# Applications of Graph Neural Networks

CS224W: Analysis of Networks

Jure Leskovec, R. Ying and J. You, Stanford University

http://cs224w.stanford.edu

# Outline of Today's Lecture

**Three topics for today:**

1. **GNN recommendation (PinSage)** ☜

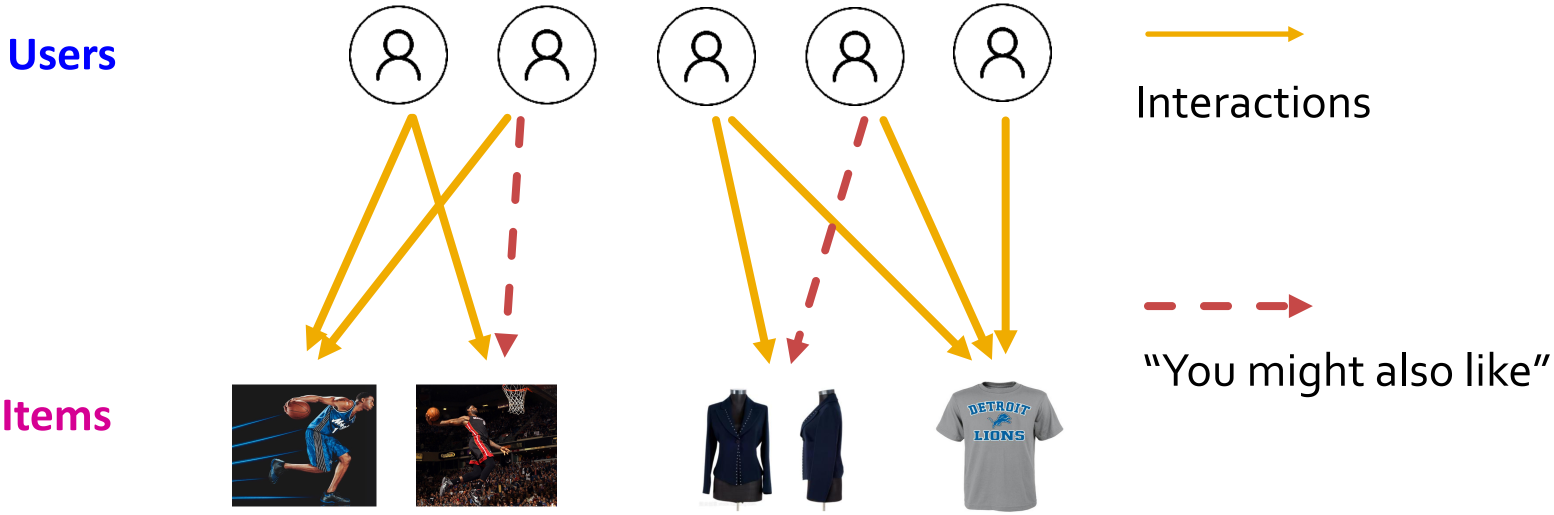
2. **Heterogeneous GNN (Decagon)**

3. **Goal-directed generation (GCPN)**

# PinSAGE: GNN for Recommender Systems

# Recommender Systems

- **Users interacts with items**
  - Watch movies, buy merchandise, listen to music
- **Goal: Recommend items users might like**
  - Customer X buys Metallica and Megadeth CDs
  - Customer Y buys Megadeth, the recommender system suggests Metallica as well



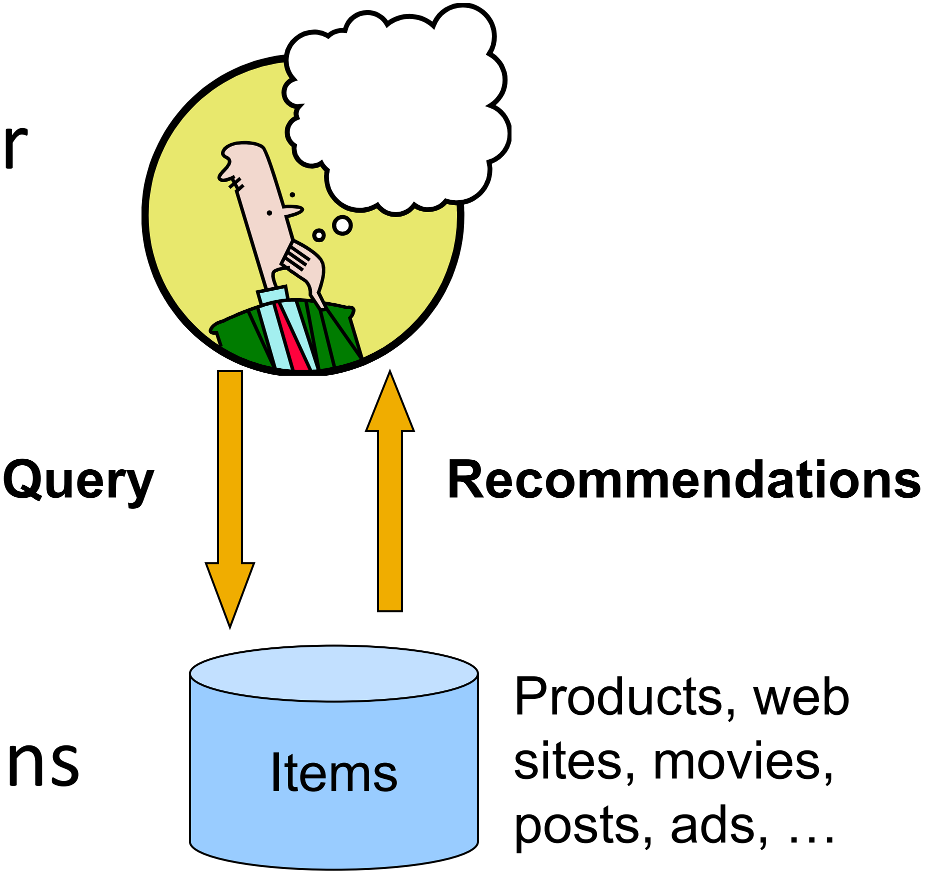
# Recommender Systems

## Goal: Learn what items are related

- For a given query item(s) Q, return a set of **similar** items that we recommend to the user

## Idea:

- User interacts with a set of items
- Formulate a query Q
- Search the items and return recommendations



Query

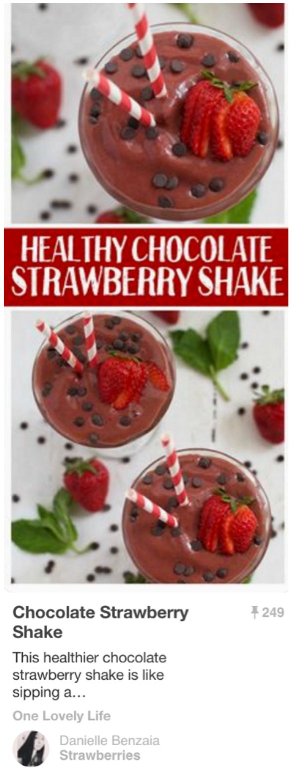
Recommendations

Items

Products, web sites, movies, posts, ads, …

# Example: Pinterest

Query:



Chocolate Strawberry Shake

249

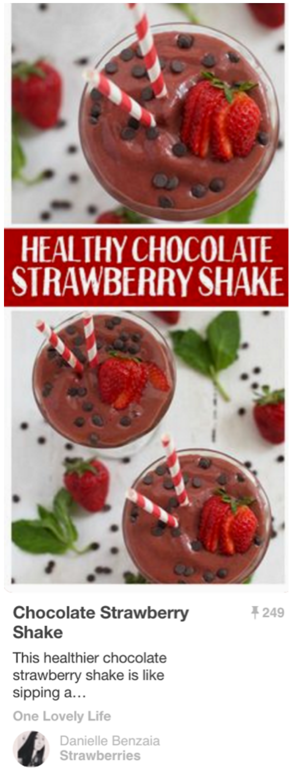
This healthier chocolate strawberry shake is like sipping a…

One Lovely Life

Danielle Benzaia
Strawberries

# Example: Pinterest

Query:



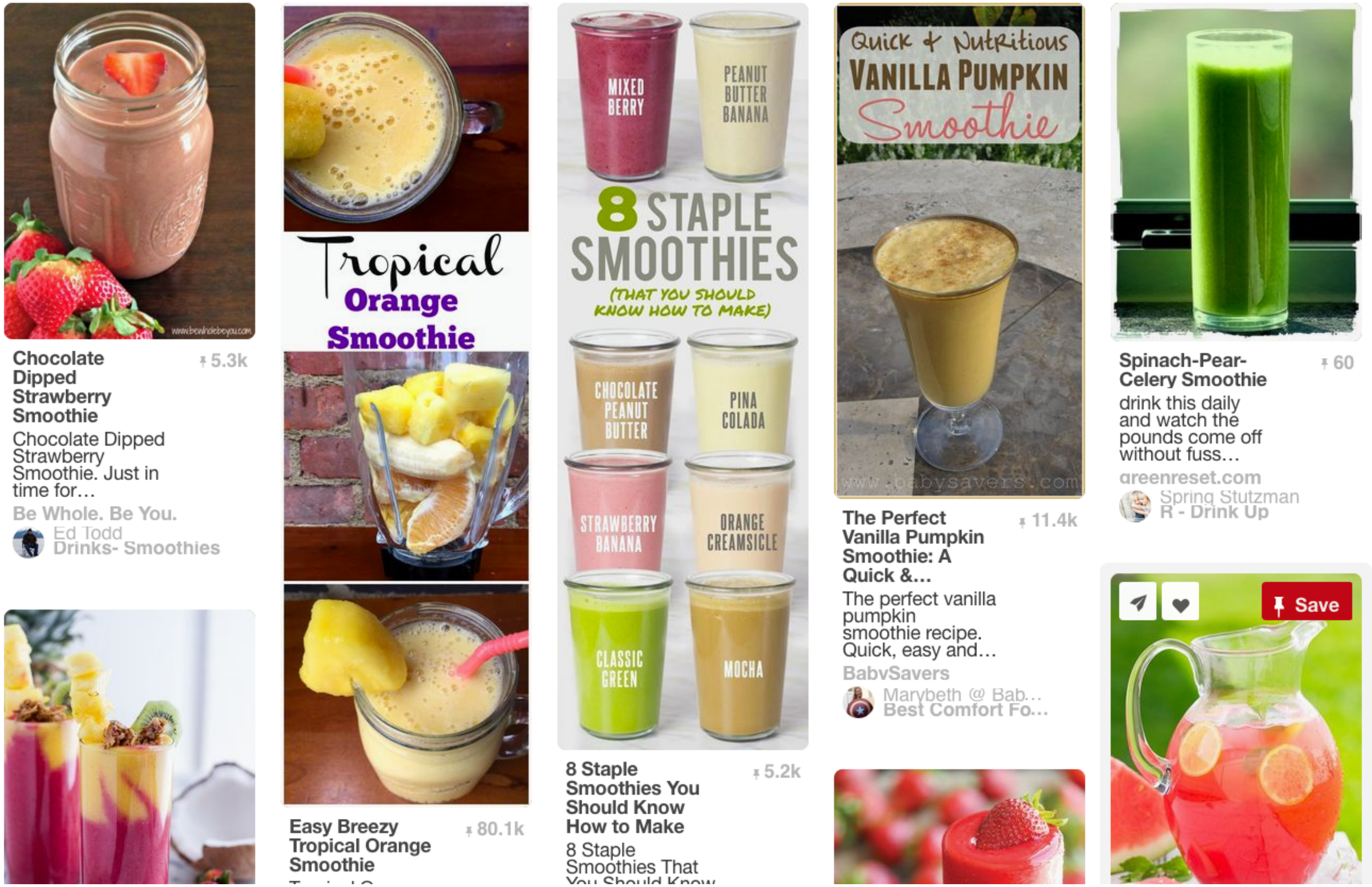
Chocolate Strawberry Shake 249

This healthier chocolate strawberry shake is like sipping a…

One Lovely Life

Danielle Benzaia
Strawberries

Recommendations:



Chocolate Dipped Strawberry Smoothie 5.3k

Chocolate Dipped Strawberry Smoothie. Just in time for…

Be Whole. Be You.

