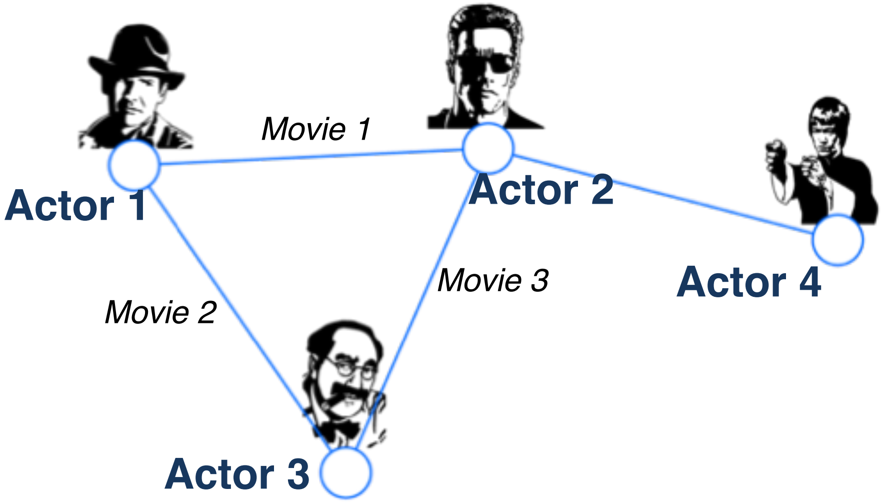
Graph Representation Learning with Graph Convolutional Networks

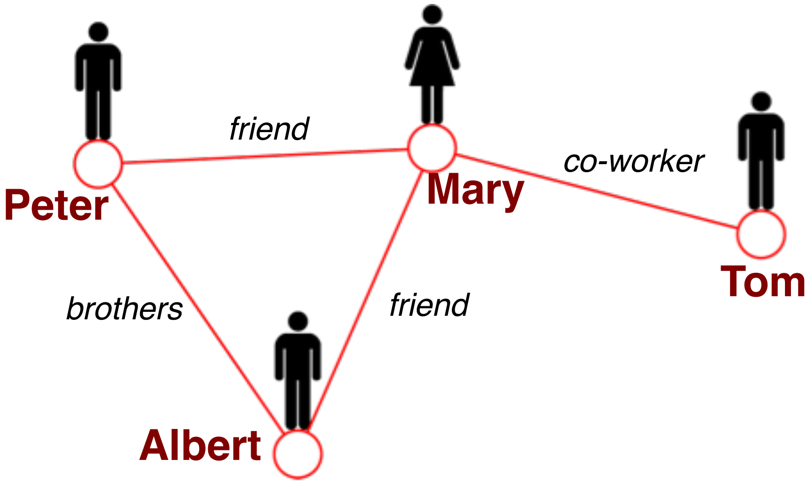
Jure Leskovec

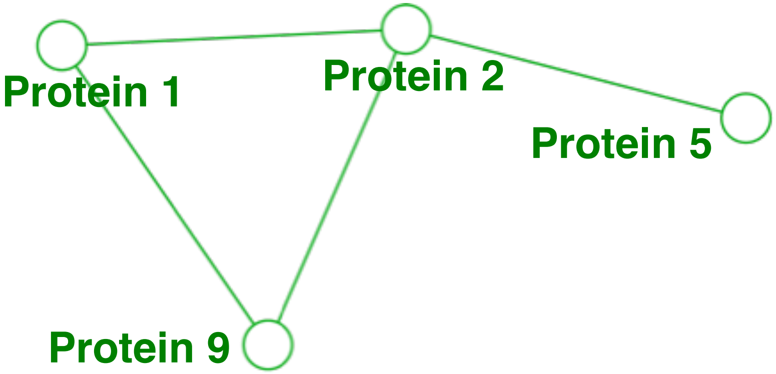
STANFORD INFOLAB

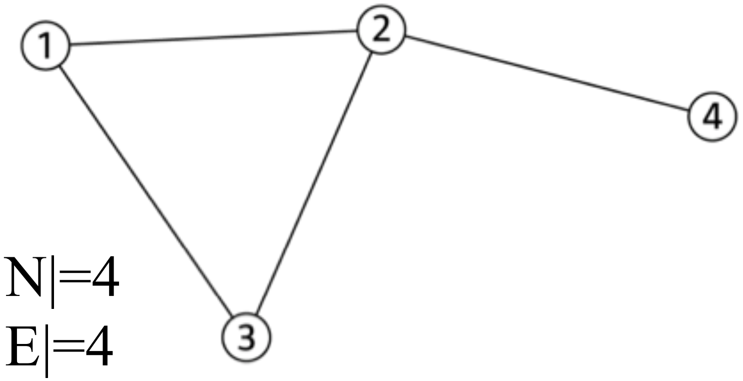
CHAN ZUCKERBERG BIOHUB

Networks: Common Language



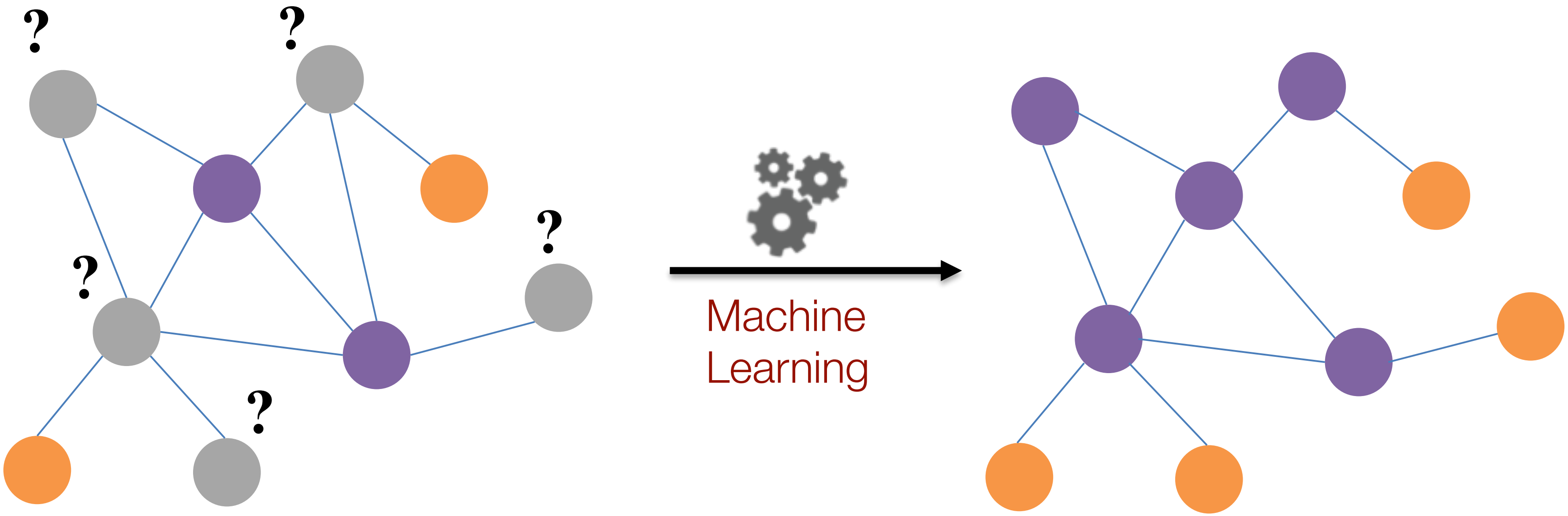






|N|=4
|E|=4

Example: Node Classification

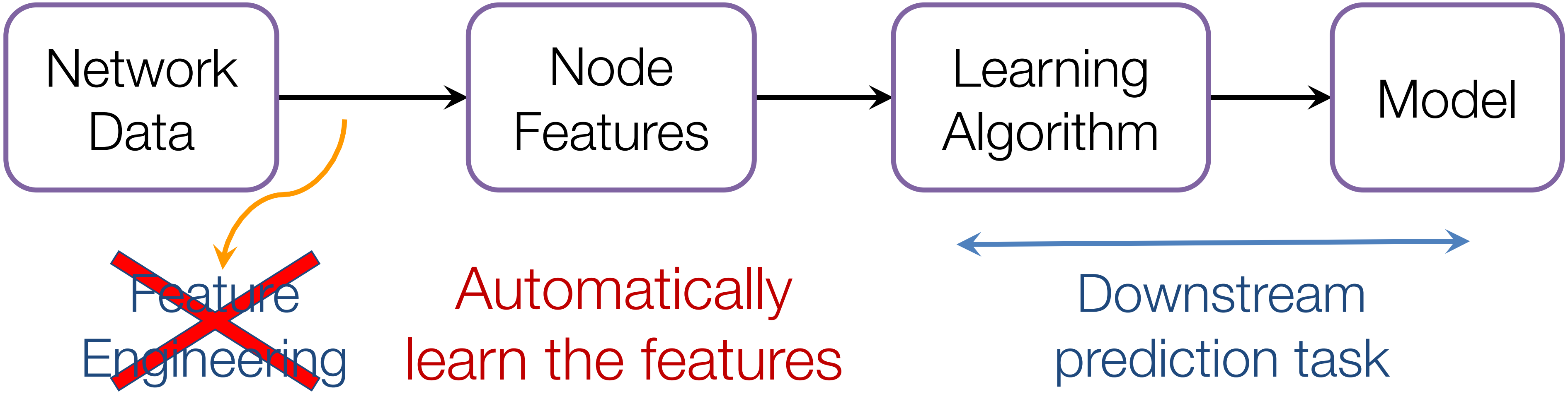


Machine Learning

Many possible ways to create node features:

- Node degree, PageRank score, motifs, …
- Degree of neighbors, PageRank of neighbors, …

Machine Learning Lifecycle



Network Data

Node Features

Learning Algorithm

Model

~~Feature Engineering~~

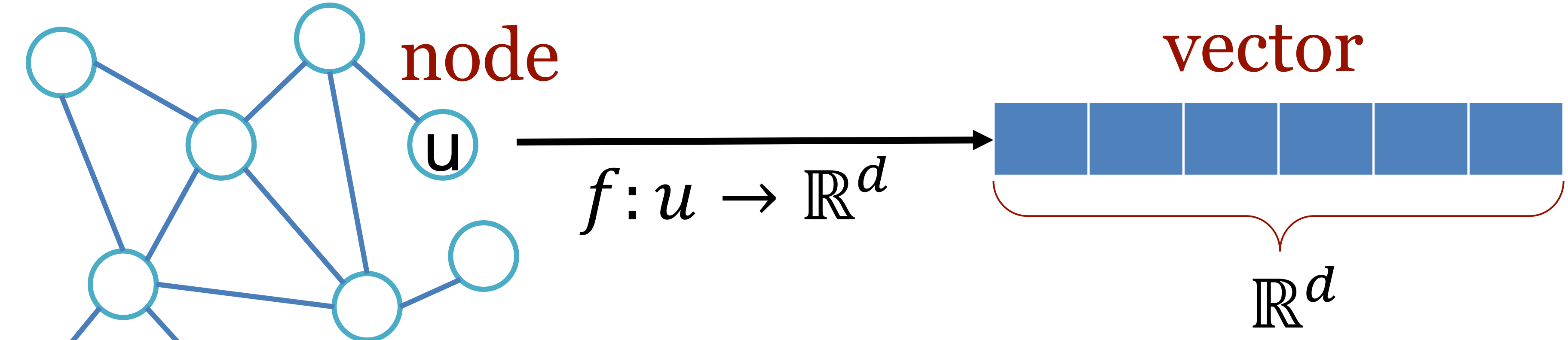
Automatically learn the features

Downstream prediction task

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

Feature Learning in Graphs

This talk: Feature learning for networks!



node

u

vector

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

$\mathbb{R}^d$

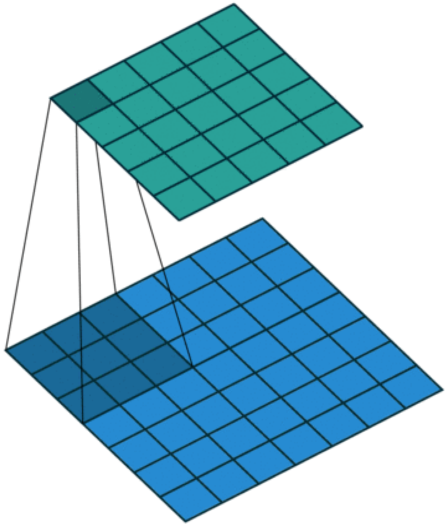
Feature representation, embedding

GraphSAGE: Graph Convolutional Networks

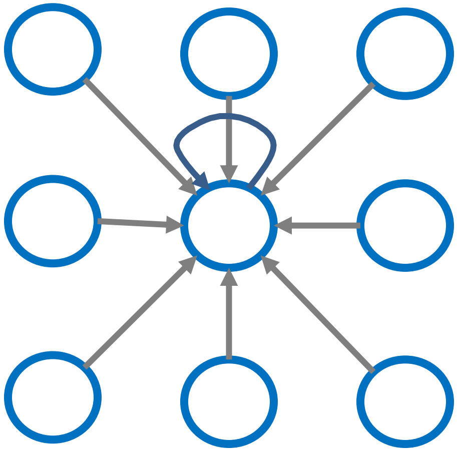
Inductive Representation Learning on Large Graphs.
W. Hamilton, R. Ying, J. Leskovec. Neural Information Processing Systems (NIPS), 2017.
Representation Learning on Graphs: Methods and Applications.
W. Hamilton, R. Ying, J. Leskovec. IEEE Data Engineering Bulletin, 2017.

