>0.934</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fridays</td>
<td>0.126</td>
<td>0.996</td>
<td>0.453</td>
<td>0.124</td>
</tr>
<tr>
<td>Boxes</td>
<td>0.924</td>
<td>0.534</td>
<td>0.195</td>
<td>0.845</td>
</tr>
</tbody>
</table>

...
word2vec’s backbone

\[ J(\theta) = \frac{1}{T} \sum_{t=1}^{T} \sum_{-m \leq j \leq m, j \neq 0} \log p(w_{t+j} | w_t) \]

\[ p(o|c) = \frac{\exp \left( u_o^T \nu_c \right)}{\sum_{w=1}^{W} \exp \left( u_w^T \nu_c \right)} \]
Window in Graph
Random Walk

Stochastic Process
Path of random steps
Feature Learning in Graphs

Goal: Learn features for a set of objects

Feature learning in graphs:
- Given: $G = (V, E)$
- Learn a function: $f : V \rightarrow \mathbb{R}^d$
  - Not task specific: Just given a graph, learn $f$. Can use the features for any downstream task!
Unsupervised Feature Learning

- **Intuition:** Find a mapping of nodes to d-dimensions that preserves some sort of node similarity

- **Idea:** Learn node embedding such that nearby nodes are close together

- **Given a node** $u$, how do we define nearby nodes?
  - $N_S(u)$ ... neighbourhood of $u$ obtained by sampling strategy $S$
How to determine $N_S(u)$

Two classic search strategies to define a neighborhood of a given node:

for $|N_S(u)| = 3$
BFS vs. DFS

BFS: Micro-view of neighbourhood

DFS: Macro-view of neighbourhood

Structural vs. Homophilic equivalence
BFS vs. DFS

Structural vs. Homophilic equivalence

BFS-based:
Structural equivalence
(structural roles)

DFS-based:
Homophily
(network communities)
Interpolating BFS and DFS

- Biased random walk procedure, that given a node $u$ samples $N_S(u)$

The walk just traversed $(t, v)$ and aims to make a next step.
## Multilabel Classification

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Dataset</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spectral Clustering</td>
<td>BlogCatalog</td>
</tr>
<tr>
<td></td>
<td>0.0405</td>
</tr>
<tr>
<td>DeepWalk</td>
<td>0.2110</td>
</tr>
<tr>
<td>LINE</td>
<td>0.0784</td>
</tr>
<tr>
<td><em>node2vec</em></td>
<td><strong>0.2581</strong></td>
</tr>
<tr>
<td><em>node2vec</em> settings (p,q)</td>
<td>0.25, 0.25</td>
</tr>
<tr>
<td>Gain of <em>node2vec</em> [%]</td>
<td><strong>22.3</strong></td>
</tr>
</tbody>
</table>

- Spectral embedding
- DeepWalk [B. Perozzi et al., KDD ‘14]
- LINE [J. Tang et al.. WWW ‘15]
Trade-offs

- task-specific heuristics
- inefficient usage of statistics
Thank you