Ed Todd
Drinks- Smoothies

Easy Breezy Tropical Orange Smoothie 80.1k

8 Staple Smoothies You Should Know How to Make 5.2k

8 Staple Smoothies That You Should Know

The Perfect Vanilla Pumpkin Smoothie: A Quick &… 11.4k

The perfect vanilla pumpkin smoothie recipe. Quick, easy and…

BabySavers

Marybeth @ Bab…
Best Comfort Fo…

Spinach-Pear-Celery Smoothie 60

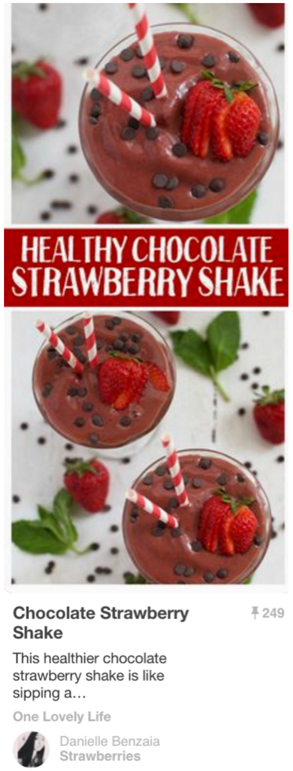
drink this daily and watch the pounds come off without fuss…

greenreset.com

Spring Stutzman
R - Drink Up

# Example (2): Pinterest

Query:

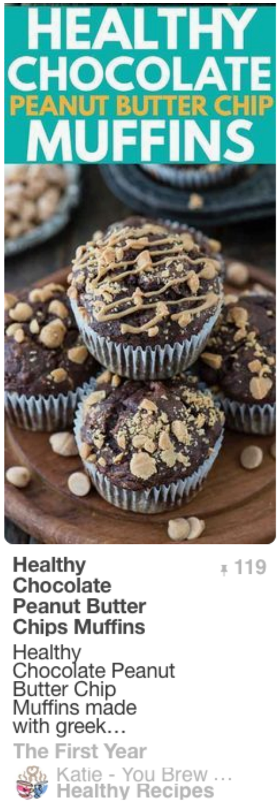


**Chocolate Strawberry Shake** 249

This healthier chocolate strawberry shake is like sipping a…

One Lovely Life

Danielle Benzaia
Strawberries

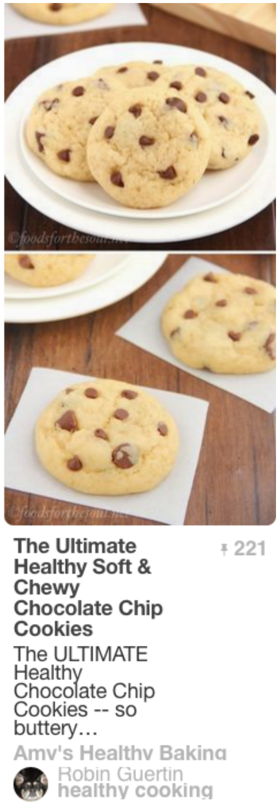


**Healthy Chocolate Peanut Butter Chips Muffins** 119

Healthy Chocolate Peanut Butter Chip Muffins made with greek…

The First Year

Katie - You Brew …
Healthy Recipes



**The Ultimate Healthy Soft & Chewy Chocolate Chip Cookies** 221

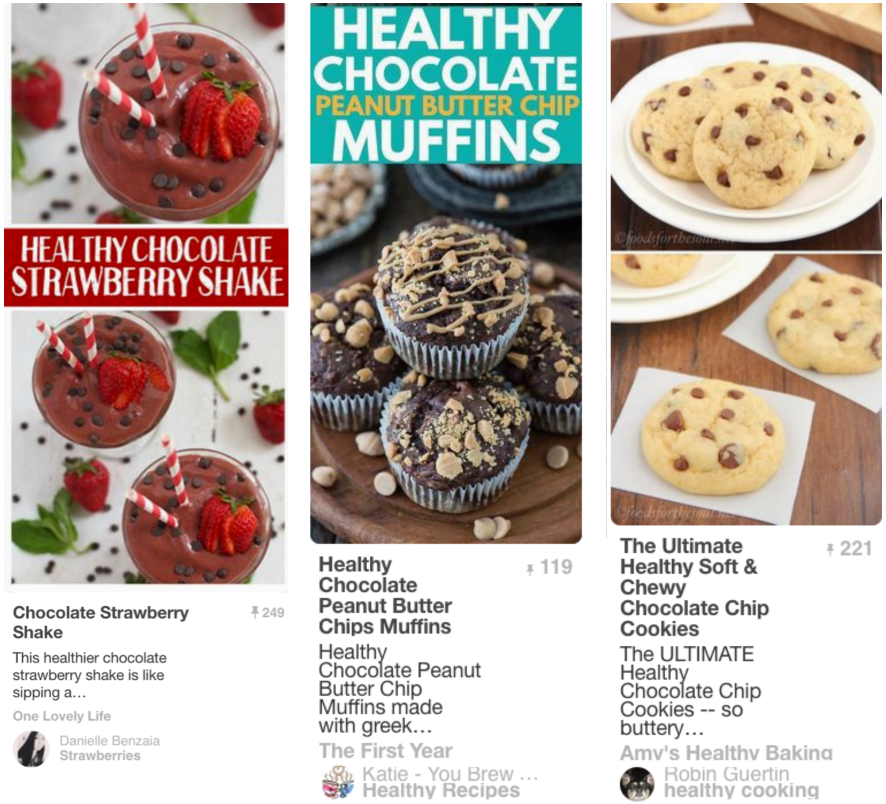
The ULTIMATE Healthy Chocolate Chip Cookies -- so buttery…

Amy's Healthy Baking

Robin Guertin
healthy cooking

# Example (2): Pinterest

Query:



Chocolate Strawberry Shake 249

This healthier chocolate strawberry shake is like sipping a…

One Lovely Life

Danielle Benzaia
Strawberries

Healthy Chocolate Peanut Butter Chips Muffins 119

Healthy Chocolate Peanut Butter Chip Muffins made with greek…

The First Year

Katie - You Brew …
Healthy Recipes

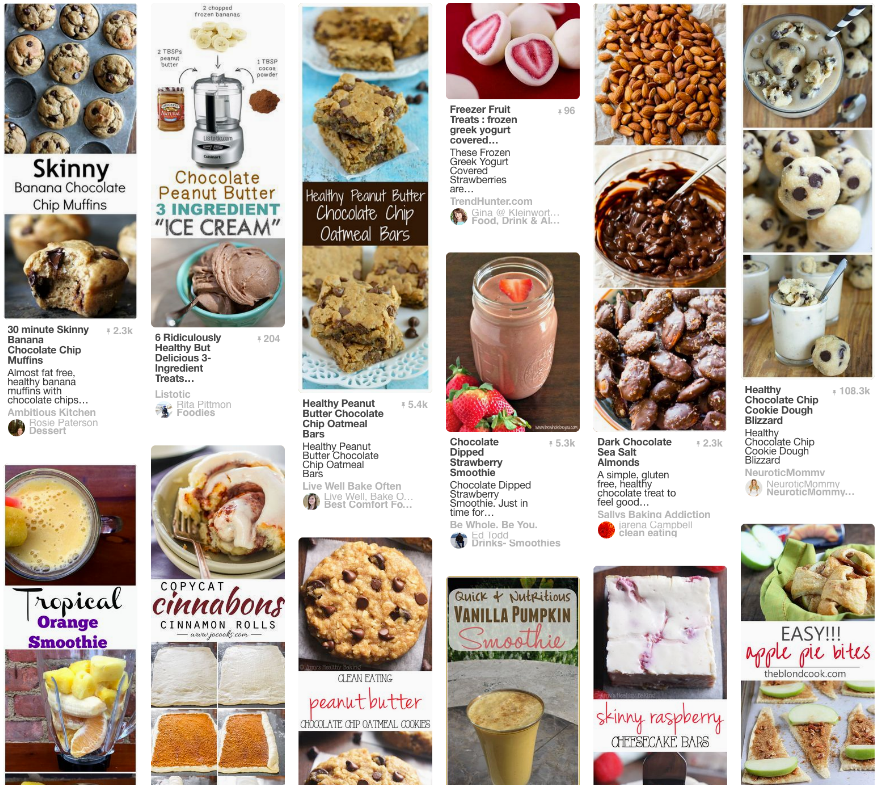
The Ultimate Healthy Soft & Chewy Chocolate Chip Cookies 221

The ULTIMATE Healthy Chocolate Chip Cookies -- so buttery…

Amy's Healthy Baking

Robin Guertin
healthy cooking

Recommendations:



30 minute Skinny Banana Chocolate Chip Muffins 2.3k

Almost fat free, healthy banana muffins with chocolate chips…

Ambitious Kitchen

Rosie Paterson
Dessert

6 Ridiculously Healthy But Delicious 3-Ingredient Treats… 204

Listotic

Rita Pittmon
Foodies

Healthy Peanut Butter Chocolate Chip Oatmeal Bars 5.4k

Healthy Peanut Butter Chocolate Chip Oatmeal Bars

Live Well Bake Often

Live Well, Bake O…
Best Comfort Fo…

Freezer Fruit Treats : frozen greek yogurt covered… 96

These Frozen Greek Yogurt Covered Strawberries are…

TrendHunter.com

Gina @ Kleinwort…
Food, Drink & Al…

Chocolate Dipped Strawberry Smoothie 5.3k

Chocolate Dipped Strawberry Smoothie. Just in time for…

Be Whole. Be You.