From Images to Networks

Single CNN layer with 3x3 filter:



Image



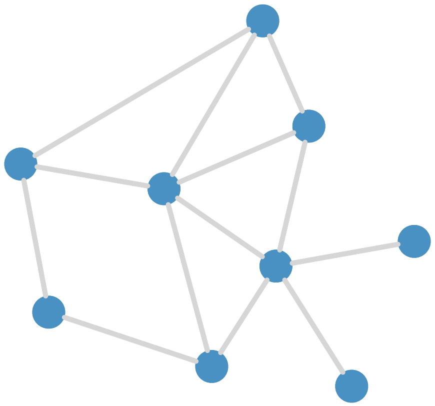
Graph

Transform information at the neighbors and combine it

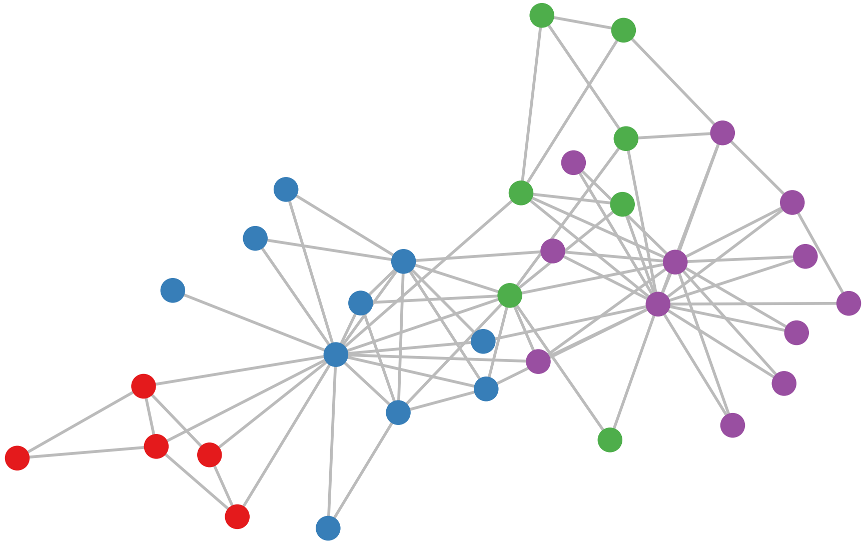
- Transform "messages" h_i from neighbors: $W_i\ h_i$
- Add them up: $\sum_i W_i\ h_i$

Real-World Graphs

But what if your graphs look like this?



or this:

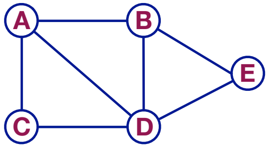


- Examples:

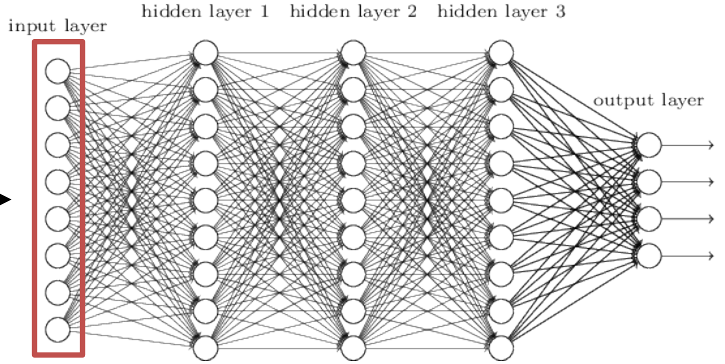
  Social networks, Information networks, Knowledge graphs, Communication networks, Web graph, …

A Naïve Approach

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



$$\begin{array}{c|ccccc|cc} & A & B & C & D & E & \text{Feat} & \\ \hline A & 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ B & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ C & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ D & 1 & 1 & 1 & 0 & 1 & 1 & 1 \\ E & 0 & 1 & 0 & 1 & 0 & 1 & 0 \end{array} \longrightarrow$$

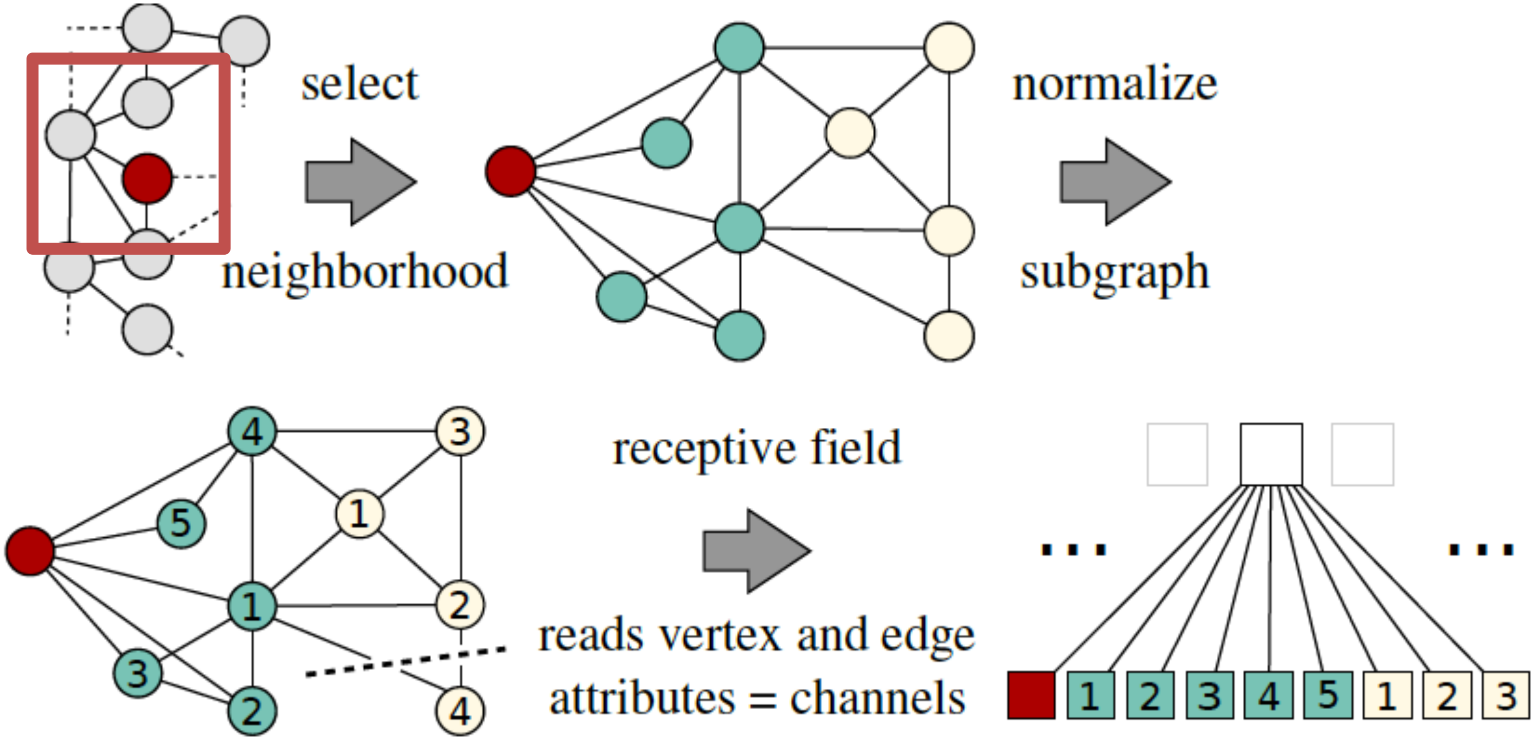


?

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

Graph Convolutional Networks

- Graph Convolutional Networks:

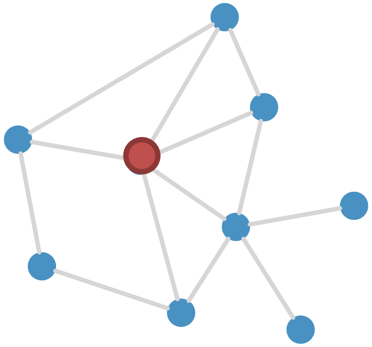


- Problem: For a given subgraph how to come with canonical node ordering?

Niepert, Mathias, Mohamed Ahmed, and Konstantin Kutzkov. "Learning convolutional neural networks for graphs." ICML. 2016. (image source)

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

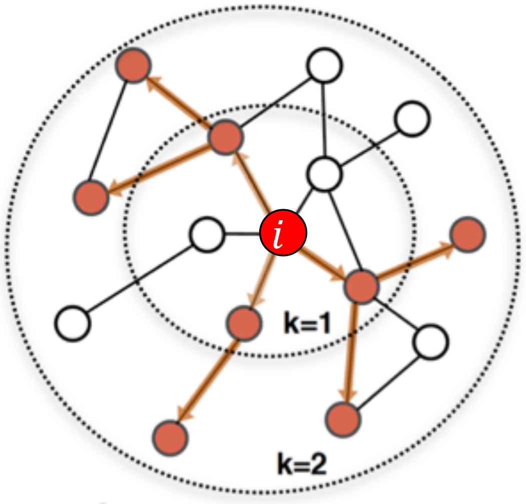


GraphSAGE

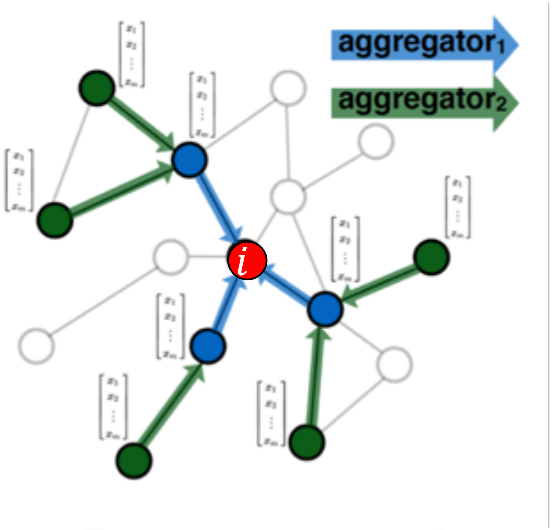
- Adapt the GCN idea to inductive node embedding
- Generalize beyond simple convolutions
- Demonstrate that this generalization
  - Leads to significant performance gains
  - Allows the model to learn about local structures

Idea: Graph defines computation

Idea: Node’s neighborhood defines a computation graph



Determine node computation graph

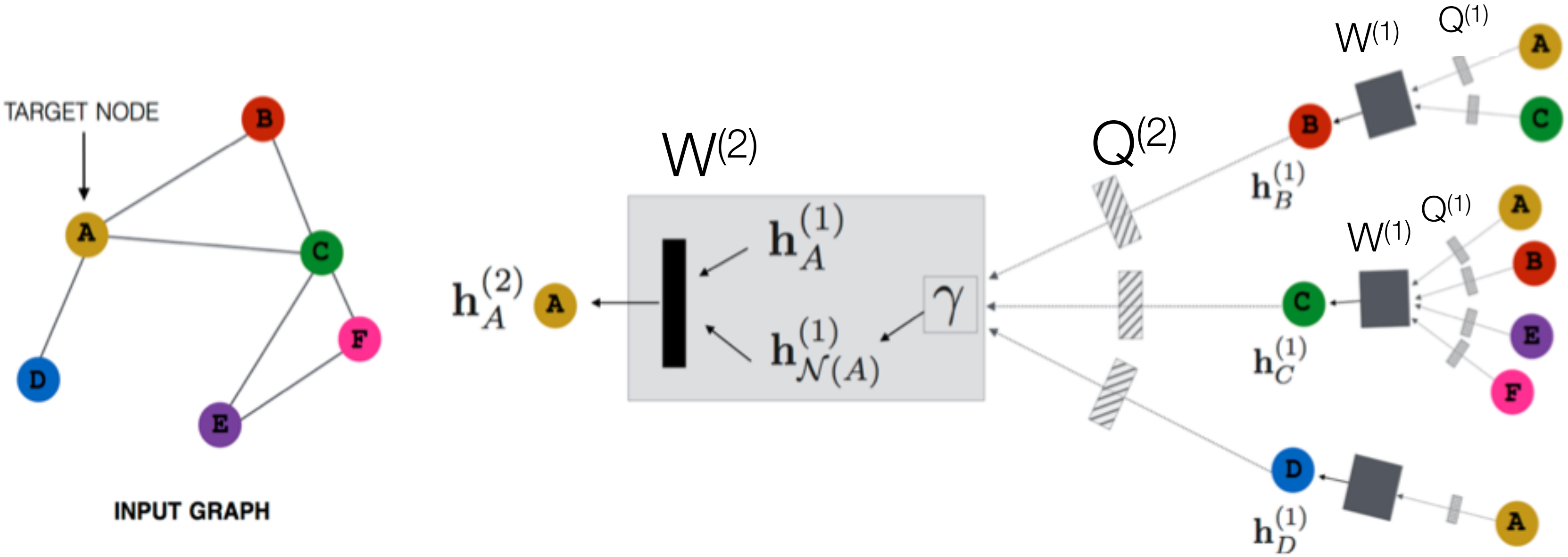


Propagate and transform information

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

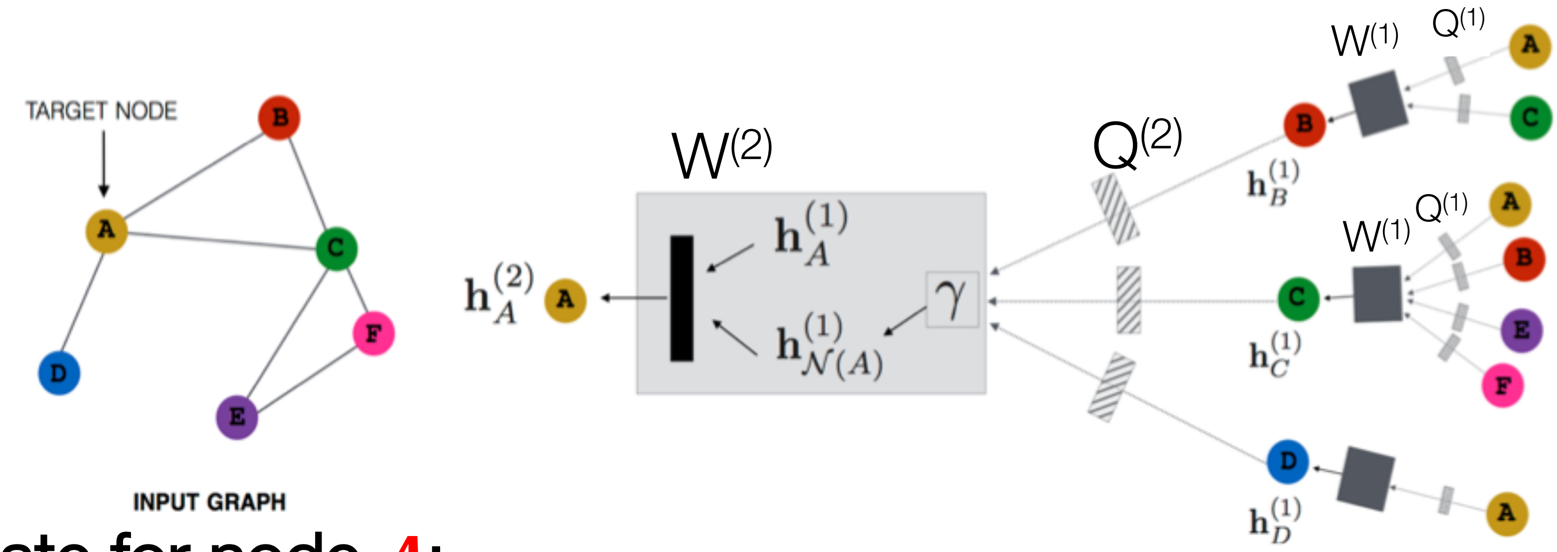
Semi-Supervised Classification with Graph Convolutional Networks. T. N. Kipf, M. Welling, ICLR 2017

