Graph Convolutional Neural Networks

Jure Leskovec
Networks: Common Language

Actors:
- Peter
- Mary
- Albert
- Tom

Movies:
- Movie 1
- Movie 2
- Movie 3

Proteins:
- Protein 1
- Protein 2
- Protein 5
- Protein 9

Network properties:
- Number of nodes, $|N| = 4$
- Number of edges, $|E| = 4$
Tasks on Networks

Classical ML tasks in networks:

- **Node classification**
  - Predict a type of a given node
- **Link prediction**
  - Predict whether two nodes are linked
- **Community detection**
  - Identify densely linked clusters of nodes
- **Network similarity**
  - How similar are two (sub)networks
Example: Node Classification

Many possible ways to create node features:

- Node degree, PageRank score, motifs, …
- Degree of neighbors, PageRank of neighbors, …
(Supervised) Machine Learning Lifecycle: This feature, that feature. Every single time!
This talk: Feature learning for networks!
Why Learn Embeddings?

The goal is to map each node into a low-dimensional space

- Distributed representation for nodes
- Similarity between nodes indicates link strength
- Encodes network information and generate node representation
Zachary’s Karate Club network:
GraphSAGE: Graph Convolutional Networks

Inductive Representation Learning on Large Graphs.
Why Is It Hard?

Images have fixed 2D structure

- Can define convolutions (CNNs)
Why Is It Hard?

Text and Speech have linear 1D structure
- Can define sliding windows

But graphs are non-Euclidean!
- Graphs have arbitrary size
- Node numbering is arbitrary (node isomorphism problem)
- Much more complex structure
From Images to Networks

Single CNN layer with 3x3 filter:

Transform information at the neighbors and combine it

- Transform “messages” $h_i$ from neighbors: $W_i h_i$
- Add them up: $\sum_i W_i h_i$
Real-World Graphs

But what if your graphs look like this?

or this:

- Examples:
  Social networks, Information networks, Knowledge graphs, Communication networks, Web graph, …
A Naïve Approach

- Join adjacency matrix and features
- Feed them into a deep neural net:

Issues with this idea:
- \( O(N) \) parameters
- Not applicable to graphs of different sizes
- Not invariant to node ordering
Graph Convolutional Networks

### Problem:
For a given subgraph how to come with canonical node ordering?

Desiderata

- Invariant to node ordering
  - No graph isomorphism problem
- Locality – operations depend on the neighbors of a given node
- Number of model parameters should be independent of graph size
- Model should be independent of graph structure and we should be able to transfer the model across graphs
Idea: Graph defines computation

Idea: Node’s neighborhood defines a computation graph

Determine node computation graph
Propagate and transform information

Learn how to propagate information across the graph to compute node features

Semi-Supervised Classification with Graph Convolutional Networks, T. N. Kipf, M. Welling, ICLR 2017

Jure Leskovec, Stanford University
Our Approach: GraphSAGE

- Each node defines its own computational graph
  - Each edge in this graph is a transformation/aggregation function
Our Approach: GraphSAGE

Update for node $A$:

$$h_A^{(k+1)} = \text{ReLU} \left( W^{(k)} h_A^{(k)} , \sum_{n \in \mathcal{N}(A)} \left( \text{ReLU} \left( Q^{(k)} h_n^{(k)} \right) \right) \right)$$

- $h_A^{(0)} =$ attributes of node $A$
- $\Sigma(\cdot)$: Aggregator function (e.g., avg., LSTM, max-pooling)

Semi-Supervised Classification with Graph Convolutional Networks. T. N. Kipf, M. Welling, ICLR 2017
GraphSAGE Algorithm

initialize representations as features

\[ h^0_v \leftarrow x_v, \forall v \in \mathcal{V}; \]

for \( k = 1 \ldots K \) do

\[ \text{for } u \in \mathcal{V} \text{ do} \]

\[ h^k_{\mathcal{N}(v)} \leftarrow \text{AGGREGATE}_k(\{h^{k-1}_u, \forall u \in \mathcal{N}(v)\}); \]

\[ h^k_v \leftarrow \sigma \left( W^k \cdot \text{CONCAT}(h^{k-1}_v, h^k_{\mathcal{N}(v)}) \right) \]

end

\[ h^k_v \leftarrow h^k_v / \|h^k_v\|_2, \forall v \in \mathcal{V} \]

end

\[ z_v \leftarrow h^K_v, \forall v \in \mathcal{V} \]

classification (cross-entropy) loss

\[ J = -\log (\sigma(z_u^T z_v)) - \frac{1}{|Q|} \cdot \sum_{q=1}^Q \mathbb{E}_{v_n \sim P_n(v)} \log (-\sigma(z_u^T z_{v_n})) \]
Overview of Model Design

1) Define a neighborhood aggregation function.