Ed Todd
Drinks- Smoothies

Dark Chocolate Sea Salt Almonds 2.3k

A simple, gluten free, healthy chocolate treat to feel good…

Sallys Baking Addiction

jarena Campbell
clean eating

Healthy Chocolate Chip Cookie Dough Blizzard 108.3k

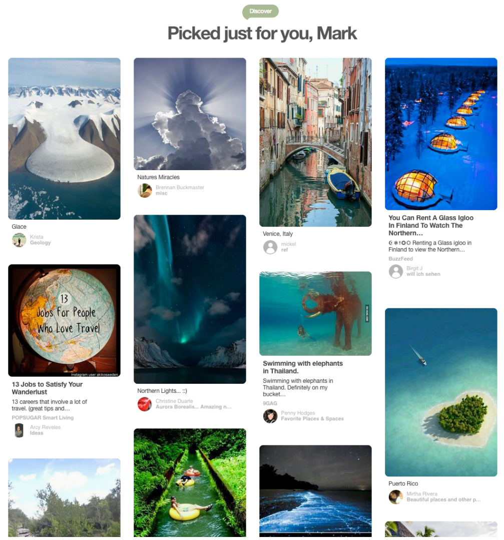
Healthy Chocolate Chip Cookie Dough Blizzard

NeuroticMommy

NeuroticMommy
NeuroticMommy…

# Many Applications

## Having a universal similarity function allows for many applications:



Picked just for you, Mark

Glace
Krista
Geology

Natures Miracles
Brennan Buckmaster
misc

Venice, Italy
mickel
ref

You Can Rent A Glass Igloo In Finland To Watch The Northern…
Renting a Glass igloo in Finland to view the Northern…
BuzzFeed
Birgit J
will ich sehen

13 Jobs For People Who Love Travel

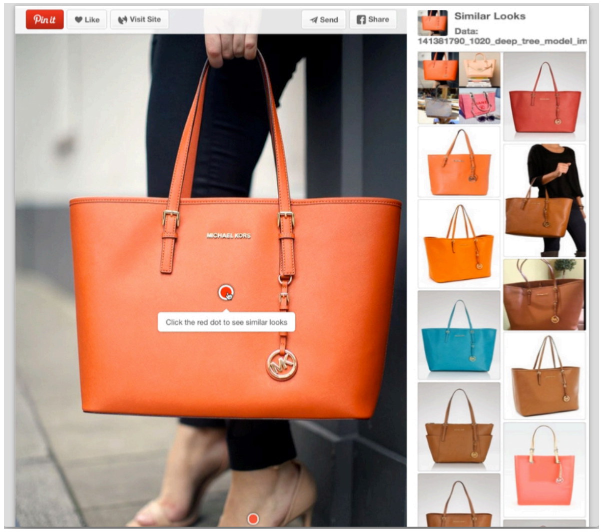
13 Jobs to Satisfy Your Wanderlust
13 careers that involve a lot of travel. (great tips and…
POPSUGAR Smart Living
Arcy Reveles
Ideas

Northern Lights... :)
Christine Duarte
Aurora Borealis... Amazing n…

Swimming with elephants in Thailand.
Swimming with elephants in Thailand. Definitely on my bucket…
9GAG
Penny Hodges
Favorite Places & Spaces

Puerto Rico
Mirtha Rivera
Beautiful places and other p…

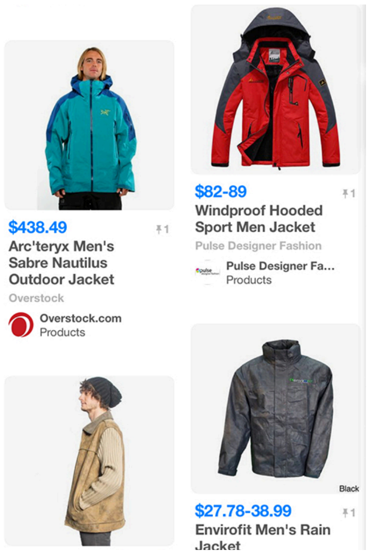
**Homefeed**
(endless feed of recommendations)



Similar Looks
Data:
141381790_1020_deep_tree_model_im

Click the red dot to see similar looks

**Related pins**
(find most similar/related pins)



$438.49
Arc'teryx Men's Sabre Nautilus Outdoor Jacket
Overstock
Overstock.com
Products

$82-89
Windproof Hooded Sport Men Jacket
Pulse Designer Fashion
Pulse Designer Fa…
Products

$27.78-38.99
Envirofit Men's Rain Jacket

**Ads and shopping**
(use organic for the query and search the ads database)

# Key Problem: Defining Similarity

**Question: How do we define similarity?**

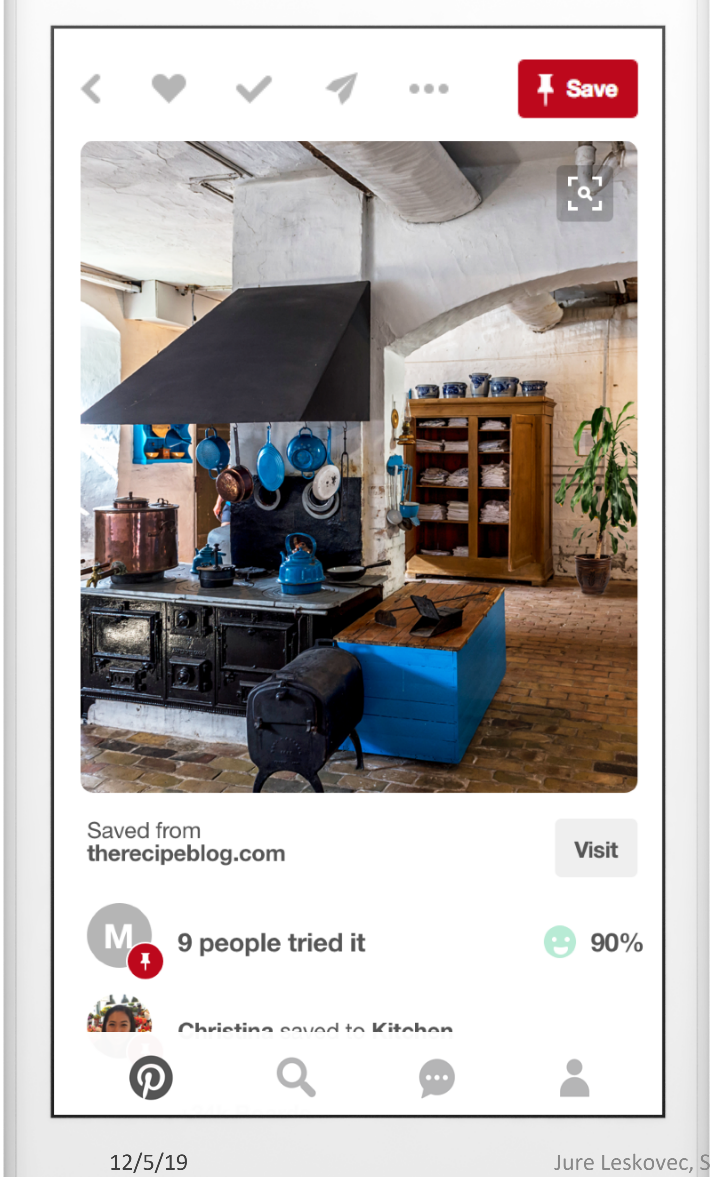
- **1) Content-based:** User and item features, in the form of images, text, categories, etc.
- **2) Graph-based:** User-item interactions, in the form of graph/network structure
  - This is called **collaborative filtering:**
    - For a given user X, find others who liked similar items
    - Estimate what X will like based on what similar others like

# Key Problems

## How do we define similarity:

- **(1) Gathering "known" similarities**
    - How to collect the data about what users like
- **(2) Extrapolating unknown similarities from the known ones**
    - Mainly interested in high unknown similarities
        - We are not interested in knowing what you don't like but what you like
- **(3) Evaluating methods**
    - How to measure success/performance of recommendation methods