Our Approach: GraphSAGE



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

Our Approach: GraphSAGE



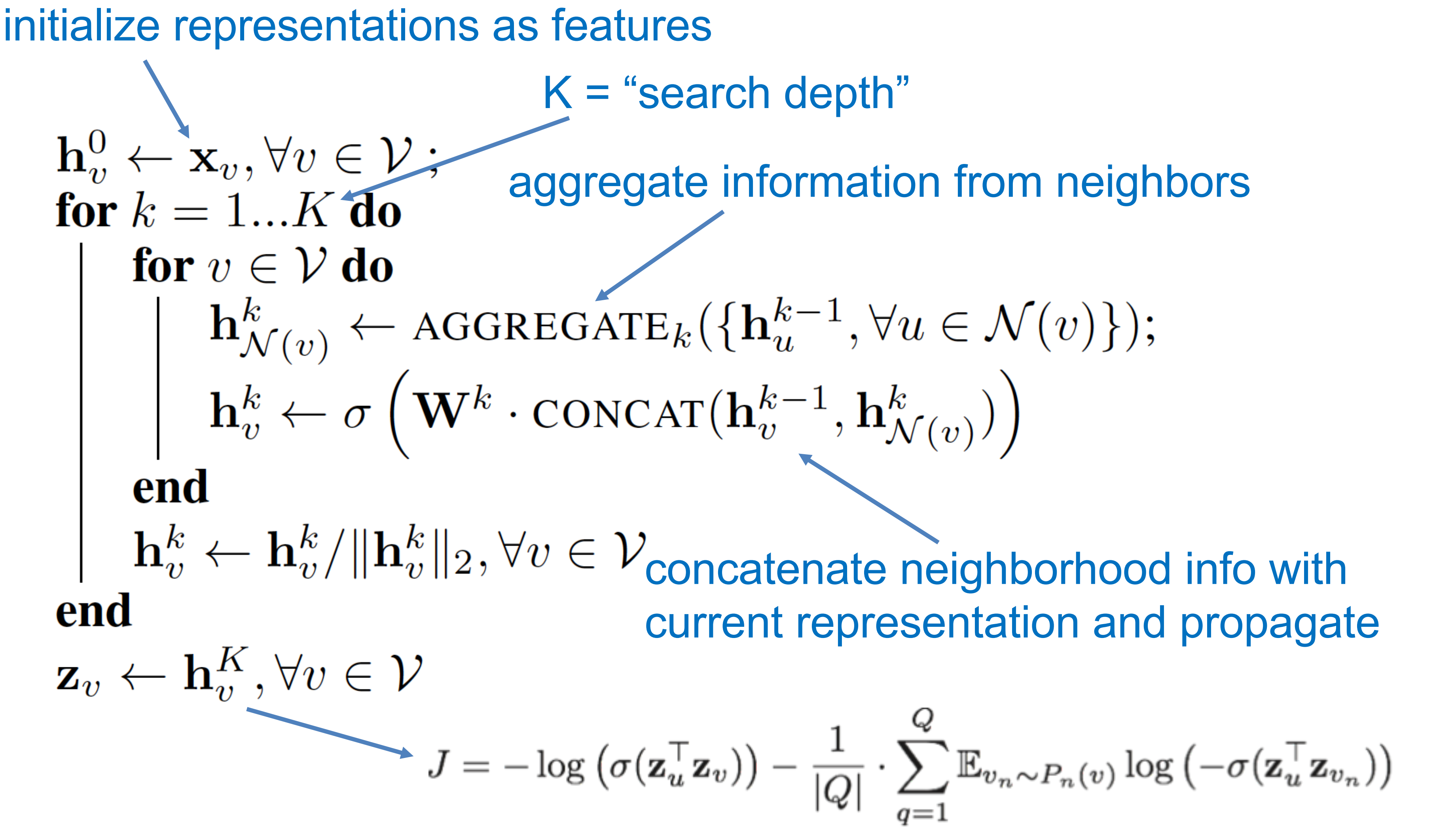
Update for node ***A***:

$$\underbrace{h_A^{(k+1)}}_{\substack{k+1^{th}\text{ level}\\ \text{features of node } A}} = ReLU\left(\underbrace{W^{(k)}h_A^{(k)}}_{\substack{\text{Transform } A\text{'s own}\\ \text{features from level } k}}, \sum_{n\in\mathcal{N}(A)} \underbrace{\left(ReLU\left(Q^{(k)}h_n^{(k)}\right)\right)}_{\substack{\text{Transform and aggregate}\\ \text{features of neighbors } n}}\right)$$

- $h_A^{(0)}$ = 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

K = “search depth”

aggregate information from neighbors

```
$\mathbf{h}_v^0 \leftarrow \mathbf{x}_v, \forall v \in \mathcal{V}$ ;
for $k = 1...K$ do
    for $v \in \mathcal{V}$ do
        $\mathbf{h}_{\mathcal{N}(v)}^k \leftarrow \text{AGGREGATE}_k(\{\mathbf{h}_u^{k-1}, \forall u \in \mathcal{N}(v)\})$;
        $\mathbf{h}_v^k \leftarrow \sigma\left(\mathbf{W}^k \cdot \text{CONCAT}(\mathbf{h}_v^{k-1}, \mathbf{h}_{\mathcal{N}(v)}^k)\right)$
    end
    $\mathbf{h}_v^k \leftarrow \mathbf{h}_v^k / \|\mathbf{h}_v^k\|_2, \forall v \in \mathcal{V}$
end
$\mathbf{z}_v \leftarrow \mathbf{h}_v^K, \forall v \in \mathcal{V}$
```

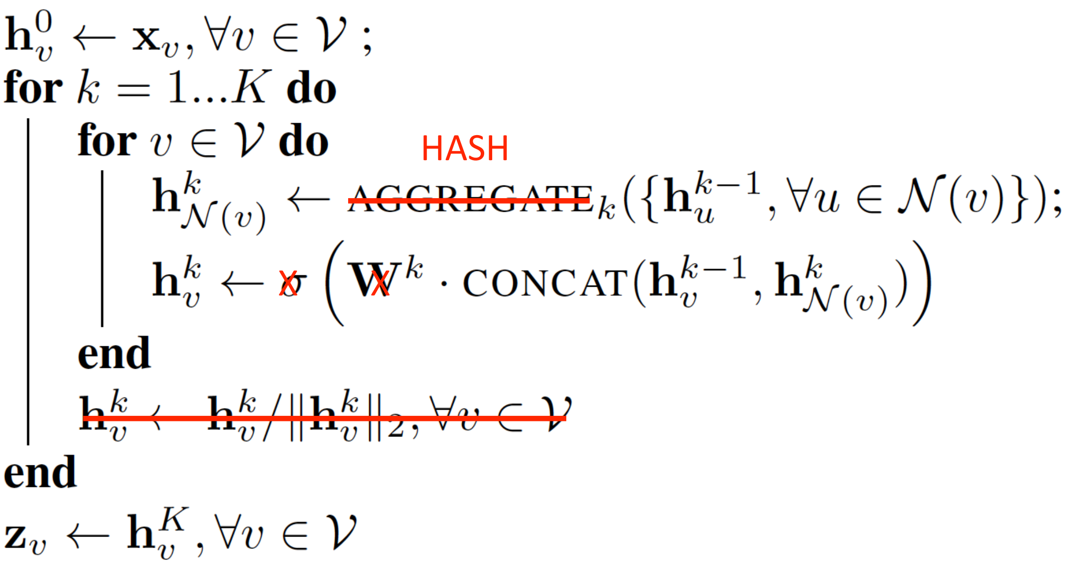
concatenate neighborhood info with current representation and propagate

$$J = -\log\left(\sigma(\mathbf{z}_u^\top \mathbf{z}_v)\right) - \frac{1}{|Q|} \cdot \sum_{q=1}^{Q} \mathbb{E}_{v_n \sim P_n(v)} \log\left(-\sigma(\mathbf{z}_u^\top \mathbf{z}_{v_n})\right)$$

classification (cross-entropy) loss

WL isomorphism test

- The classic Weisfeiler-Lehman graph isomorphism test is a special case of GraphSAGE
- We replace the hash function with trainable neural nets:

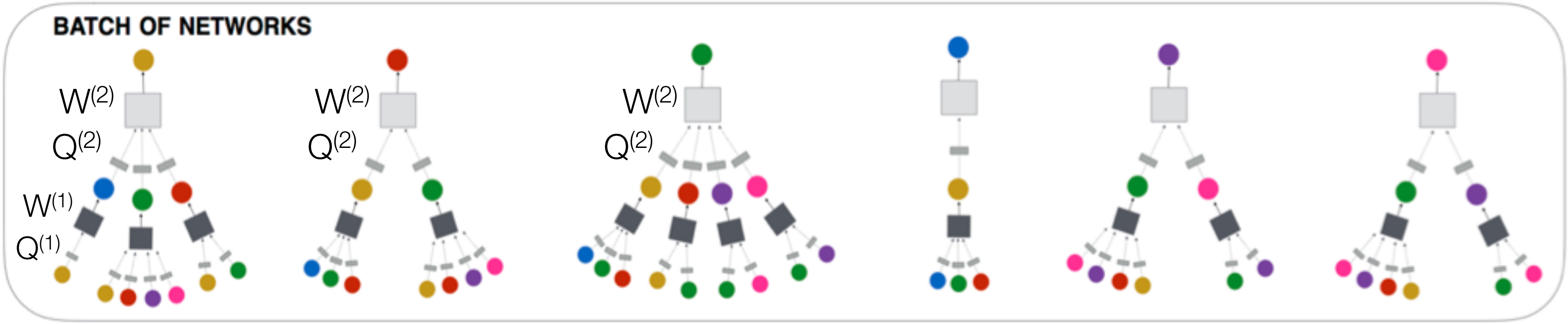


```
$\mathbf{h}_v^0 \leftarrow \mathbf{x}_v, \forall v \in \mathcal{V}$ ;
for $k = 1...K$ do
    for $v \in \mathcal{V}$ do
        $\mathbf{h}^k_{\mathcal{N}(v)} \leftarrow \overset{\text{HASH}}{\cancel{\text{AGGREGATE}}}_k(\{\mathbf{h}_u^{k-1}, \forall u \in \mathcal{N}(v)\})$;
        $\mathbf{h}_v^k \leftarrow \cancel{\sigma}\left(\cancel{\mathbf{W}}^k \cdot \text{CONCAT}(\mathbf{h}_v^{k-1}, \mathbf{h}^k_{\mathcal{N}(v)})\right)$
    end
    ~~$\mathbf{h}_v^k \leftarrow \mathbf{h}_v^k / \|\mathbf{h}_v^k\|_2, \forall v \in \mathcal{V}$~~
end
$\mathbf{z}_v \leftarrow \mathbf{h}_v^K, \forall v \in \mathcal{V}$
```

Shervashidze, Nino, et al. "Weisfeiler-Lehman graph kernels." Journal of Machine Learning Research (2011).