2) Define a loss function on the embeddings, $\mathcal{L}(z_u)$
Overview of Model Design

3) Train on a set of nodes, i.e., a batch of compute graphs
The same aggregation parameters are shared for all nodes.

The number of model parameters is sublinear in $|V|$ and we can generalize to unseen nodes!
Neighborhood "Convolutions"

- Neighborhood aggregation can be viewed as a center-surround filter.

- Mathematically related to spectral graph convolutions (see Bronstein et al., 2017)
Overview of Model

4) Generate embeddings for nodes as needed

Even for nodes we never trained on!!!!
Inductive Capability

Inductive node embedding generalizes to entirely unseen graphs.

- Train on one graph
- Generalize to new graph

E.g., train on protein interaction graph from model organism A and generate embeddings on newly collected data about organism B.
Inductive Capability

Many application settings constantly encounter previously unseen nodes. e.g., Reddit, YouTube, GoogleScholar, ....

Need to generate new embeddings “on the fly”
GraphSAGE: Training

- Assume parameter sharing:

  - Two types of parameters:
    - Aggregate function can have params.
    - Matrix $W^{(k)}$

- Adapt to inductive setting (e.g., unsupervised loss, neighborhood sampling, minibatch optimization)

- Generalized notion of “aggregating neighborhood”
GraphSAGE: Benefits

- Can use different aggregators $\gamma$
  - Mean (simple element-wise mean), LSTM (to a random order of nodes), Max-pooling (element-wise max)
- Can use different loss functions:
  - Cross entropy, Hinge loss, ranking loss
- Model has a constant number of parameters
- Fast scalable inference
- Can be applied to any node in any network
Quick Recap

- **Recap**: Generate node embeddings by aggregating neighborhood information.
  - Allows for parameter sharing in the encoder.
  - Allows for inductive learning.

- We saw a basic variant of this idea... now we will cover some state of the art variants from the literature.
GraphSAGE Performance: Experiments

- Compare GraphSAGE to alternative methods
  - Logistic regression on features (no network information)
  - Node2vec, extended node2vec with features

- Task: Node classification, transfer learning
  - Citation graph: 302,424 papers from 2000-05
    - Predict 6 subject codes; Train on 2000-04, test on ‘05
    - Reddit posts: 232,965 posts, 50 communities, Sep ‘14
      - What community does a post belong to? Train on first 20 days, test on remaining 10 days
  - Protein-protein interaction networks: 24 PPI networks from different tissues
    - Transfer learning of protein function: Train on 20 networks, test on 2
GraphSAGE Performance: Results

GraphSAGE performs best in all experiments. Achieves ~40% average improvement over raw features.
Application: Pinterest

Human curated collection of pins

Pin: A visual bookmark someone has saved from the internet to a board they’ve created.

Pin: Image, text, link

Board: A greater collection of ideas (pins having sth. in common).
Large-Scale Application

- Semi-Supervised node embedding for graph-based recommendations
- **Graph**: 2B pins, 1B boards, 20B edges
Graph is dynamic: need to apply to new nodes without model retraining

Rich node features: content, image
Task: Item-Item Recs

Related Pin recommendations

- Given user is looking at pin Q, what pin X are they going to save next:

  Query
  Positive
  Rnd. negative
  Hard negative
GraphSAGE Training

- Leverage inductive capability, and train on individual subgraphs
  - 300 million nodes, 1 billion edges, 1.2 billion pin pairs \((Q, X)\)
  - Large batch size: 2048 per minibatch
GraphSAGE: Inference

- Use MapReduce for model inference
- Avoids repeated computation
Experiments

Related Pin recommendations

- Given user is looking at pin Q, predict what pin X are they going to save next

- Baselines for comparison
  - Visual: VGG-16 visual features
  - Annotation: Word2Vec model
  - Combined: combine visual and annotation
  - GraphSAGE

- Setup: Embed 2B pins, perform nearest neighbor to generate recommendations
Results: Ranking