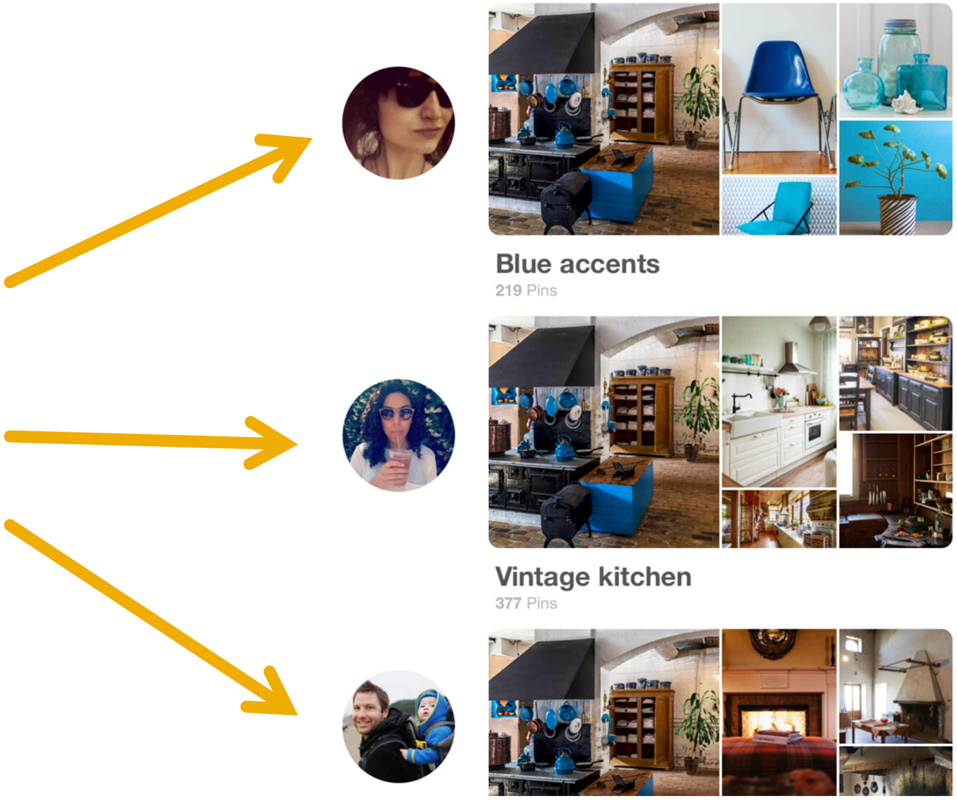
# Pinterest



Saved from
therecipeblog.com

9 people tried it 90%

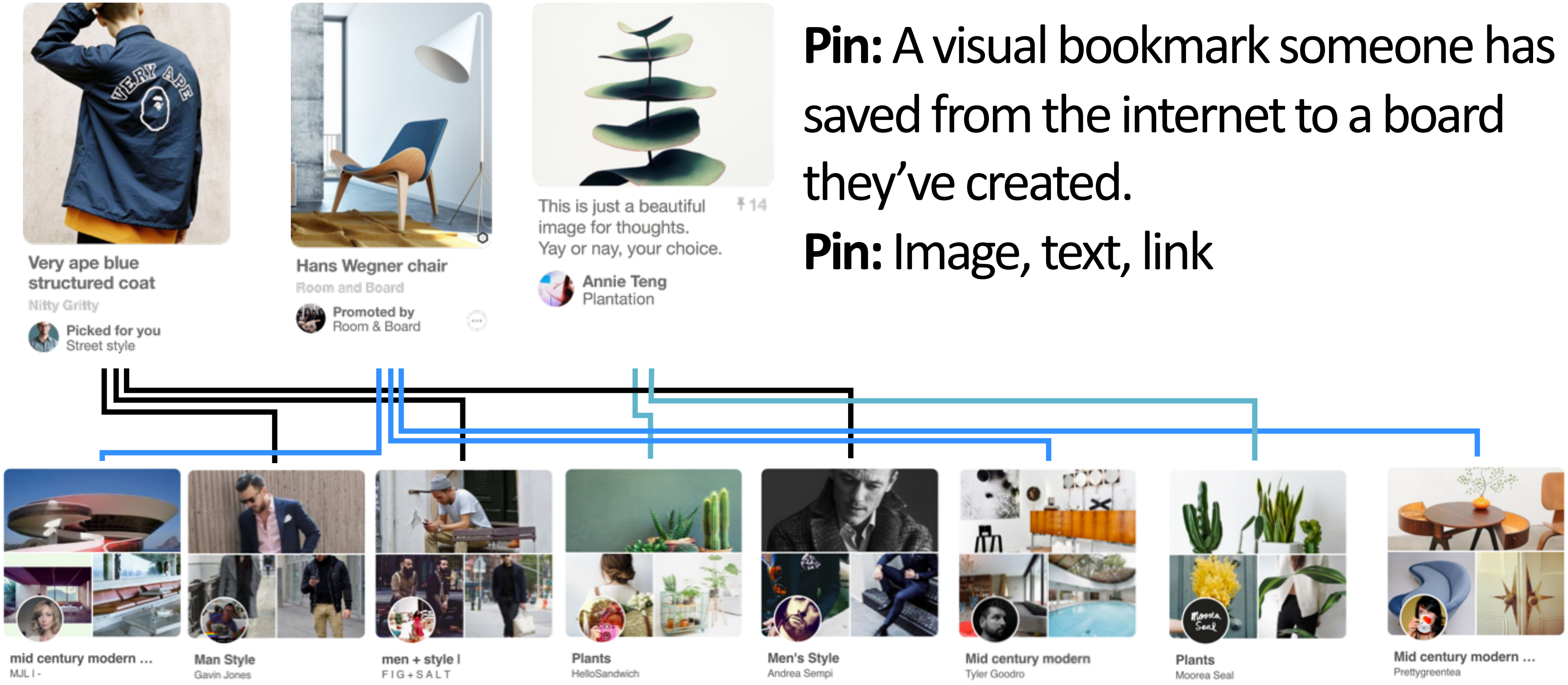
Christina saved to Kitchen



- 300M users
- 4+B pins, 2+B boards

# Pinterest

## Pinterest: Human curated collection of pins

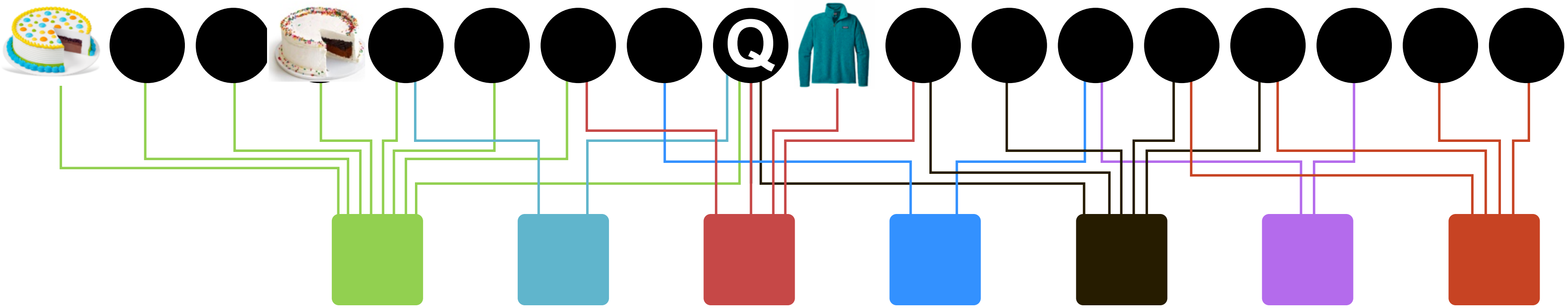


**Pin:** A visual bookmark someone has saved from the internet to a board they've created.

**Pin:** Image, text, link

**Board:** A collection of ideas (pins having something in common)

# Pinterest: 2 Sources of Signal



**Two sources of signal:**

**Features:**

- Image and text of each pin

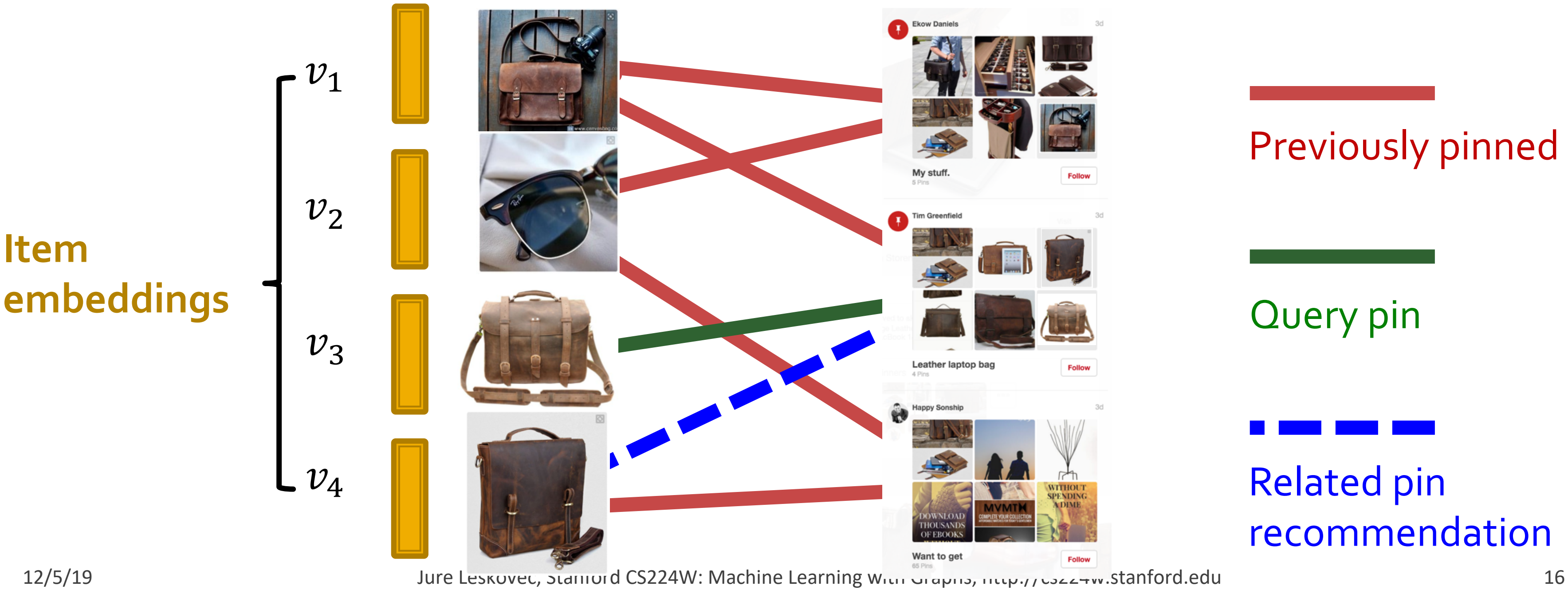
**Graph:**

- Graph is dynamic: Need to apply to new nodes without model retraining

# Recommendations via Embeddings

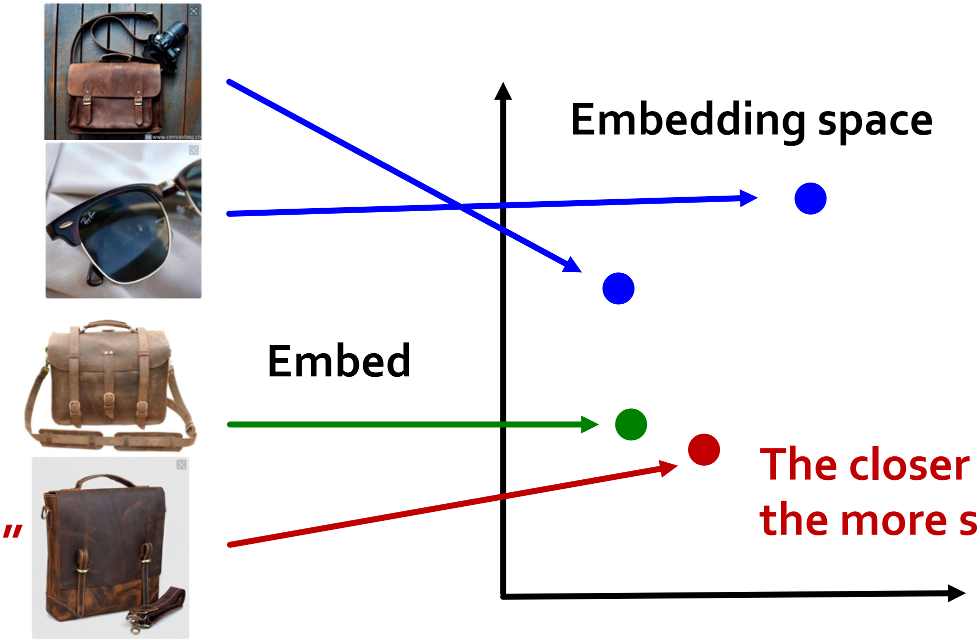
## Goal: Learn embeddings for items

- **Related Pins Query**: Which pin to recommend when a user interacts with a pin $v_3$?
- **Answer**: Find the closest embedding ($v_4$) to $v_3$ by nearest neighbor. Recommend it.