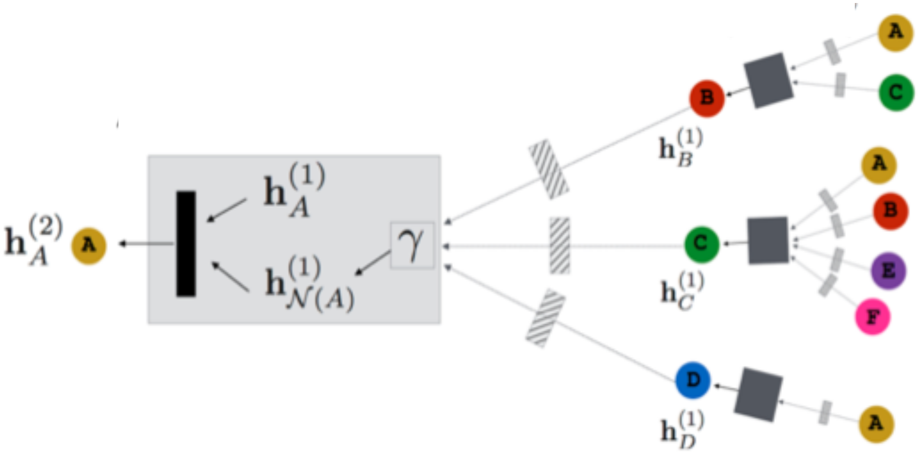
GraphSAGE: Training

- 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

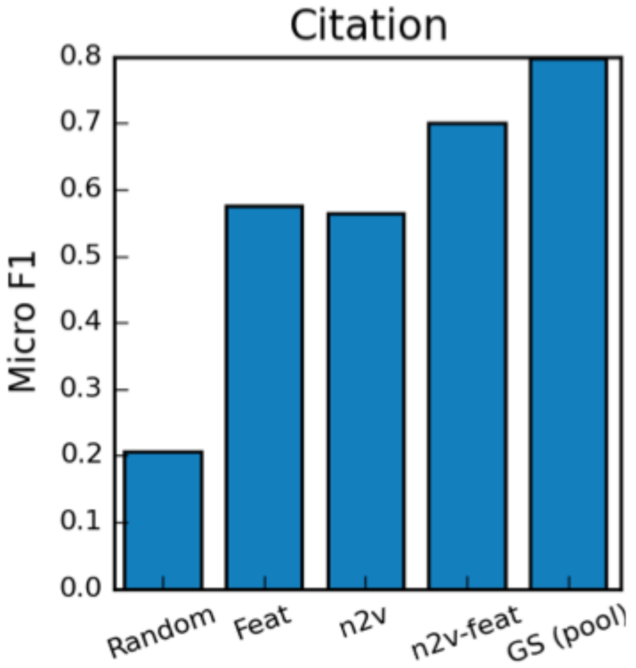


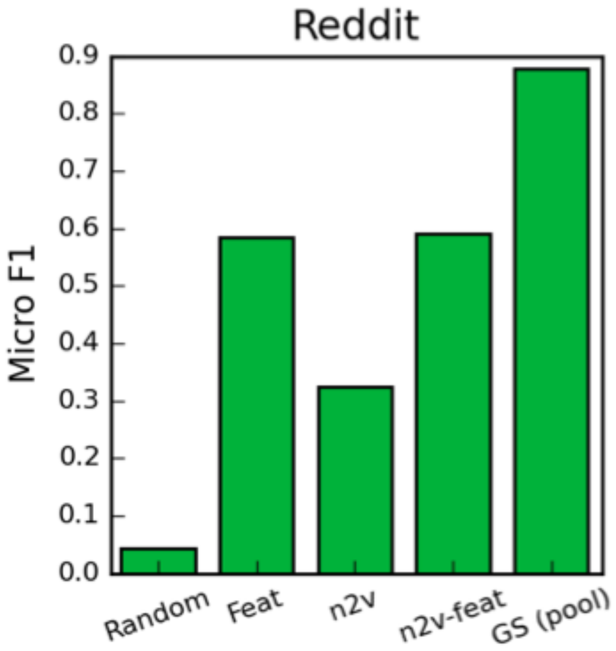
- Can use different aggregators γ
  - Mean (simple element-wise mean), LSTM (to a random order of nodes), Max-pooling (element-wise max)
- Can use different loss functions:
  - Cross entropy, Hinge loss, ranking loss
- Model has a constant number of parameters
- Fast scalable inference
- Can be applied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

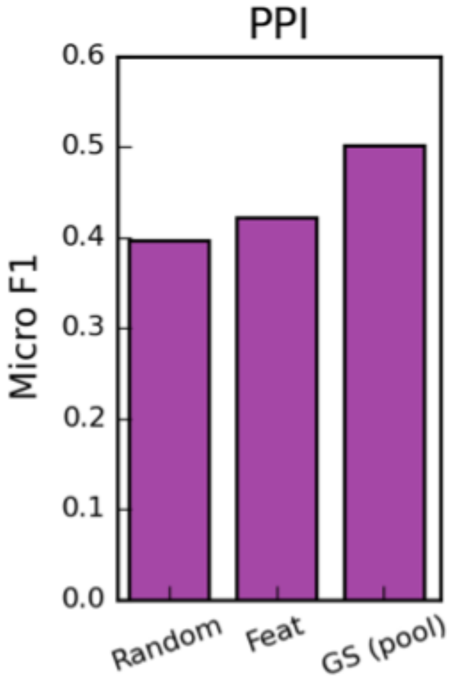
GraphSAGE Performance: Experiments

- 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



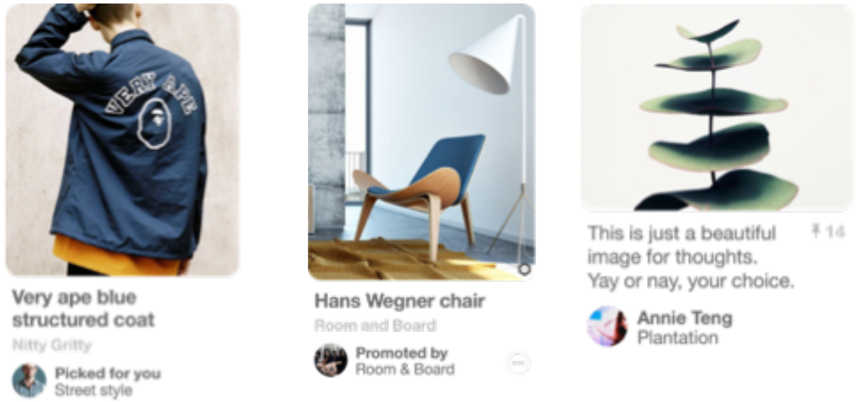




GraphSAGE performs best in all experiments.
Achieves ~40% average improvement over raw features.

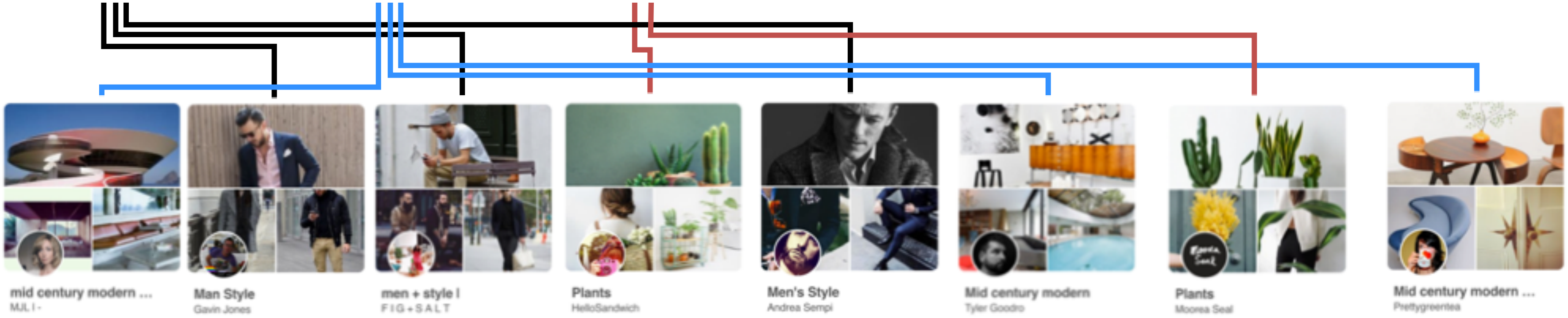
Application: Pinterest

Human curated collection of pins



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

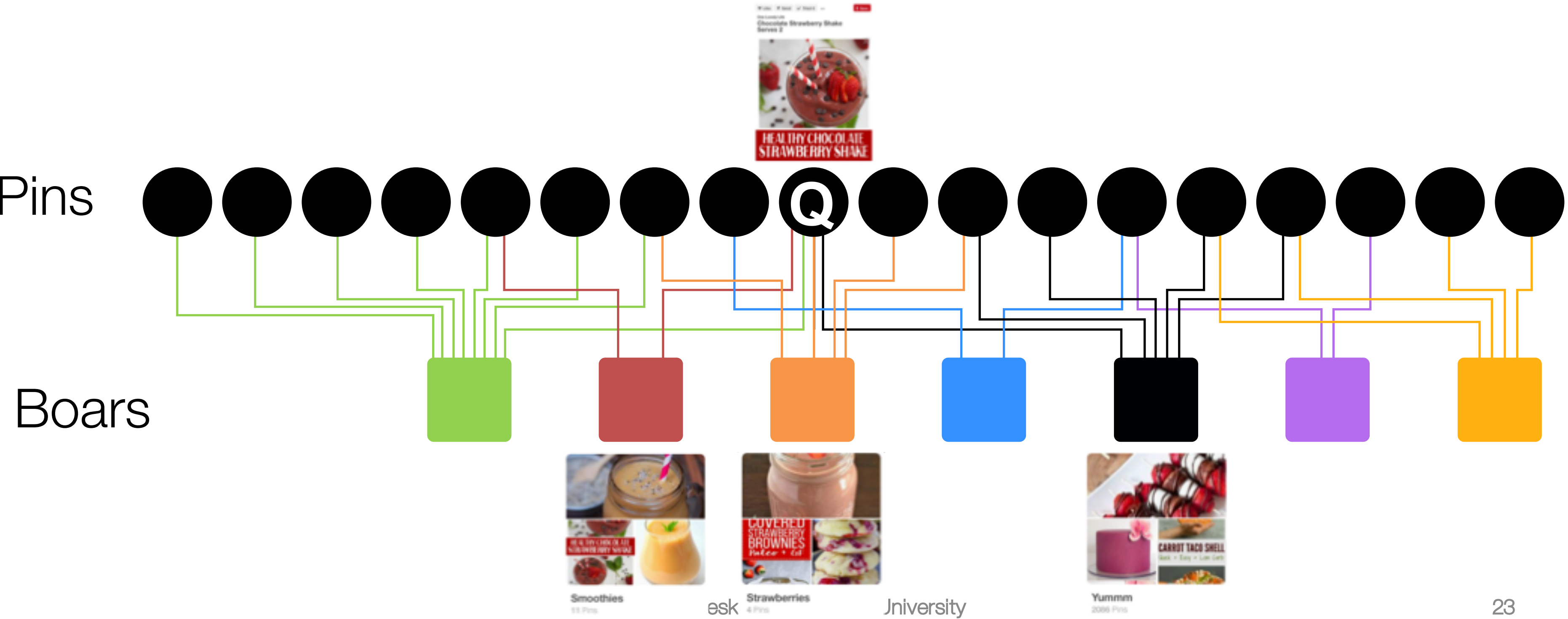
Pin: Image, text, link



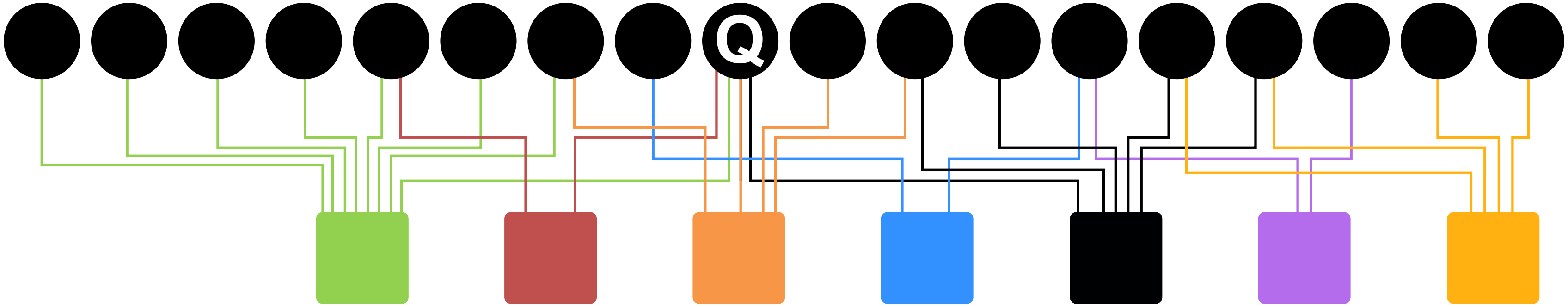
Board: A greater collection of ideas (pins having sth. in common).

Large-Scale Application

- Semi-Supervised node embedding for graph-based recommendations
- **Graph:** 2B pins, 1B boards, 20B edges


HEALTHY CHOCOLATE STRAWBERRY SHAKE
Pins
Q
Boars
Smoothies
COVERED STRAWBERRY BROWNIES
Strawberries
CARROT TACO SHELL
Yummm

Pinterest Graph

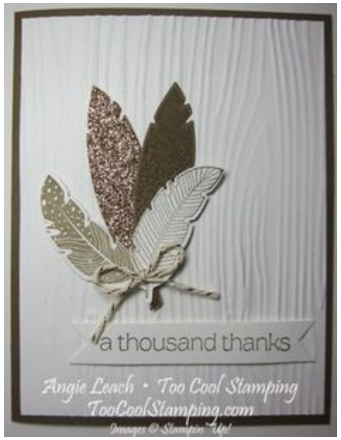


- Graph is dynamic: need to apply to new nodes without model retraining
- Rich node features: content, image

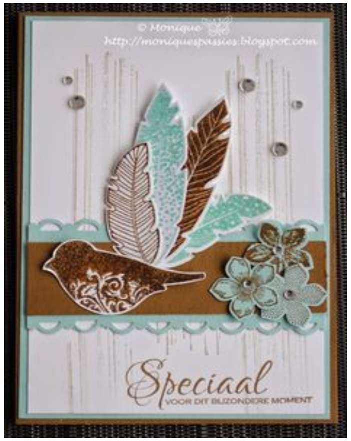
Task: Item-Item Recs

Related Pin recommendations

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

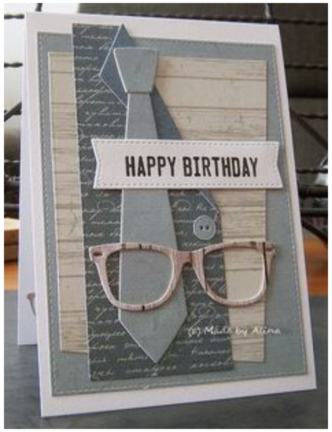


Query


© Monique
http://moniquespassies.blogspot.com
Speciaal
VOOR DIT BIJZONDERE MOMENT