Task: Given Q, rank X as high as possible among 2B pins

- Hit-rate: Pct. P was among top-k
- MRR: Mean reciprocal rank

<table>
<thead>
<tr>
<th>Method</th>
<th>Hit-rate</th>
<th>MRR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Visual</td>
<td>17%</td>
<td>0.23</td>
</tr>
<tr>
<td>Annotation</td>
<td>14%</td>
<td>0.19</td>
</tr>
<tr>
<td>Combined</td>
<td>27%</td>
<td>0.37</td>
</tr>
<tr>
<td>GraphSAGE</td>
<td>46%</td>
<td>0.56</td>
</tr>
</tbody>
</table>
Example Recommendations
GraphSAGE: Summary

- Graph Convolution Networks
  - Generalize beyond simple convolutions
  - Fuses node features & graph info
    - State-of-the-art accuracy for node classification and link prediction.
- Model size independent of graph size; can scale to billions of nodes
  - Largest embedding to date (3B nodes, 20B edges)
- Leads to significant performance gains
How can this technology be used for biomedical problems?

- **Two examples:**
  - **Pairs of nodes:** Predicting side-effects of drug combinations
  - **Subgraph prediction:** Predicting which drug treats what disease

Polypharmacy Side Effects

Patient’s medications

Drug combination

Patient’s side effects

Polypharmacy side effect

Jure Leskovec, Stanford University
Polypharmacy Side Effects

- Polypharmacy is common to treat complex diseases and co-existing conditions
- High risk of side effects due to interactions
- 15% of the U.S. population affected
- Annual costs exceed $177 billion
- Difficult to identify manually:
  - Rare, occur only in a subset of patients
  - Not observed in clinical testing
Network & Indications Data

- **Idea:** Construct a heterogeneous graph of drugs and proteins
- **Train:** Fit a model to predict known associations of drug pairs and side effects
- **Test:** Given a query drug pair, predict candidate polypharmacy side effects

**Data:**
- **Protein-protein interaction network** [Menche et al. *Science* 15]
  - 19K nodes, 350K edges
- **Drug-protein and disease-protein links:**
  - 9k proteins, 800k drug-protein links
- **Drug side effects:** SIDER, OFFSIDES, TWOSIDES
Link Prediction Task

- Predict **labeled edges** between drugs
- Given a drug pair \((c, s)\), predict how likely an edge \((c, r_2, s)\) exists
- **Meaning:** Drug combination \((c, s)\) leads to polypharmacy side effect \(r_2\)
Neural Architecture: Encoder

Graph encoder:
- **Input**: graph, additional node features
- **Output**: node embeddings
Neural Architecture: Decoder

Graph decoder:
- **Input**: Query drug pairs and their embeddings
- **Output**: predicted links

- **Doxycycline**
- **Ciprofloxacin**
- **Simvastatin**
- **Mupirocin**

- **Input:** Query drug pairs and their embeddings
- **Output:** predicted links
Prediction Performance

- Up to 54% improvement over baselines
- First time to computationally identify side effects of drugs
How can this technology be used for biomedical problems?

- **Two examples:**
  - **Pairs of nodes:** Predicting side-effects of drug combinations
  - **Subgraph prediction:** Predicting which drug treats what disease
Prediction Problem

Goal: Predict which diseases a new drug (molecule) could treat
Insight: Networks

- Subgraphs of disease-associated proteins
- Subgraphs of drug target proteins
A drug is likely to treat a disease if they are *nearby* in "pharmacological space"
Drug repurposing: Predicting associations between subgraphs

Predict new indications:
- Obtain subgraphs by projecting drug and disease on the graph
- Predict associations between subgraphs
SUGAR: Message Passing

Embedding for subgraph $C$:

$\mu_{C \rightarrow i}$

$\mu_{i \rightarrow j}$

$\mu_{j \rightarrow C}$
Neural Network Model
Network & Indications Data

- **Protein-protein interaction network** culled from 15 knowledge databases [Menche et al. *Science* 15]
  - 19K nodes, 350K edges
- **Drug-protein and disease-protein links:**
  - DrugBank, OMIM, DisGeNET, STITCH DB and others
  - 5K drugs, 20K diseases
  - 20K drug-protein links, 560K disease-protein links
- **Drug medical indications:**
  - DrugBank, MEDI-HPS, DailyMed, RepoDB and others
  - 6K drug-disease indications
- **Side information:** Molecular pathways, disease symptoms, side effects
Experimental Setup

- **Disease-centric cross-validation**

- For each cross-validation fold:
  - Exclude *all* indications of test diseases
  - Use the remaining data to train a model

- **Query**: Given a disease, rank all drugs based on scores returned by the model
Experimental Results