# Recommendations via Embeddings

- **Goal 1**: Efficiently learn embeddings for billions of pins (items, nodes) using neural networks
- **Goal 2**: Perform nearest neighbor query to recommend items in real-time



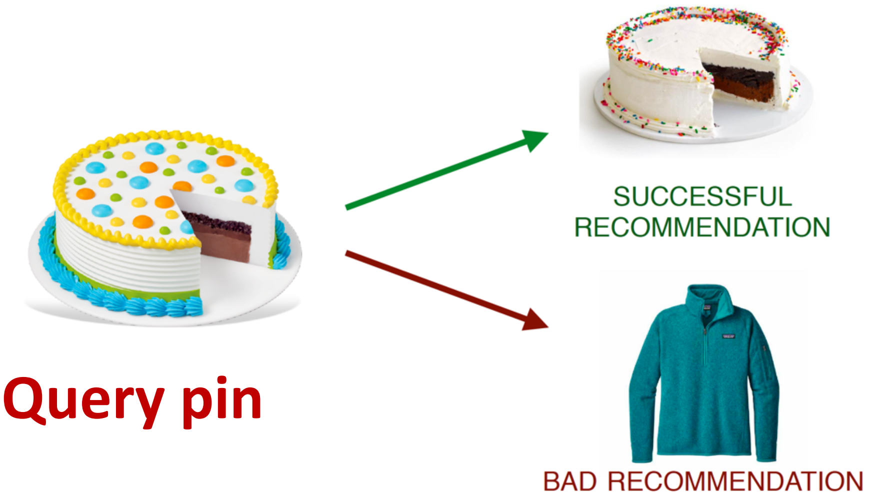
Query pin

"Predicted" related pin

The closer the embeddings are, the more similar the pins are

# Overview: Pin Recommendation

**Task:** Recommend related pins to users



SUCCESSFUL RECOMMENDATION

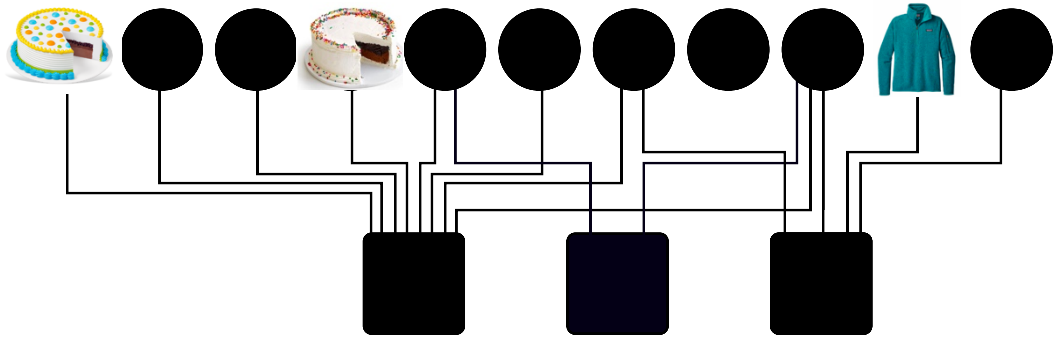
BAD RECOMMENDATION

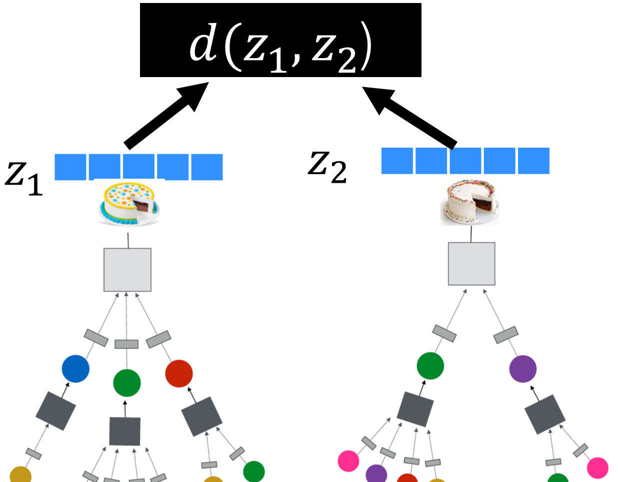
**Query pin**

**Task:** Learn node embeddings $z_i$ such that

$$d(z_{cake1}, z_{cake2}) < d(z_{cake1}, z_{sweater})$$

**Predict whether two nodes in a graph are related**





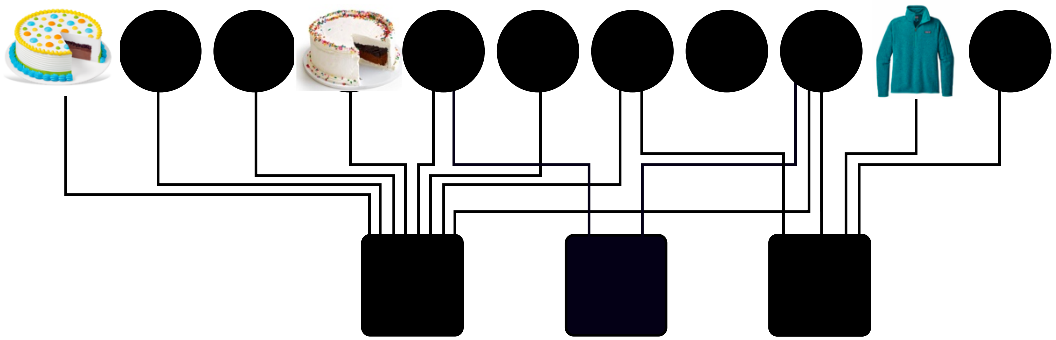
$d(z_1, z_2)$

$z_1$

$z_2$

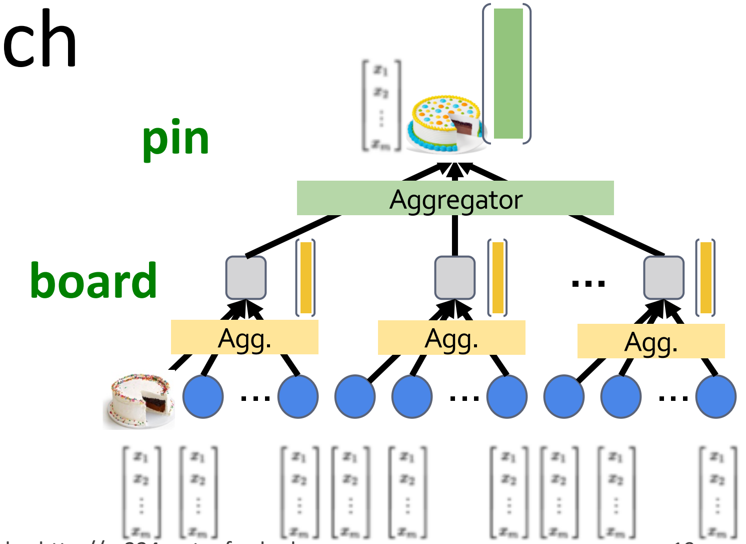
# PinSage: Graph Neural Networks

## Predict whether two nodes in a graph are related



## Approach:

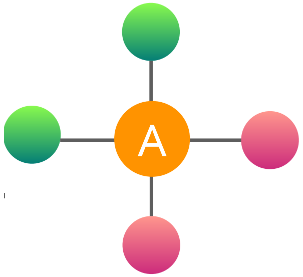
- Pins have embeddings at each layer
- Layer-0 embedding of a node are its features:
  - Text, image, …



[Ying et al., WWW 2018]

# PinSage: Why it Works

- PinSage graph convolutional network:
  - **Goal:** Generate embeddings for nodes (e.g., pins) in the Pinterest graph containing billions of objects
  - **Key Idea:** Borrow information from nearby nodes
    - E.g., bed rail Pin might look like a garden fence, but gates and beds are rarely adjacent in the graph



  - Pin embeddings are essential to many different tasks. Aside from the “Related Pins” task, it can also be used in:
    - Recommend related ads
    - Homefeed recommendation
    - Cluster users by their interest

# PinSage Pipeline

1. **Collect** billions of training pairs from logs.
   - **Positive pair:** Two pins that are **consecutively saved into the same board** within a time interval (1 hour)
   - **Negative pair:** A random pair of 2 pins
     - With high probability the pins are not on the same board

# PinSage Pipeline

1. **Collect** billions of training pairs from logs.
   - **Positive pair:** Two pins that are **consecutively saved into the same board** within a time interval (1 hour)
   - **Negative pair:** A random pair of 2 pins
     - With high probability the pins are not on the same board
2. **Train GNN** to generate similar embeddings for training pairs
3. **Inference**: Generate embeddings for all pins
4. **Nearest neighbor search** in embedding space to make recommendations.

# Training Objective Function

- Train so that **pins that are consecutively pinned have similar embeddings**
- **Max-margin loss:**

$$\mathcal{L} = \sum_{(u,v)\in \boxed{\mathcal{D}}} \max(0, -\boxed{\mathbf{z}_u^\top \mathbf{z}_v} + \mathbf{z}_u^\top \boxed{\mathbf{z}_n} + \boxed{\Delta})$$

set of training pairs from user logs

“positive”/true training pair

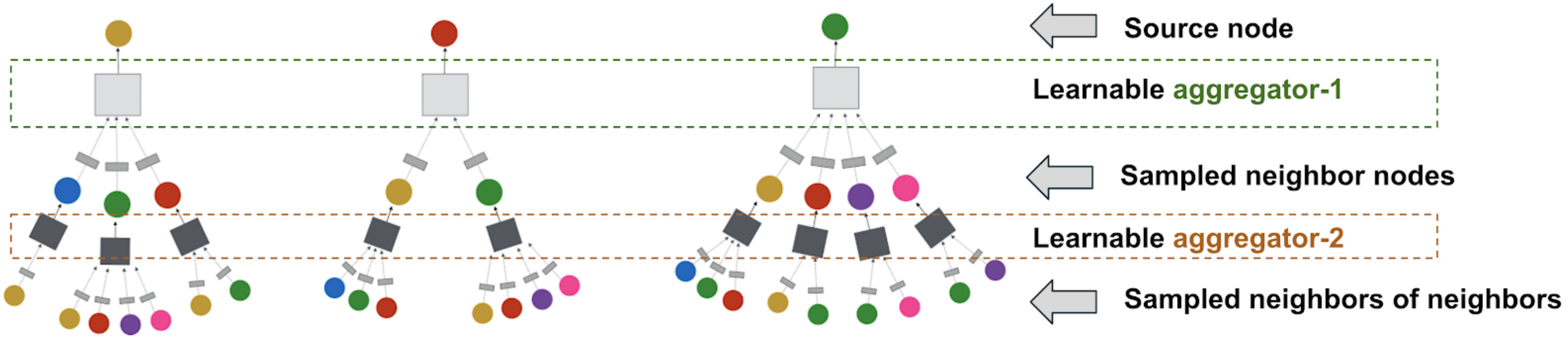
“negative” example

“margin” (i.e., how much larger positive pair similarity should be compared to negative)