Positive

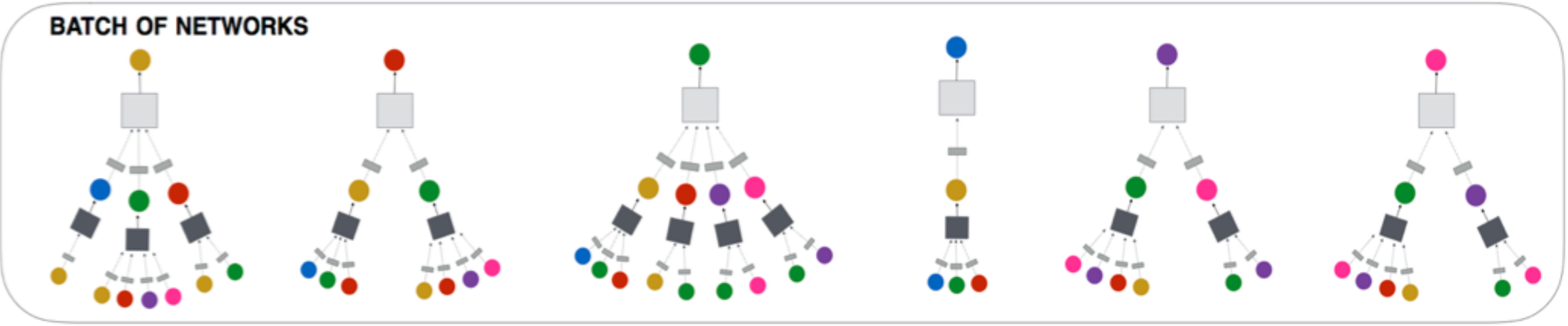
Rnd. negative



Hard negative

GraphSAGE Training

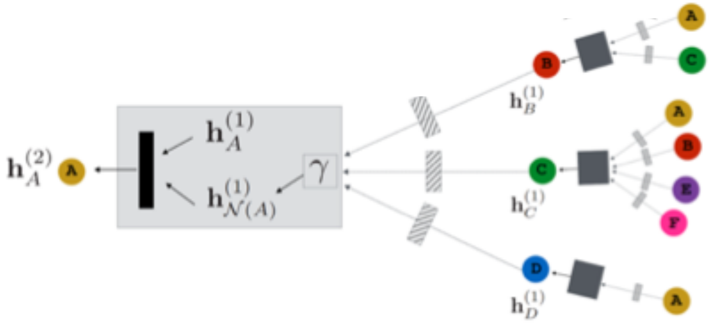
- Leverage inductive capability, and train on individual subgraphs
  - 300 million nodes, 1 billion edges, 1.2 billion pin pairs (Q, X)

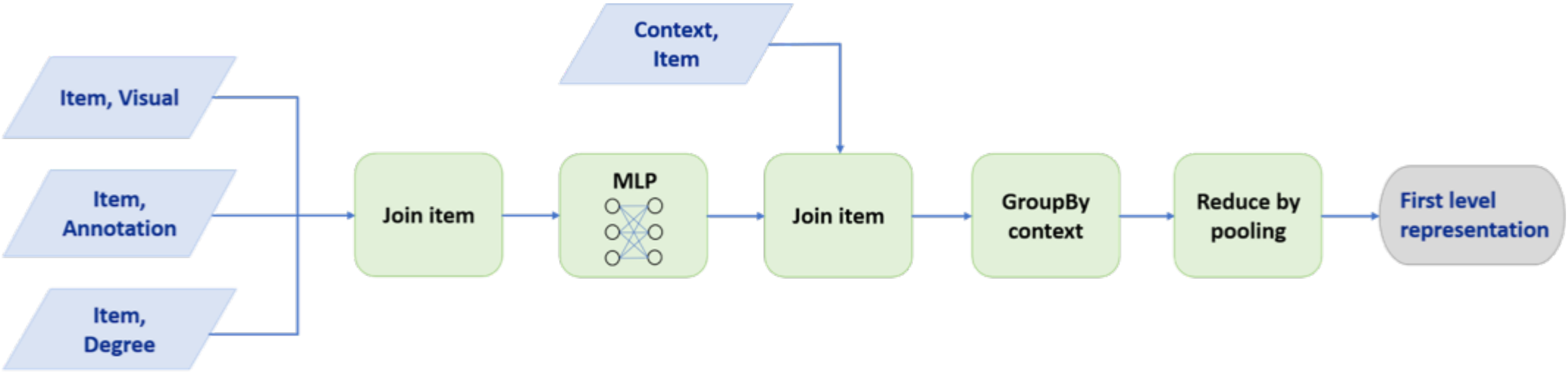


- Large batch size: 2048 per minibatch

GraphSAGE: Inference

- Use MapReduce for model inference





Item, Visual

Item, Annotation

Item, Degree

Join item

MLP

Context, Item

Join item

GroupBy context

Reduce by pooling

First level representation

- 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
  - **RW:** Random-walk based algorithm
  - **GraphSAGE**
- **Setup:** Embed 2B pins, perform nearest neighbor to generate recommendations

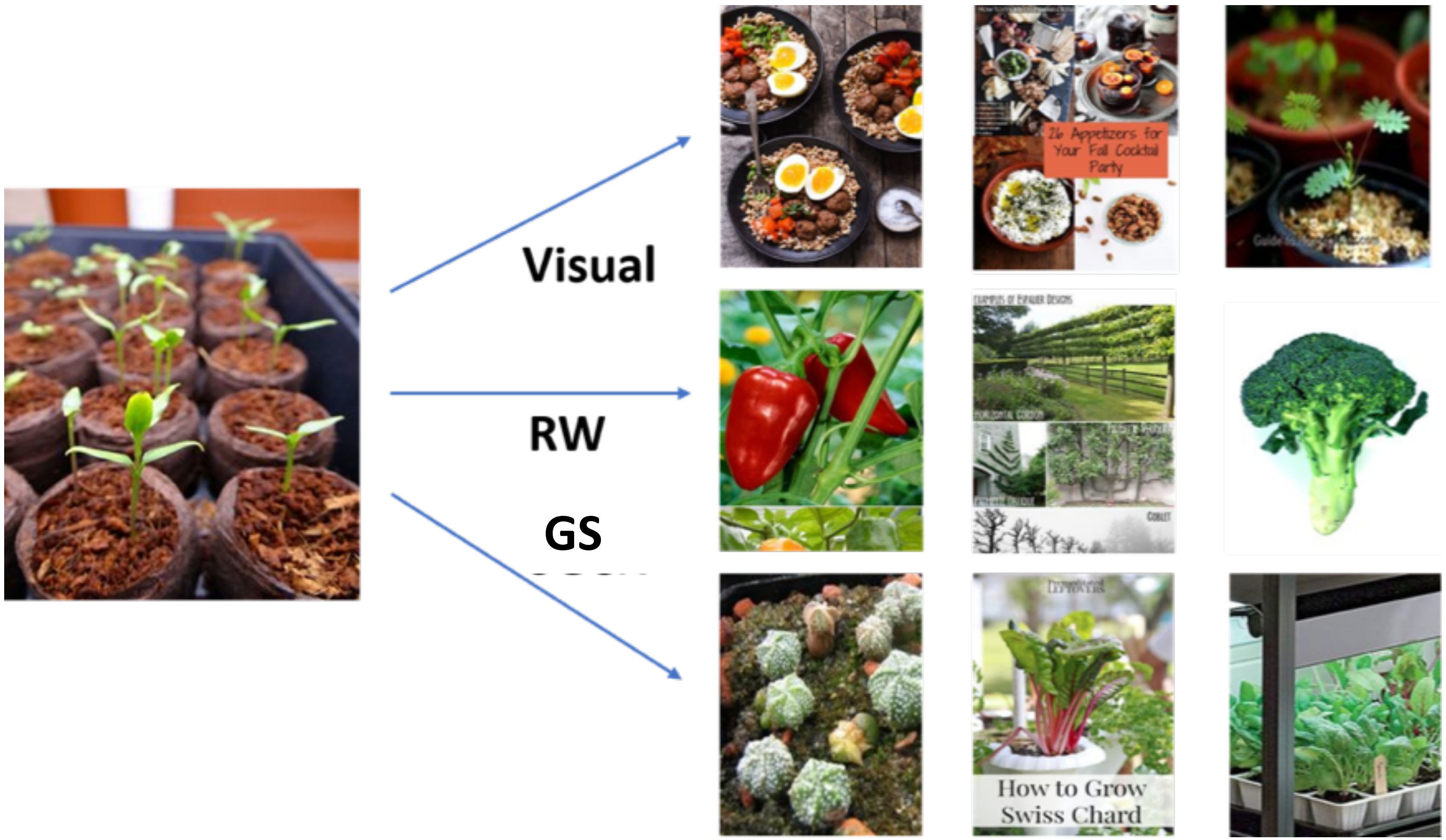
Results: Ranking

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

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

Method	Hit-rate	MRR
Visual	17%	0.23
Annotation	14%	0.19
Combined	27%	0.37
GraphSAGE	46%	0.56

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?

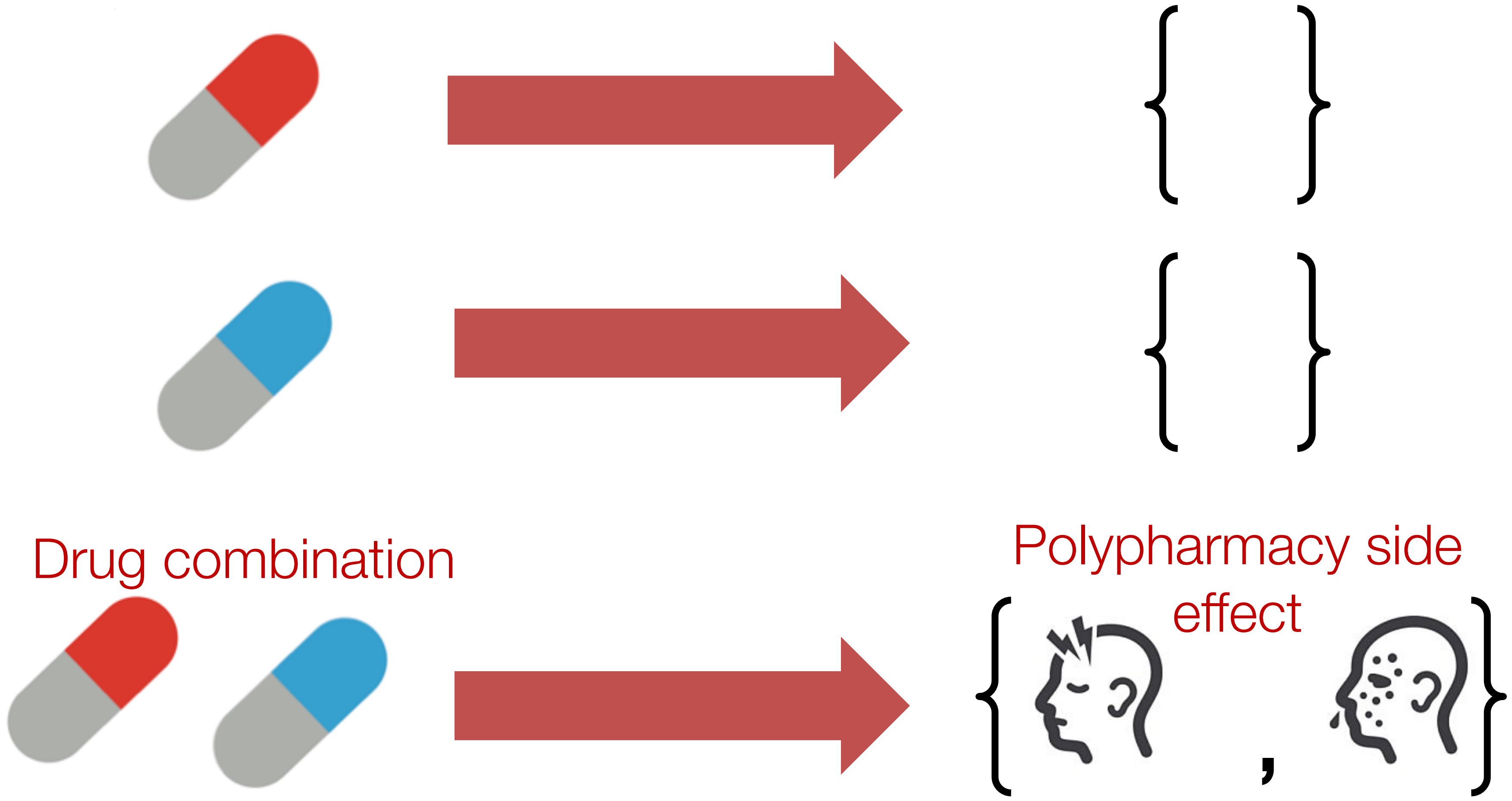
- **Two examples:**
  - **Pairs of nodes:** Predicting side-effects of drug combinations
  - **Subgraph prediction:** Predicting which drug treats what disease

Modeling polypharmacy side effects with graph convolutional networks. M. Zitnik, M. Agrawal, J. Leskovec. BioArxiv, 2017.