Comparison to current state of the art:

- Up to 49% improvement over methods for drug repurposing
- Up to 172% improvement over methods for scoring drug-disease pairs

<table>
<thead>
<tr>
<th>Approach</th>
<th>AUPRC</th>
<th>AUROC</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUGAR</td>
<td>0.851</td>
<td>0.888</td>
</tr>
<tr>
<td>Graphlet representation</td>
<td>0.606</td>
<td>0.689</td>
</tr>
<tr>
<td>Bi-directional random walk (MBiRW)</td>
<td>0.689</td>
<td>0.692</td>
</tr>
<tr>
<td>Heterogeneous graph inference (TL_HGBI)</td>
<td>0.673</td>
<td>0.664</td>
</tr>
<tr>
<td>PREdicting Drug IndiCaTions (PREDICT)</td>
<td>0.699</td>
<td>0.680</td>
</tr>
<tr>
<td>Drug-disease network closeness (r_c)</td>
<td>0.545</td>
<td>0.631</td>
</tr>
<tr>
<td>Drug-disease network dispersion (r_d)</td>
<td>0.692</td>
<td>0.623</td>
</tr>
<tr>
<td>Gene-based drug-disease network overlap (r_o)</td>
<td>0.512</td>
<td>0.548</td>
</tr>
</tbody>
</table>
Integrating Side Information

Including additional biomedical knowledge:

<table>
<thead>
<tr>
<th>Metabolic pathways</th>
<th>Molecular functions</th>
<th>Biological processes</th>
<th>Cellular components</th>
<th>AUPRC</th>
<th>AUROC</th>
</tr>
</thead>
<tbody>
<tr>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>0.851</td>
<td>0.888</td>
</tr>
<tr>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>0.869</td>
<td>0.893</td>
</tr>
<tr>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>0.874</td>
<td>0.912</td>
</tr>
<tr>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>0.893</td>
<td>0.912</td>
</tr>
<tr>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>0.901</td>
<td>0.928</td>
</tr>
</tbody>
</table>
Drug Repurposing @ SPARK

<table>
<thead>
<tr>
<th>Drug</th>
<th>Disease</th>
<th>Rank</th>
</tr>
</thead>
<tbody>
<tr>
<td>N-acetyl-cysteine</td>
<td>cystic fibrosis</td>
<td></td>
</tr>
<tr>
<td>Xamoterol</td>
<td>neurodegeneration</td>
<td></td>
</tr>
<tr>
<td>Plerixafor</td>
<td>cancer</td>
<td></td>
</tr>
<tr>
<td>Sodium selenite</td>
<td>cancer</td>
<td></td>
</tr>
<tr>
<td><strong>Ebselen</strong></td>
<td><strong>C difficile</strong></td>
<td></td>
</tr>
<tr>
<td>Itraconazole</td>
<td>cancer</td>
<td></td>
</tr>
<tr>
<td>Bestatin</td>
<td>lymphedema</td>
<td></td>
</tr>
<tr>
<td>Bestatin</td>
<td>pulmonary arterial hypertension</td>
<td></td>
</tr>
<tr>
<td>Ketaprofen</td>
<td>lymphedema</td>
<td></td>
</tr>
<tr>
<td>Sildenafil</td>
<td>lymphatic malformation</td>
<td></td>
</tr>
<tr>
<td>Tacrolimus</td>
<td>pulmonary arterial hypertension</td>
<td></td>
</tr>
<tr>
<td>Benzamil</td>
<td>psoriasis</td>
<td></td>
</tr>
<tr>
<td>Carvedilol</td>
<td>Chagas’ disease</td>
<td></td>
</tr>
<tr>
<td>Benserazide</td>
<td>BRCA1 cancer</td>
<td></td>
</tr>
<tr>
<td>Pioglitazone</td>
<td>interstitial cystitis</td>
<td></td>
</tr>
<tr>
<td>Sirolimus</td>
<td>dystrophic epidermolysis bullosa</td>
<td></td>
</tr>
</tbody>
</table>

Given *C difficile*, where does **Ebselen** rank among all approved drugs?
## SUGAR’s Predictions