# Key Innovation (1)

- **Four key innovations:**

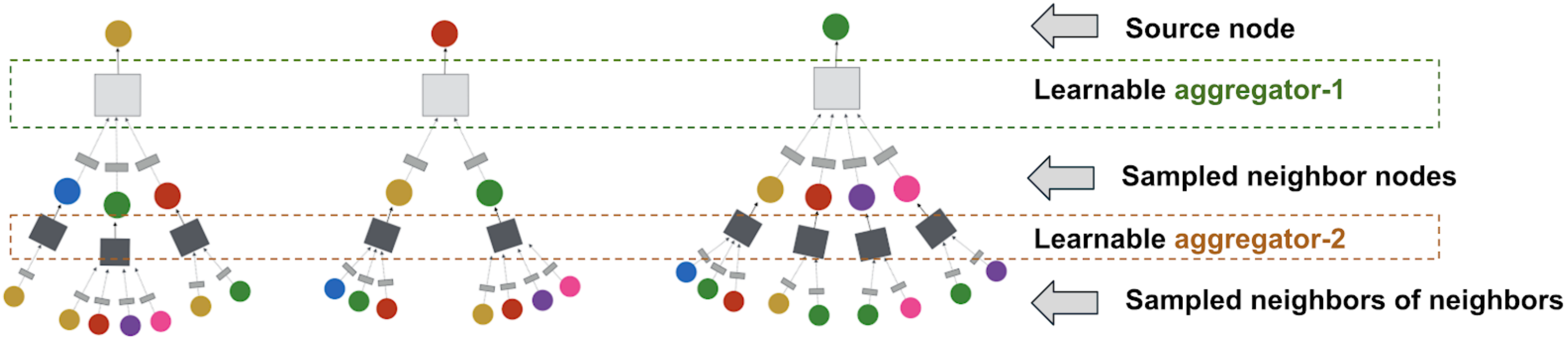
1. **On-the-fly graph convolutions**
    - Sample the neighborhood around a node and dynamically construct **a computation graph**



Minibatch of neighborhoods

# Key Innovation (1)

- **Four key innovations:**
  1. **On-the-fly graph convolutions**
     - Perform a **localized graph convolution** around a particular node
     - Does not need the entire graph during training



At every iteration, only source node embeddings are computed

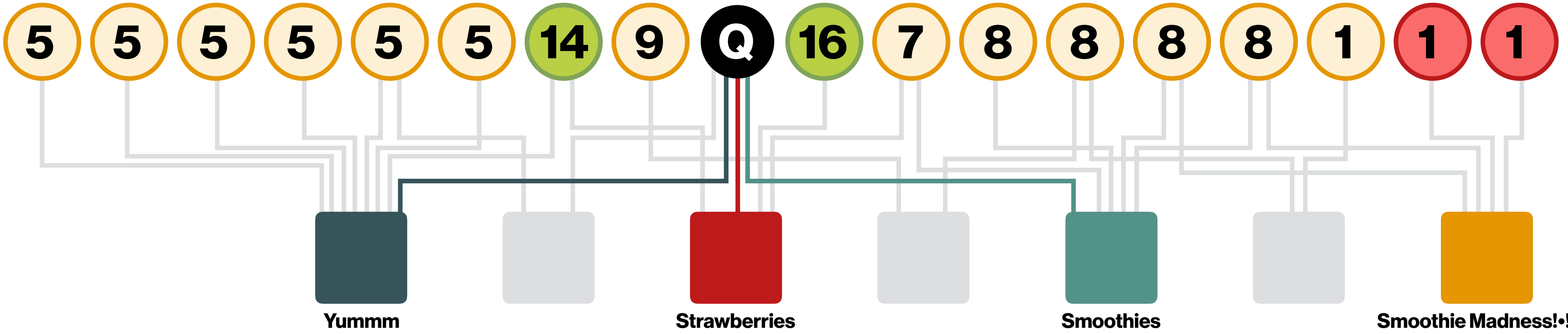
# Key Innovation (2)

- **Four key innovations:**
  2. **Selecting neighbors via random walks**
     - Performing aggregation on all neighbors is infeasible:
       - How to select the set of neighbors of a node to convolve over?
     - **Personalized PageRank can help!**
     - **Define Importance pooling**: Define importance-based neighborhoods by simulating random walks and selecting the neighbors with the highest visit counts

# Key Innovation (2): Random Walks

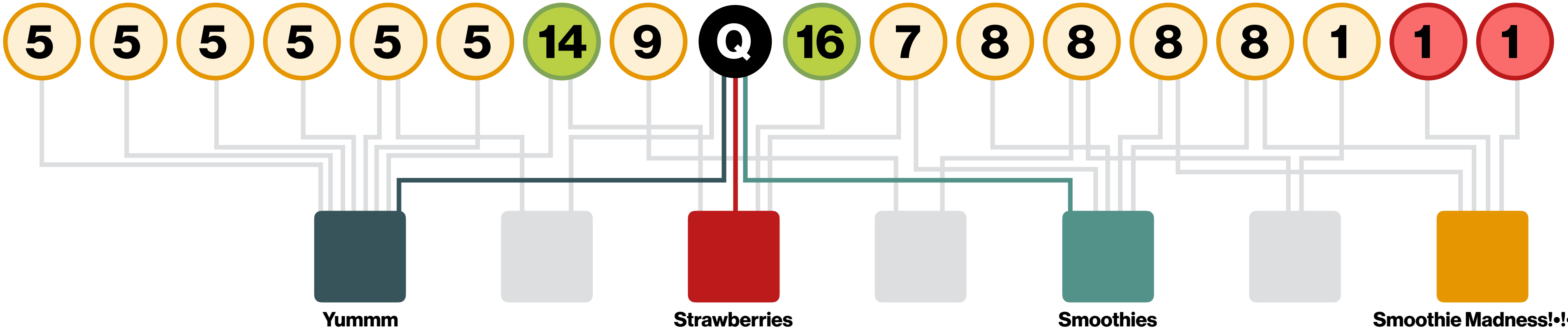
- Proximity to query node(s) ***Q***

```
ALPHA = 0.5
QUERY_NODES = { ● }
pin_node = QUERY_NODES.sample_by_weight()
for i in range(N_STEPS):
    board_node = pin_node.get_random_neighbor()
    pin_node = board_node.get_random_neighbor()
    pin_node.visit_count += 1
    if random() < ALPHA:
        pin_node = QUERY_NODES.sample_by_weight()
```



# Key Innovation (2): Random Walks

- Proximity to query node(s) ***Q***
- **Importance pooling**
  - Choose nodes with top **K** visit counts
  - Pool over the chosen nodes
  - The chosen nodes are not necessarily neighbors

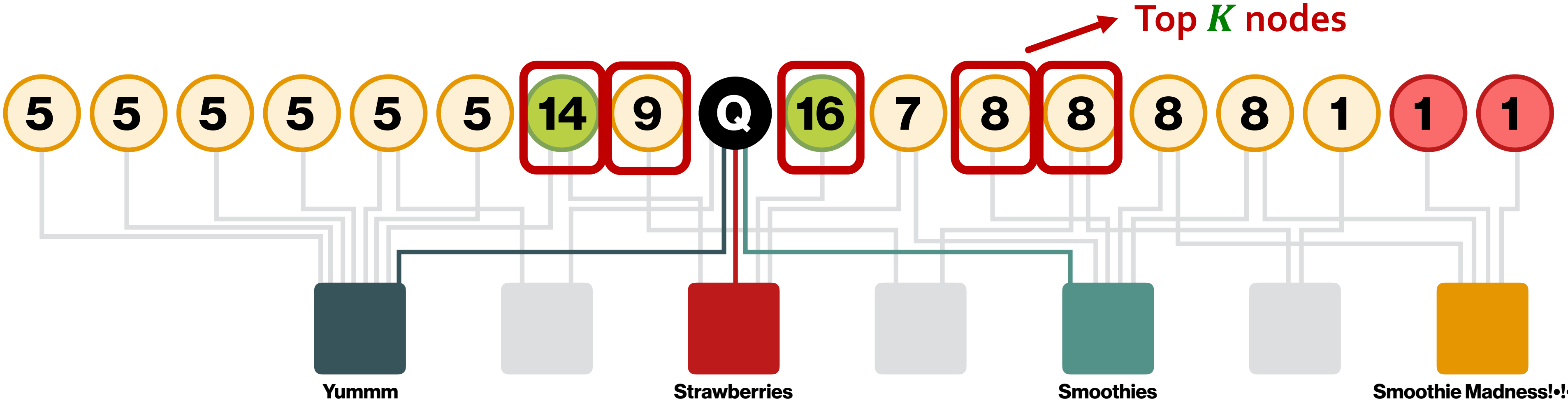


# Key Innovation (2): Importance Pooling

- **Example:** suppose $K$=5
- Rank nodes based on Random Walk visit counts
- Pick **top $K$ nodes** and normalize counts

$$\frac{16}{55}, \frac{14}{55}, \frac{9}{55}, \frac{8}{55}, \frac{8}{55}$$

- Aggregate messages from the top $K$ nodes



# Key Innovation (2): Importance Pooling

- **Pick top K nodes and normalize counts**

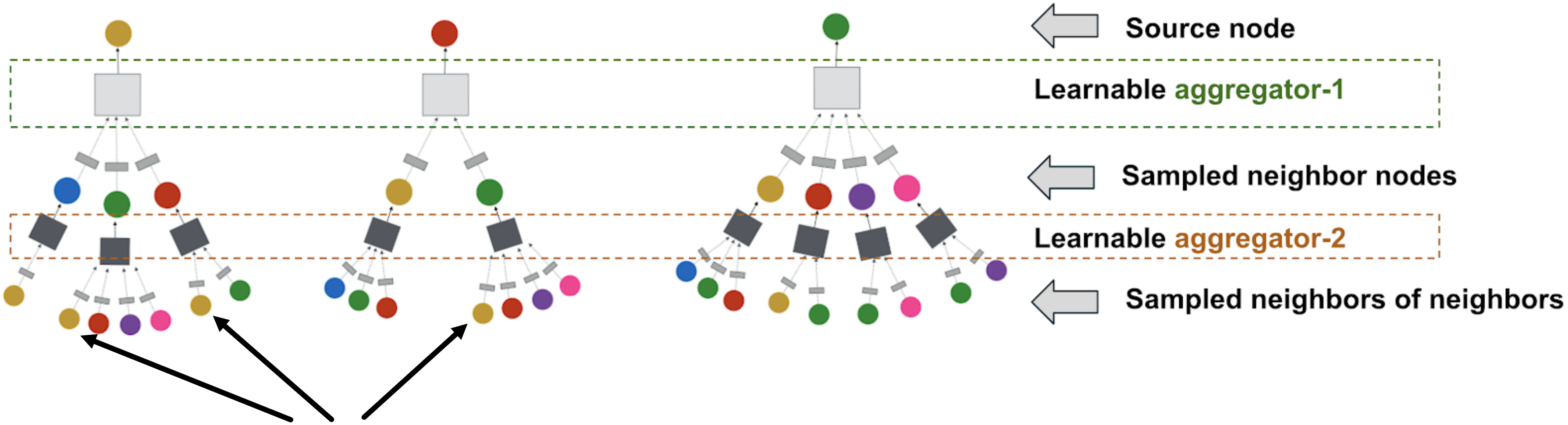
$$\frac{16}{55}, \frac{14}{55}, \frac{9}{55}, \frac{8}{55}, \frac{8}{55}$$