Polypharmacy Side Effects

Patient's medications

Patient's side effects



Drug combination

Polypharmacy side effect

Polypharmacy Side Effects

Pa cts

- 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

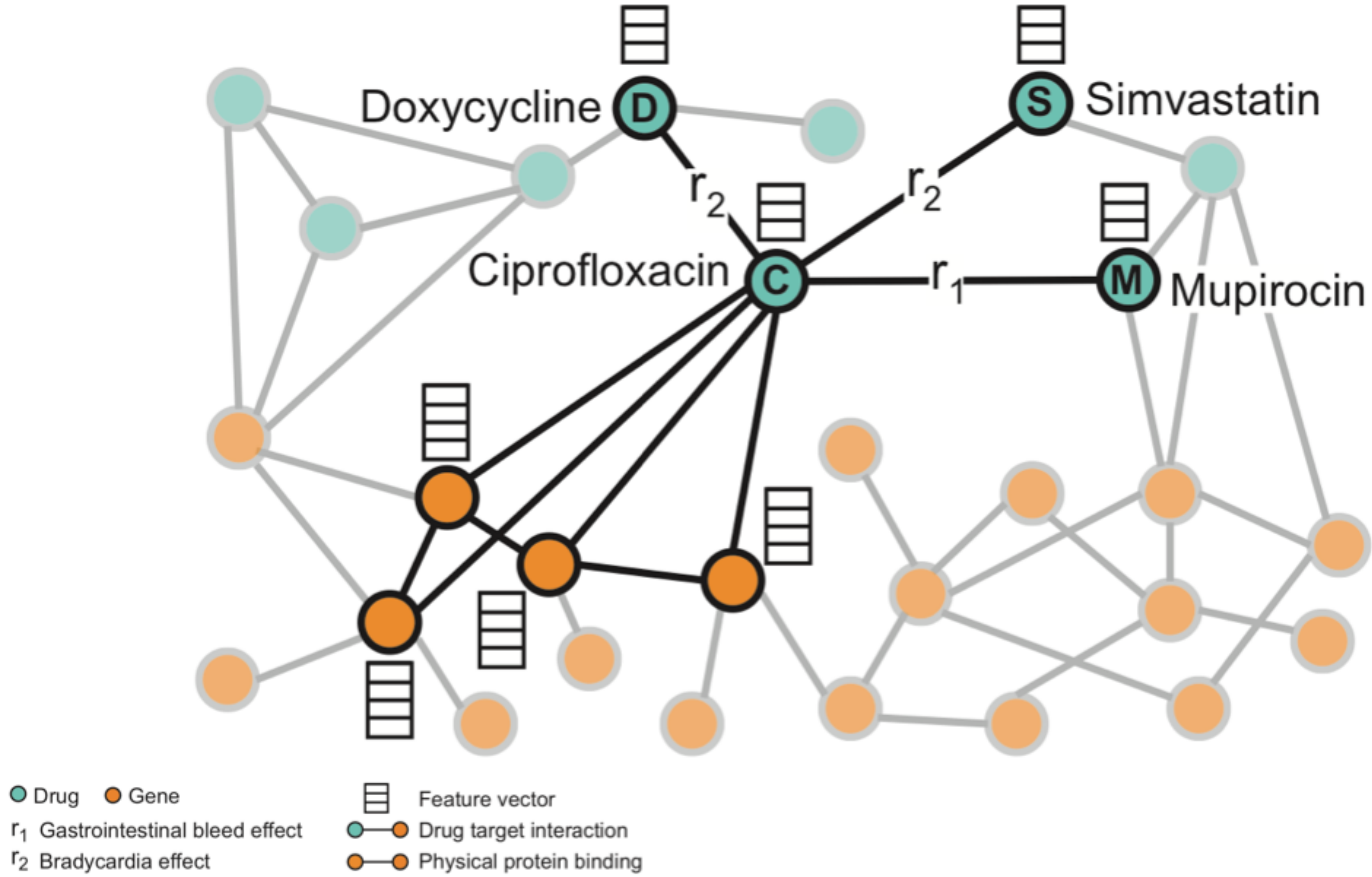
Network & Indications Data

- 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:

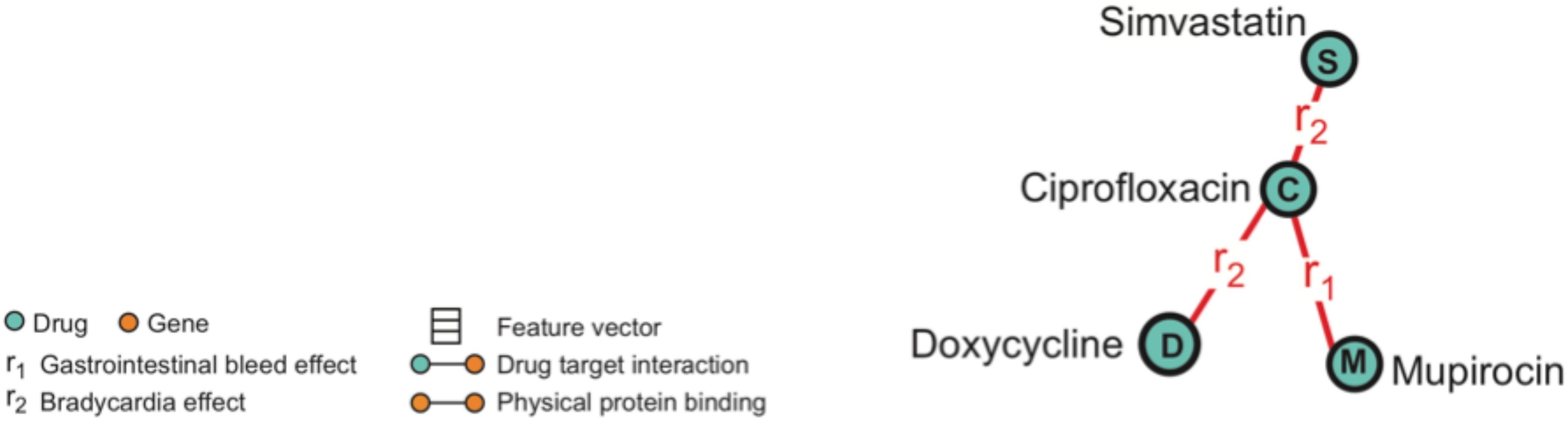
- **Protein-protein interaction network** [Menche et al. *Science* 15]
  - 19K nodes, 350K edges
- **Drug-protein and disease-protein links:**
  - 9k proteins, 800k drug-protein links
- **Drug side effects:** SIDER, OFFSIDES, TWOSIDES