<table>
<thead>
<tr>
<th>Drug</th>
<th>Disease</th>
<th>Rank:</th>
</tr>
</thead>
<tbody>
<tr>
<td>N-acetyl-cysteine</td>
<td>cystic fibrosis</td>
<td>14/5000</td>
</tr>
<tr>
<td>Xamoterol</td>
<td>neurodegeneration</td>
<td>26/5000</td>
</tr>
<tr>
<td>Plerixafor</td>
<td>cancer</td>
<td>54/5000</td>
</tr>
<tr>
<td>Sodium selenite</td>
<td>cancer</td>
<td>36/5000</td>
</tr>
<tr>
<td><strong>Ebselen</strong></td>
<td><strong>C difficile</strong></td>
<td><strong>10/5000</strong></td>
</tr>
<tr>
<td>Itraconazole</td>
<td>cancer</td>
<td>26/5000</td>
</tr>
<tr>
<td>Bestatin</td>
<td>lymphedema</td>
<td>11/5000</td>
</tr>
<tr>
<td>Bestatin</td>
<td>pulmonary arterial hypertension</td>
<td>16/5000</td>
</tr>
<tr>
<td>Ketoprofen</td>
<td>lymphedema</td>
<td>28/5000</td>
</tr>
<tr>
<td>Sildenafil</td>
<td>lymphatic malformation</td>
<td>26/5000</td>
</tr>
<tr>
<td>Tacrolimus</td>
<td>pulmonary arterial hypertension</td>
<td>46/5000</td>
</tr>
<tr>
<td>Benzamil</td>
<td>psoriasis</td>
<td>114/5000</td>
</tr>
<tr>
<td>Carvedilol</td>
<td>Chagas’ disease</td>
<td>9/5000</td>
</tr>
<tr>
<td>Benserazide</td>
<td>BRCA1 cancer</td>
<td>41/5000</td>
</tr>
<tr>
<td>Pioglitazone</td>
<td>interstitial cystitis</td>
<td>13/5000</td>
</tr>
<tr>
<td>Sirolimus</td>
<td>dystrophic epidermolysis bullosa</td>
<td>46/5000</td>
</tr>
</tbody>
</table>

Higher rank is better

Example: SUGAR predicted Ebselen as 10th most likely candidate drug for C difficile
Feature learning for networks

$$\mathbf{vec}$$

node $$\mathbf{u}$$

$$f : \mathbf{u} \rightarrow \mathbb{R}^d$$

Feature representation, embedding
Conclusion

Results from the past 1-2 years have shown:

- Representation learning paradigm can be extended to graphs
- No feature engineering necessary
- Can effectively combine node attribute data with the network information
- State-of-the-art results in a number of domains/tasks
- Use end-to-end training instead of multi-stage approaches for better performance
Next steps:

- Multimodal & dynamic/evolving settings
- Domain-specific adaptations (e.g. for recommender systems)
- Graph generation
- Prediction beyond simple pairwise edges
  - Multi-hop edge prediction
- Theory
PhD Students

Claire Donnat  
Mitchell Gordon  
David Hallac  
Emma Pierson  
Geet Sethi  
Himabindu Lakkaraju  
Rex Ying  
Tim Althoff  
Will Hamilton

Post-Doctoral Fellows

Baharan Mirzasoleiman  
Marinka Zitnik  
Michele Catasta  
Srijan Kumar

Research Staff

Stephen Bach  
Adrijan Bradaschia  
Rok Sosic

Industry Partnerships

Funding

Collaborators

Dan Jurafsky, Linguistics, Stanford University  
Christian Danescu-Miculescu-Mizil, Information Science, Cornell University  
Stephen Boyd, Electrical Engineering, Stanford University  
David Gleich, Computer Science, Purdue University  
VS Subrahmanian, Computer Science, University of Maryland  
Sarah Kunz, Medicine, Harvard University  
Russ Altman, Medicine, Stanford University  
Jochen Profit, Medicine, Stanford University  
Eric Horvitz, Microsoft Research  
Jon Kleinberg, Computer Science, Cornell University  
Sendhil Mullainathan, Economics, Harvard University  
Scott Delp, Bioengineering, Stanford University  
Jens Ludwig, Harris Public Policy, University of Chicago

Jure Leskovec, Stanford University
References

- **node2vec: Scalable Feature Learning for Networks**

- **Predicting multicellular function through multi-layer tissue networks.**

- **Inductive Representation Learning on Large Graphs.**
  W. Hamilton, R. Ying, J. Leskovec. NIPS 2017

- **Representation Learning on Graphs: Methods and Applications.**
  W. Hamilton, R. Ying, J. Leskovec.

- **Modeling polypharmacy side effects with graph convolutional networks.**

- **Code:**
  - http://snap.stanford.edu/node2vec
  - http://snap.stanford.edu/graphsage