- **GraphSAGE mean pooling**
  - Average the messages from direct neighbors
- **PinSAGE Importance pooling**
  - Use the normalized counts as weights for weighted mean of messages from the top K nodes
- PinSAGE uses $K = 50$
  - Negligible performance gain for $K > 50$

# Key Innovation (3)

**Four key innovations:**

3. **Efficient MapReduce inference**
   - **Problem:** Many repeated computation if using **localized graph convolution** at inference step
   - Need to avoid repeated computation



Repeated computation

# Key Innovation (4)

- **Recall how we obtain negative examples**

$$\mathcal{L} = \sum_{(u,v)\in \boxed{\mathcal{D}}} \max(0, -\mathbf{z}_u^\top \boxed{\mathbf{z}_v} + \mathbf{z}_u^\top \boxed{\mathbf{z}_n} + \boxed{\Delta})$$

set of training pairs from logs

"positive"/true example

"negative" example

"margin" (i.e., how much larger positive pair similarity should be compared to negative)

**Positive Example**

**Random Negative**

# Key Innovation (4)

**Goal: Identify target pin among 3B pins**

- **Issue: Need to learn with resolution of 100 vs. 3B**
- **Massive size: 3** billion nodes, 20 billion edges
- **Idea:** Use harder and harder negative samples

$$\mathcal{L} = \sum_{(u,v)\in \boxed{\mathcal{D}}} \max(0, -\mathbf{z}_u^\top \boxed{\mathbf{z}_v} + \mathbf{z}_u^\top \boxed{\mathbf{z}_n} + \boxed{\Delta})$$

set of training pairs from logs

“positive”/true example

*negative examples*

“margin” (i.e., how much larger positive pair similarity should be compared to negative)

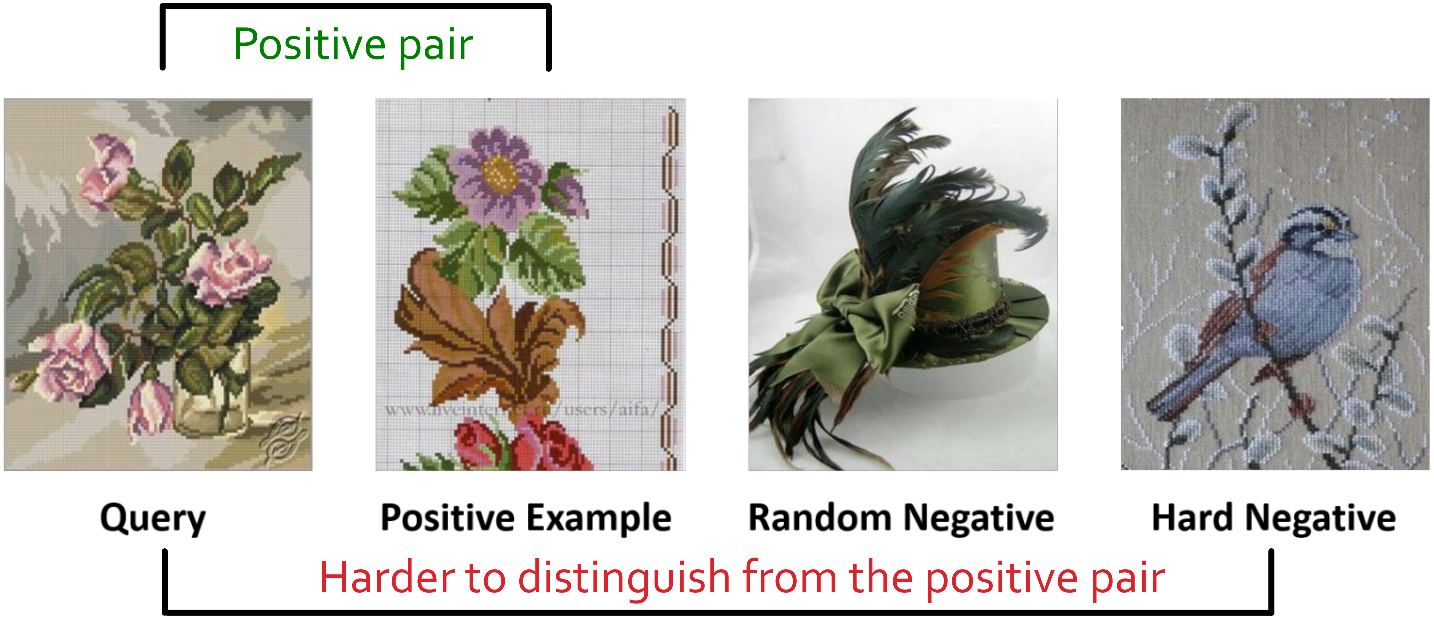
**Positive Example**

**Hard Negative**

**Force model to learn subtle distinctions between pins**

# Key Innovation (4)

- **Hard negative examples** improve performance



Positive pair

Query

Positive Example

Random Negative

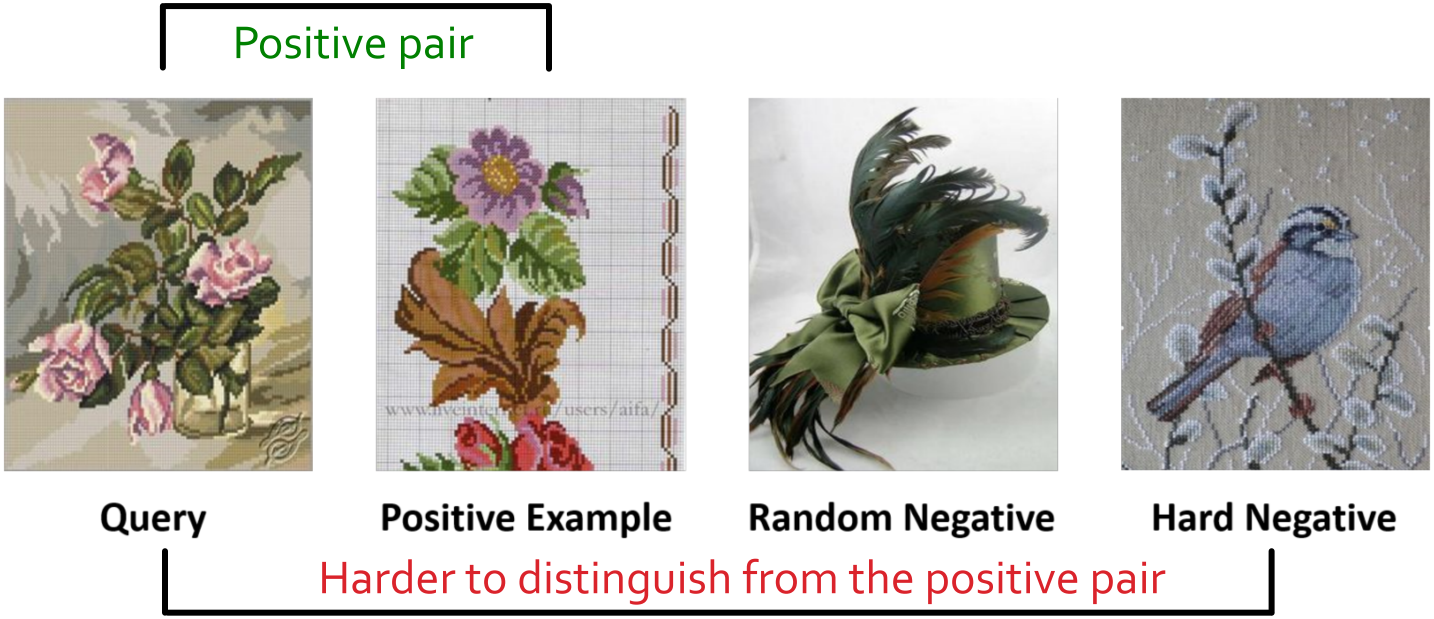
Hard Negative

Harder to distinguish from the positive pair

- **How to obtain hard negatives:** Use **random walks**:
  - Use nodes with visit counts ranked at 1000-5000 as hard negatives
  - Have something in common, but are not too similar

# Key Innovation (4)

- **Hard negative examples improve performance**



Positive pair

Query

Positive Example

Random Negative

Hard Negative

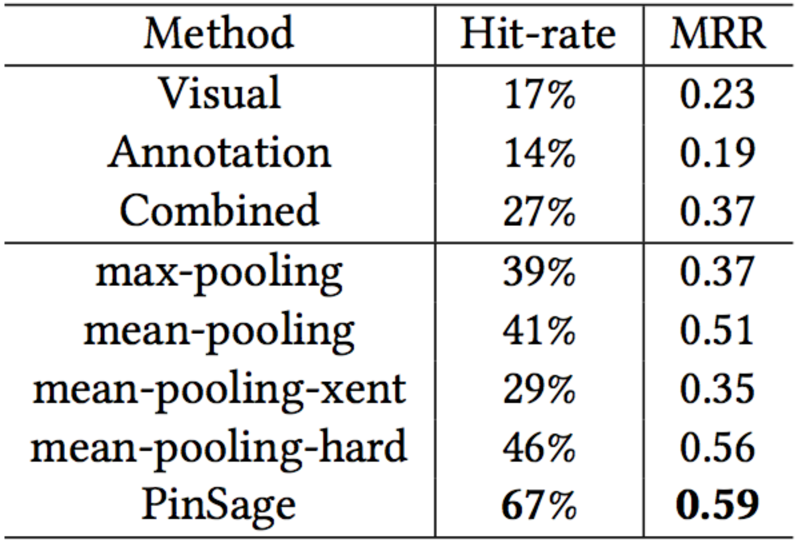
Harder to distinguish from the positive pair

- **Curriculum training** on hard negatives
  - Start with random negative examples
  - Provide **harder** negative examples over time