Heterogeneous Graph

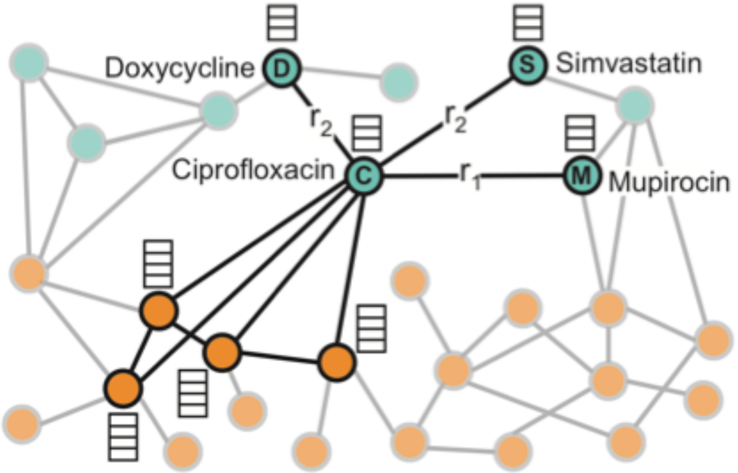


Link Prediction Task

- 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

● Drug ● Gene

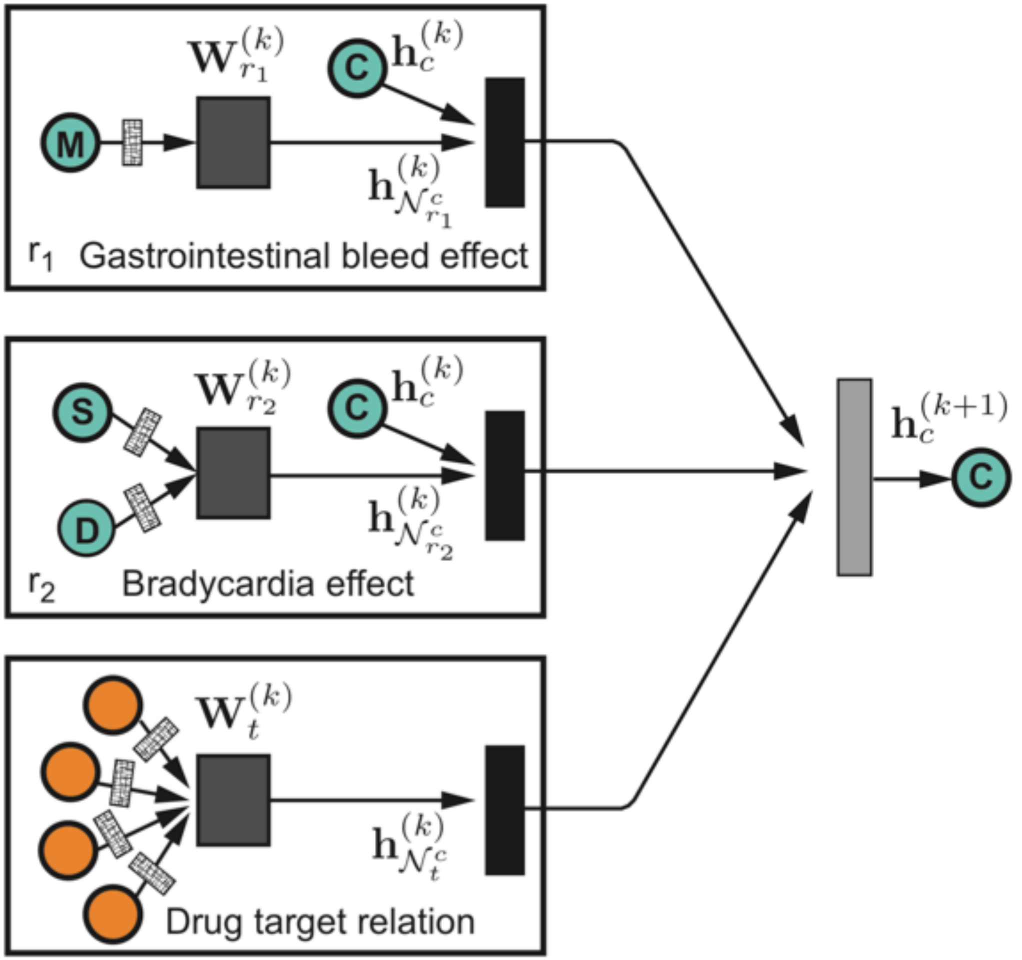
r_1 Gastrointestinal bleed effect

r_2 Bradycardia effect

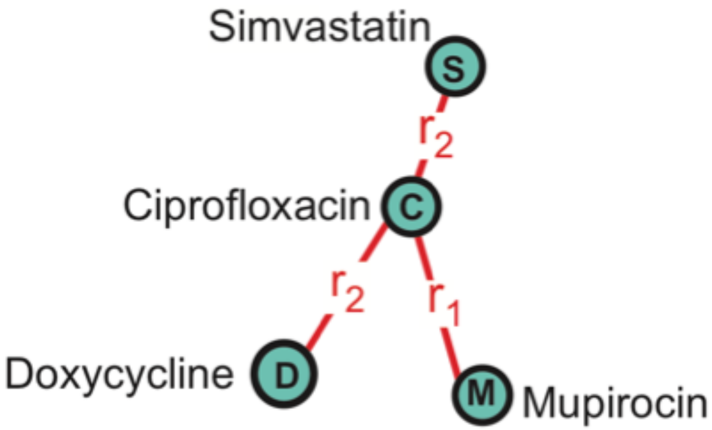
▤ Feature vector

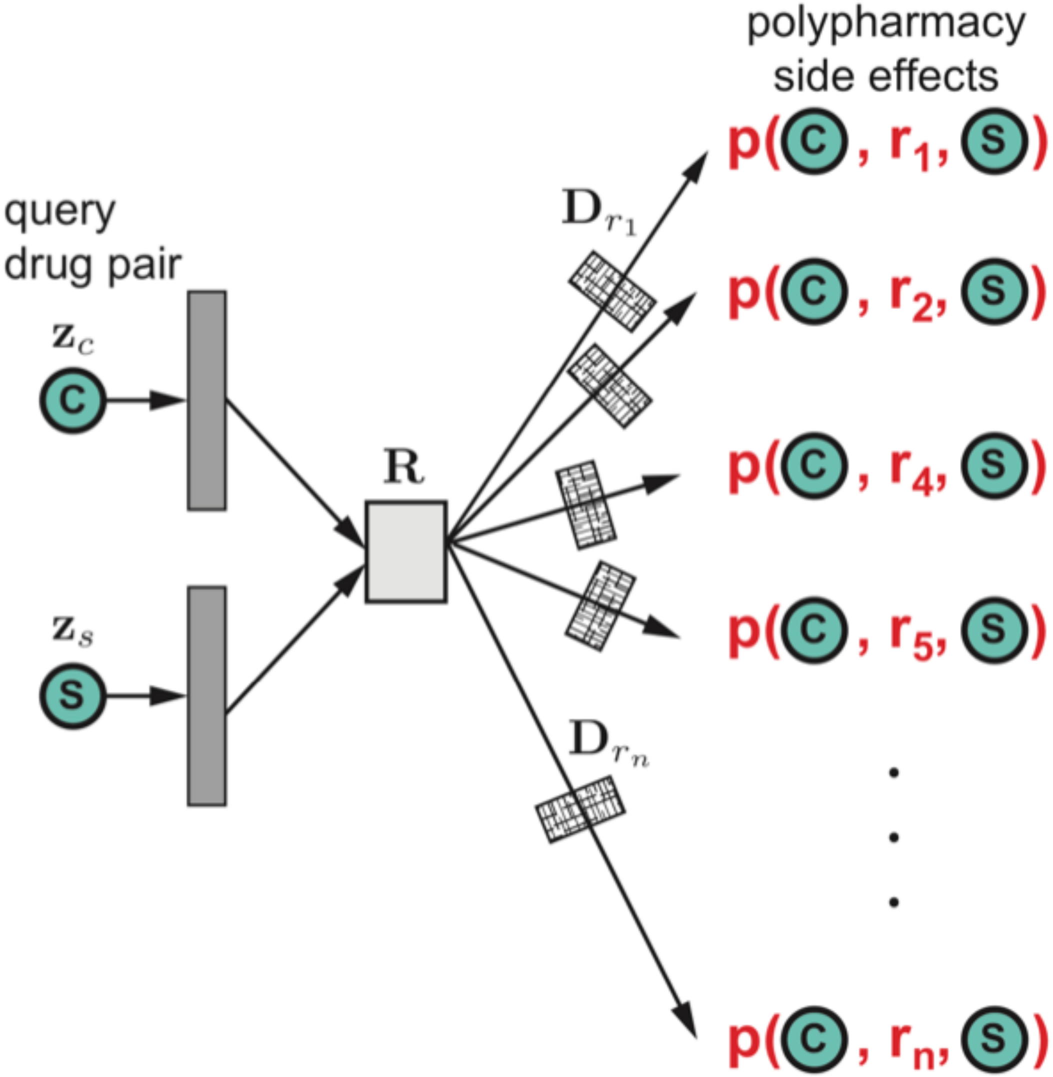
●—● Drug target interaction

●—● Physical protein binding



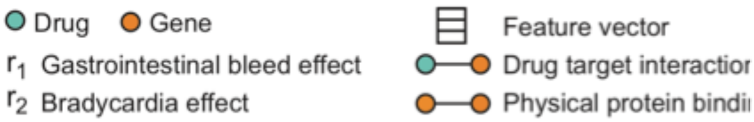
Neural Architecture: Decoder



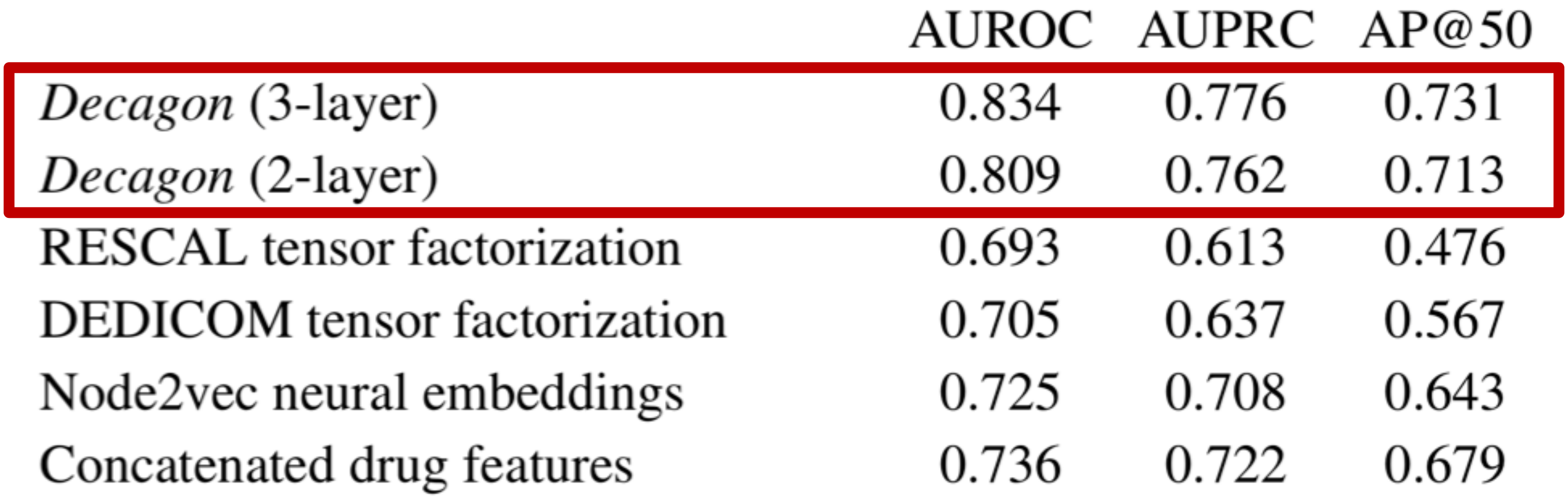


Graph decoder:

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



Prediction Performance



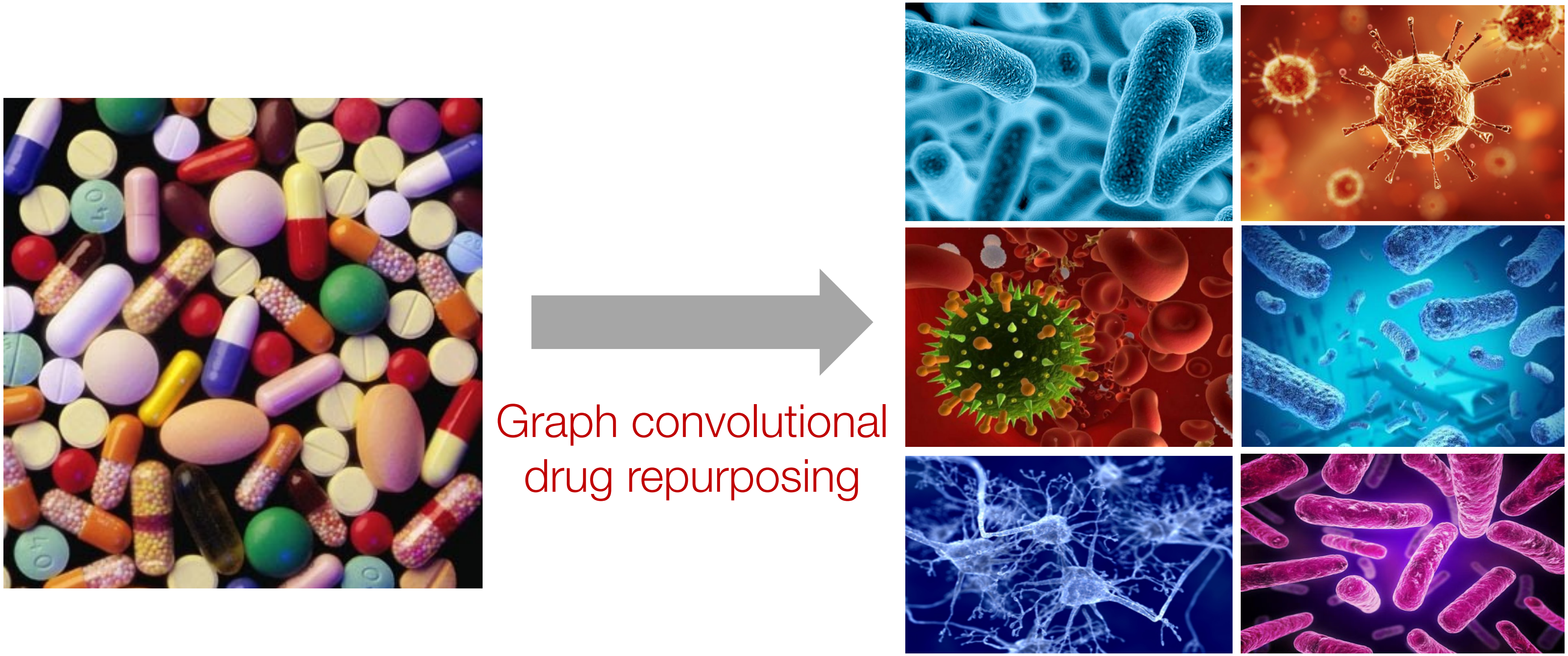
	AUROC	AUPRC	AP@50
Decagon (3-layer)	0.834	0.776	0.731
Decagon (2-layer)	0.809	0.762	0.713
RESCAL tensor factorization	0.693	0.613	0.476
DEDICOM tensor factorization	0.705	0.637	0.567
Node2vec neural embeddings	0.725	0.708	0.643
Concatenated drug features	0.736	0.722	0.679

- 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

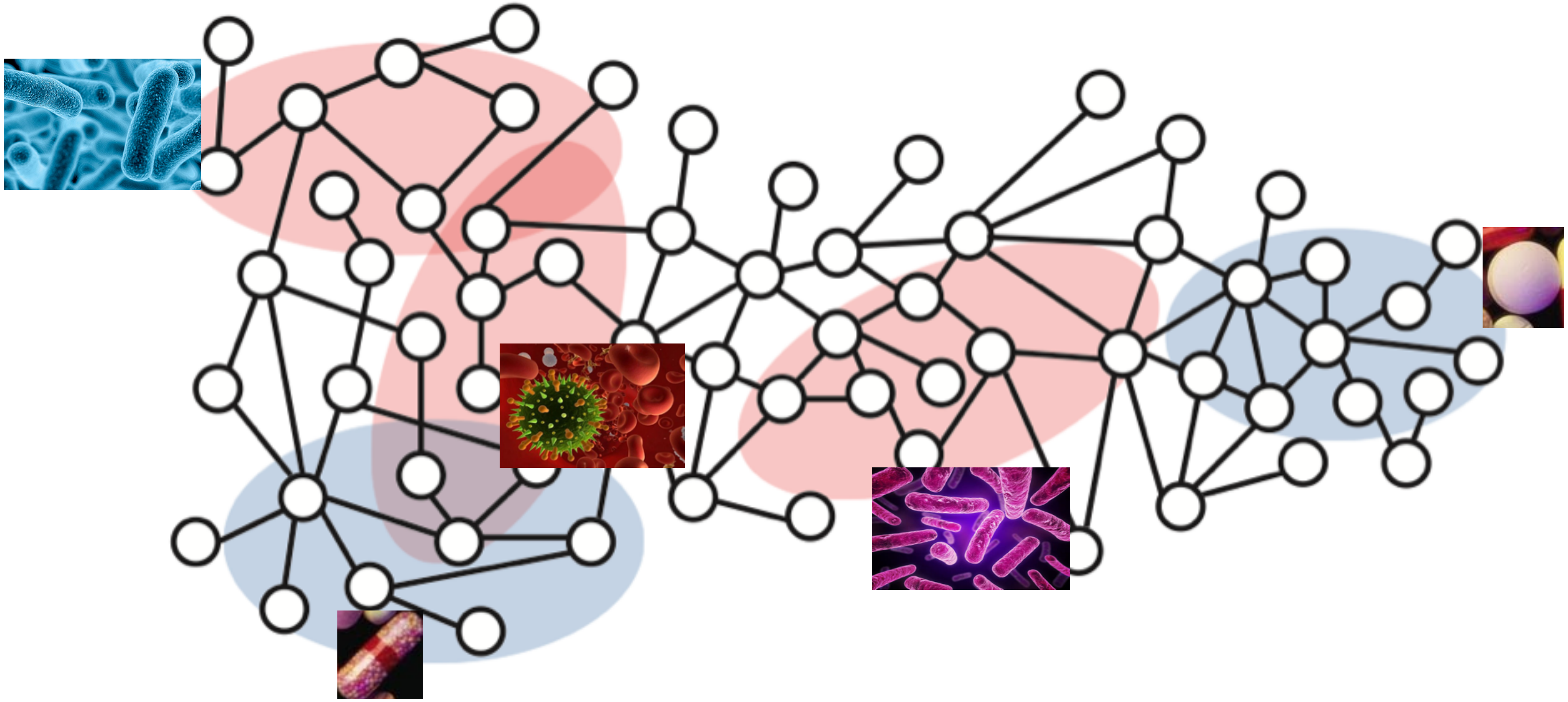
Prediction Problem



Graph convolutional drug repurposing

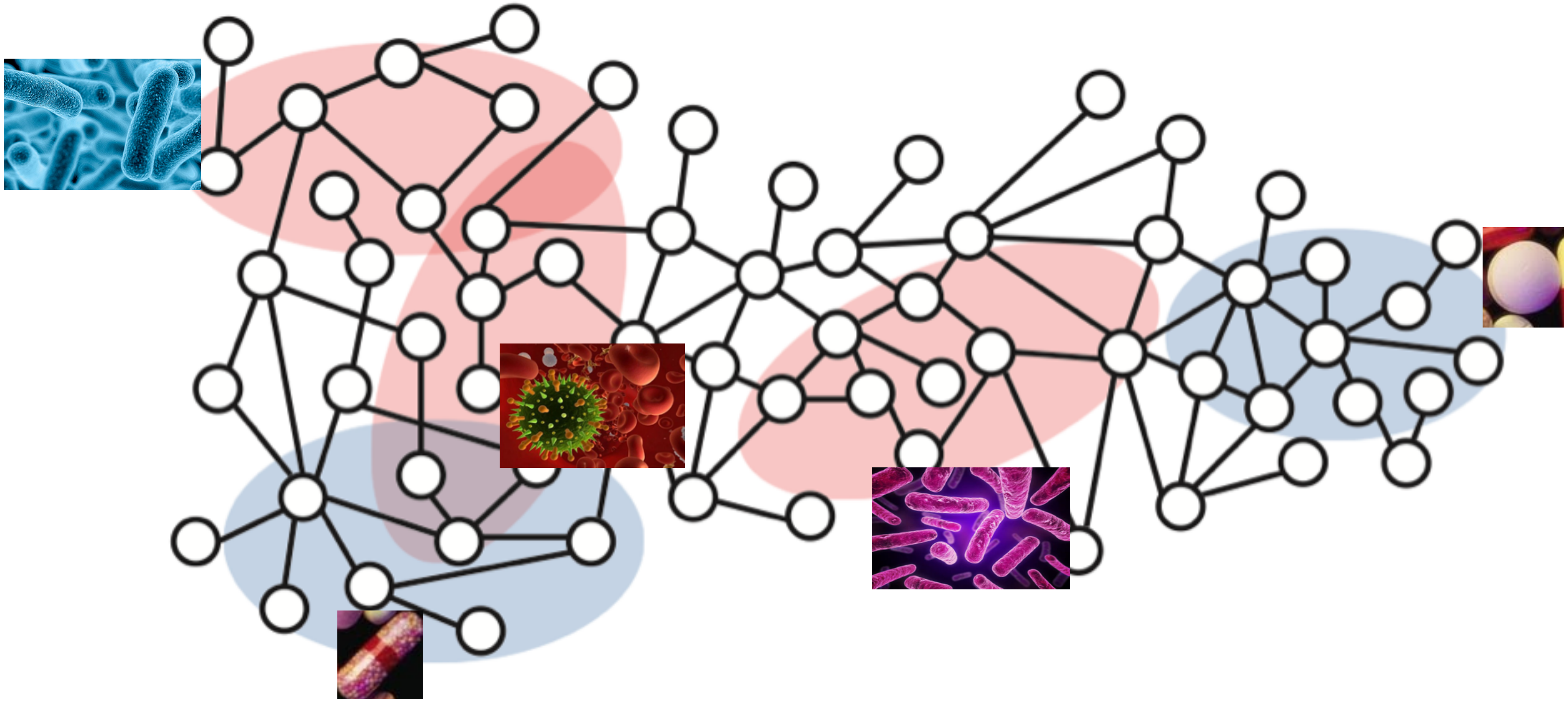
Goal: Predict which diseases a new drug (molecule) could treat

Insight: Networks



- Subgraphs of disease-associated proteins
- Subgraphs of drug target proteins

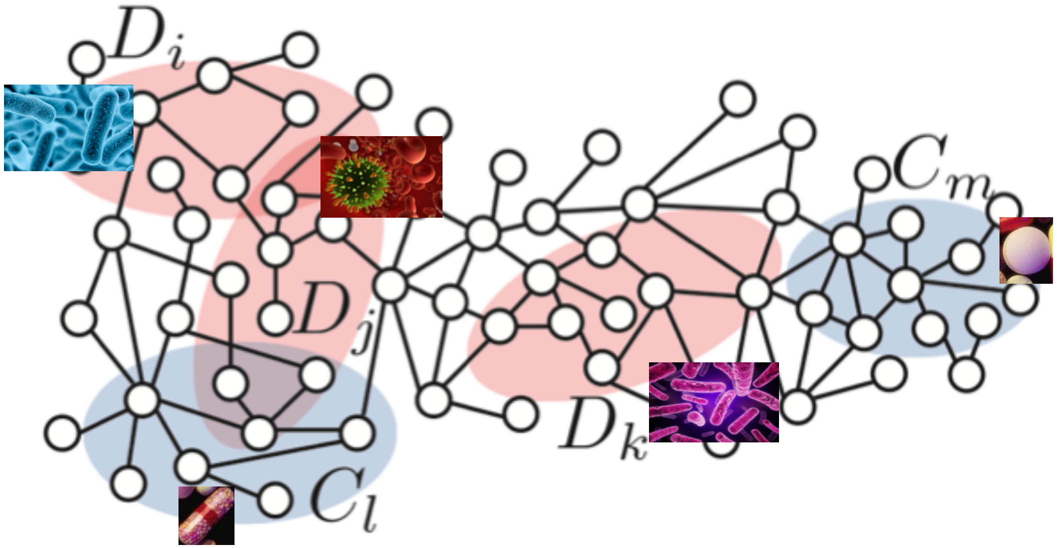
A Rationale for Graphs



A drug is likely to treat a disease if they are nearby in "pharmacological space"

[Menche et al. *Science* 2015; Guney et al. *Nat Commun* 2016; Hodos et al. *Systems Biology and Medicine* 2016]

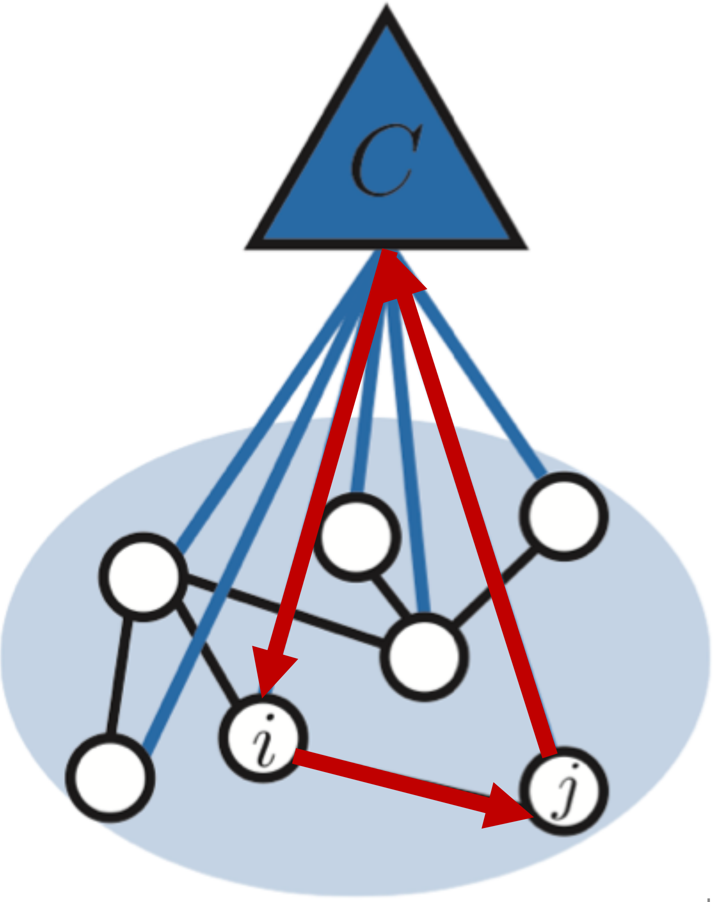
Link Prediction on Subgraphs



- Drug repurposing: Link prediction problem on subgraphs
- Predict new indications:
  - Obtain subgraphs by projecting drug and disease on the graph
  - Predict links between subgraphs

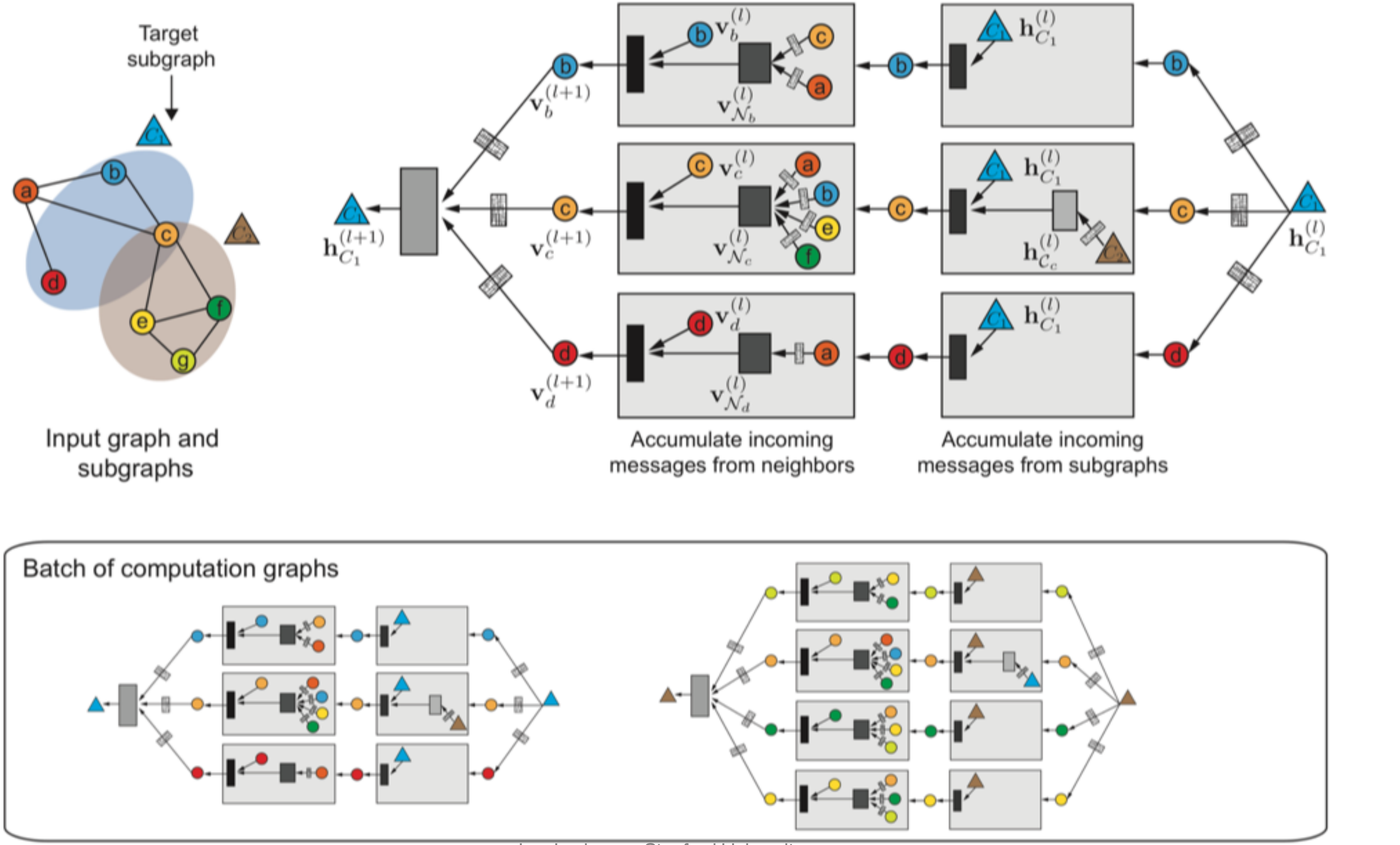
SUGAR: Message Passing

Embedding for subgraph C:



$$\begin{tikzcd}
\mu_{C\rightarrow i} \arrow[d] \\
\mu_{i\rightarrow j} \arrow[d] \\
\mu_{j\rightarrow C}
\end{tikzcd}$$

Neural Network Model


Target subgraph
C_1
C_2
a
b
c
d
e
f
g
Input graph and subgraphs
$\mathbf{h}_{C_1}^{(l+1)}$
$\mathbf{v}_b^{(l+1)}$
$\mathbf{v}_c^{(l+1)}$
$\mathbf{v}_d^{(l+1)}$
$\mathbf{v}_b^{(l)}$
$\mathbf{v}_{\mathcal{N}_b}^{(l)}$
$\mathbf{v}_c^{(l)}$
$\mathbf{v}_{\mathcal{N}_c}^{(l)}$
$\mathbf{v}_d^{(l)}$
$\mathbf{v}_{\mathcal{N}_d}^{(l)}$
$\mathbf{h}_{C_1}^{(l)}$
$\mathbf{h}_{C_c}^{(l)}$
Accumulate incoming messages from neighbors
Accumulate incoming messages from subgraphs
Batch of computation graphs

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

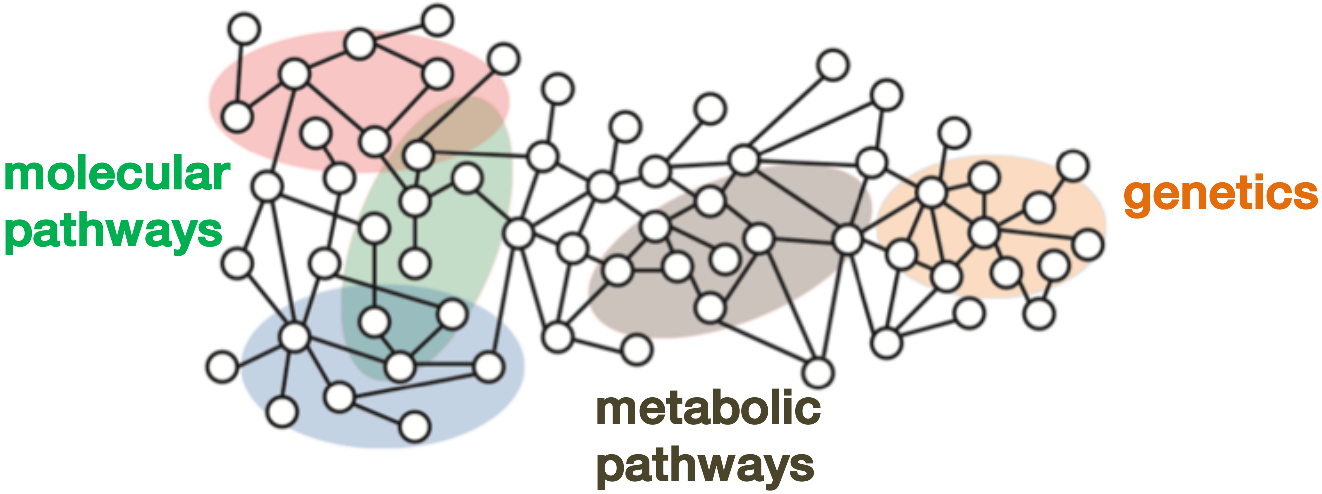
Approach	AUPRC	AUROC
SUGAR	0.851	0.888
Graphlet representation	0.606	0.689
Bi-directional random walk (MBiRW)	0.689	0.692
Heterogeneous graph inference (TL_HGBI)	0.673	0.664
PREdicting Drug IndiCaTions (PREDICT)	0.699	0.680
Drug-disease network closeness (r_c)	0.545	0.631
Drug-disease network dispersion (r_d)	0.692	0.623
Gene-based drug-disease network overlap (r_o)	0.512	0.548

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

Integrating Side Information

Including additional biomedical knowledge:



Metabolic pathways	Molecular functions	Biological processes	Cellular components	AUPRC	AUROC
				0.851	0.888
✓				0.869	0.893
✓	✓			0.874	0.912
✓	✓	✓		0.893	0.912
✓	✓	✓	✓	0.901	0.928

Drug Repurposing @ SPARK

Drug	Disease
N-acetyl-cysteine	cystic fibrosis
Xamoterol	neurodegeneration
Plerixafor	cancer
Sodium selenite	cancer
Ebselen	**C difficile**
Itraconazole	cancer
Bestatin	lymphedema
Bestatin	pulmonary arterial hypertension
Ketaprofen	lymphedema
Sildenafil	lymphatic malformation
Tacrolimus	pulmonary arterial hypertension
Benzamil	psoriasis
Carvedilol	Chagas’ disease
Benserazide	BRCA1 cancer
Pioglitazone	interstitial cystitis
Sirolimus	dystrophic epidermolysis bullosa

Given C difficile, where does Ebselen rank among all approved drugs?

Stanford MEDICINE | SPARK Translational Research Program

From Bench to Bedside

SUGAR’s Predictions

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

Higher rank is better

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

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

Conclusion

Next steps:

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

PhD Students

Claire Donnat

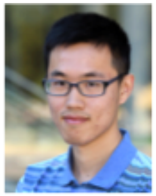
Mitchell Gordon

David Hallac

Emma Pierson

Geet Sethi

Himabindu Lakkaraju


Rex Ying

Tim Althoff

Will Hamilton

Post-Doctoral Fellows

David Jurgens

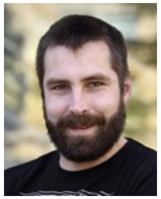
Marinka Zitnik

Michele Catasta

Srijan Kumar

Stephen Bach

Research Staff


Peter Kacin

Rok Sosic

Industry Partnerships

Ford
azumio
facebook
twitter
Pinterest
UNDER ARMOUR
LinkedIn
HYUNDAI
BOEING
Wikipedia The Free Encyclopedia
NVIDIA
DISQUS
OpenDoor REAL ESTATE
spinn3r
HUAWEI
SAP
BOSCH Invented for life
CRISIS TEXT LINE |

Funding

NIH
IARPA
DARPA
NSF
MURI ARO
CHAN ZUCKERBERG INITIATIVE

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
Sendhill Mullainathan, Economics, Harvard University
Scott Delp, Bioengineering, Stanford University
Jens Ludwig, Harris Public Policy, University of Chicago

STANFORD INFOLAB

References

- node2vec: Scalable Feature Learning for Networks
  A. Grover, J. Leskovec. KDD 2016.

- Predicting multicellular function through multi-layer tissue networks. M. Zitnik, J. Leskovec. Bioinformatics, 2017.

- 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.
  IEEE Data Engineering Bulletin, 2017.

- Modeling polypharmacy side effects with graph convolutional networks. M. Zitnik, M. Agrawal, J. Leskovec. BioArxiv, 2017.

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