# PinSage: Experiments

## Related Pin recommendations

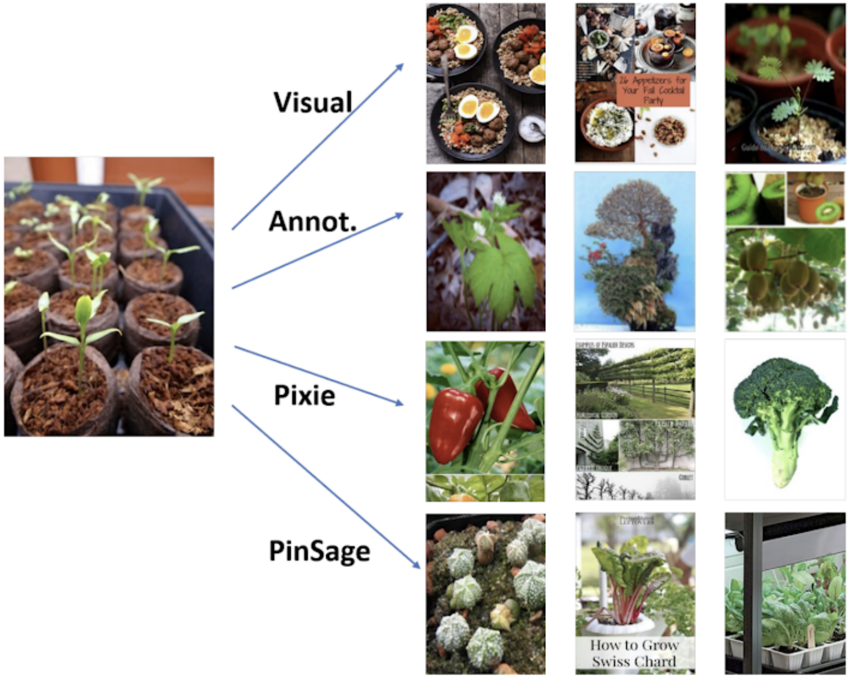
- Given a user just saved pin **Q**, predict what pin **X** are they going to save next
- **Setup:** Embed 3B pins, find nearest neighbors of **Q**
- **Baseline embeddings:**
  - **Visual**: VGG visual embeddings
    **Annotation**: Word2vec embeddings
  - **Combined**: Concatenate embeddings

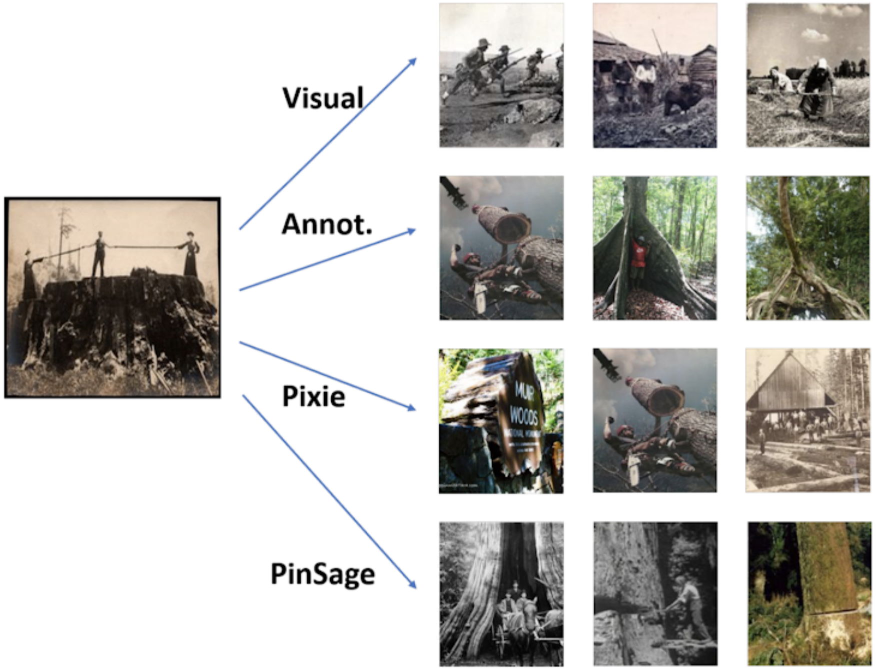


| Method | Hit-rate | MRR |
|---|---|---|
| Visual | 17% | 0.23 |
| Annotation | 14% | 0.19 |
| Combined | 27% | 0.37 |
| max-pooling | 39% | 0.37 |
| mean-pooling | 41% | 0.51 |
| mean-pooling-xent | 29% | 0.35 |
| mean-pooling-hard | 46% | 0.56 |
| PinSage | **67%** | **0.59** |

**MRR:** Mean reciprocal rank of the positive example X w.r.t Q
**Hit rate:** Fraction of times the positive example X is among top K closest to Q

# Example Pin Recommendations

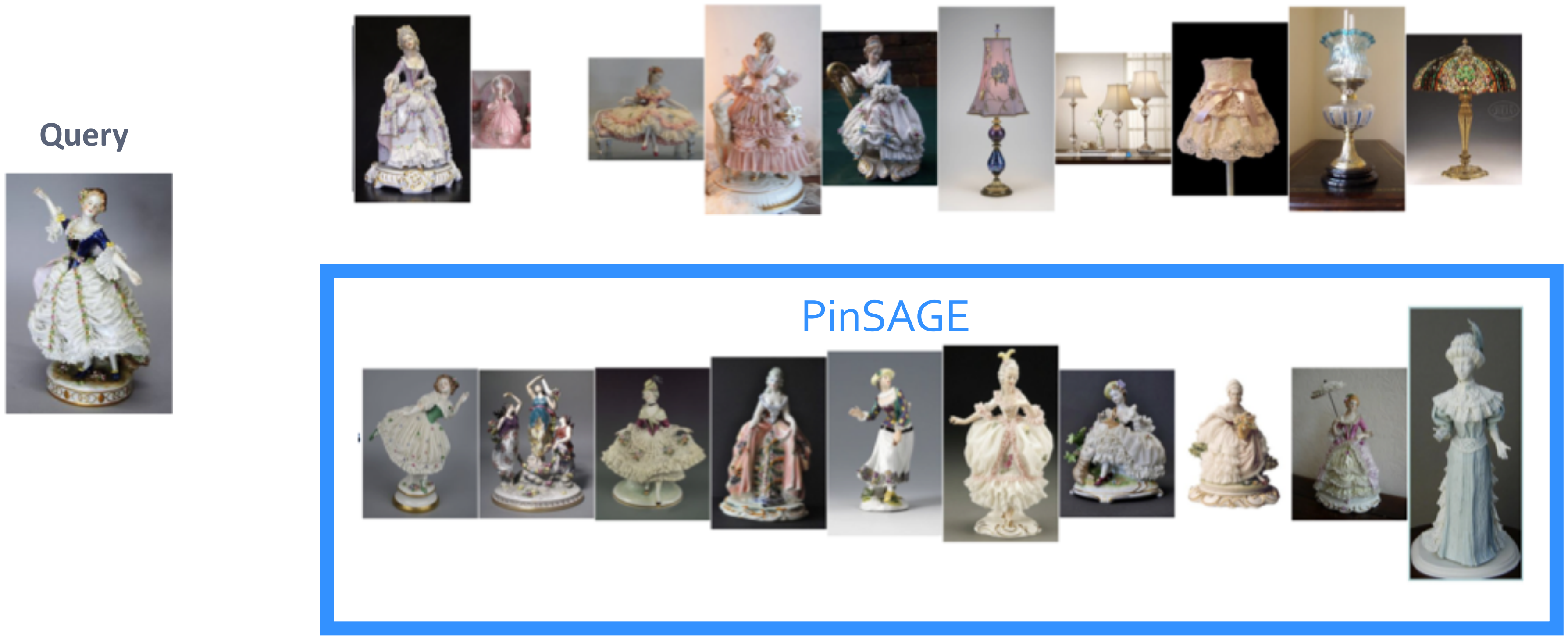




**Pixie (graph-based)**: the method of simulating random walks starting at query Pin using the Pixie algorithm in class. Items with top scores are retrieved as recommendations

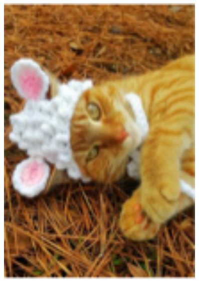
**Visual, Annot. (feature-based)**: nearest neighbor recommendation using visual (CNN) and annotation features of pins

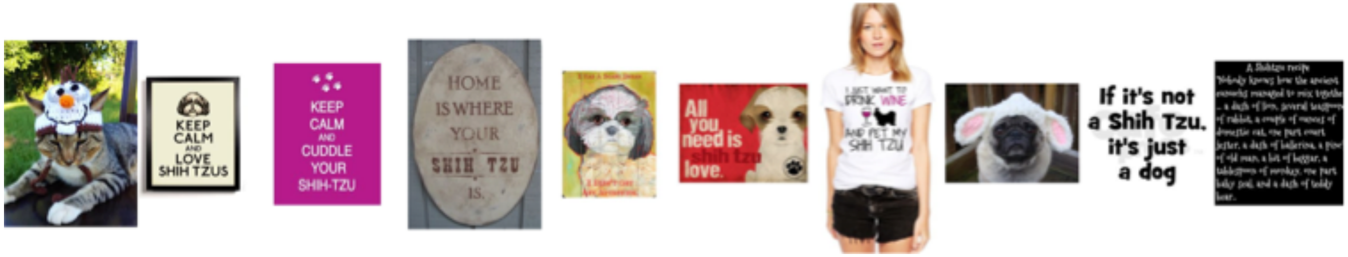
# Comparing against Prod (1)



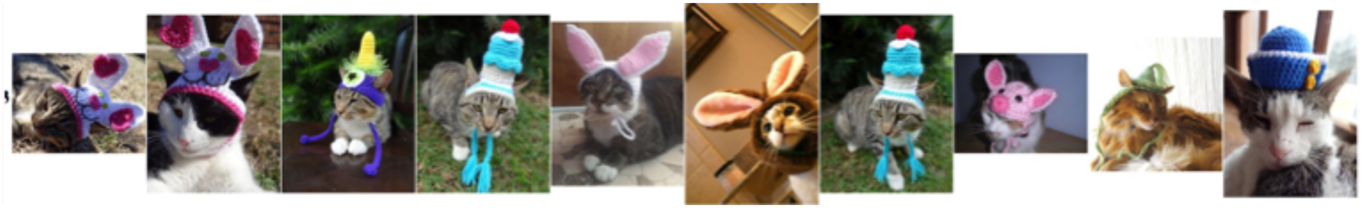
# Comparing against Prod (2)

Query




KEEP CALM AND LOVE SHIH TZUS
KEEP CALM AND CUDDLE YOUR SHIH-TZU
HOME IS WHERE YOUR SHIH TZU IS.
All you need is shih tzu love.
I JUST WANT TO DRINK WINE AND PET MY SHIH TZU
If it's not a Shih Tzu, it's just a dog

PinSAGE



# Outline of Today's Lecture

1. **GNN recommendation (PinSage)** ✓

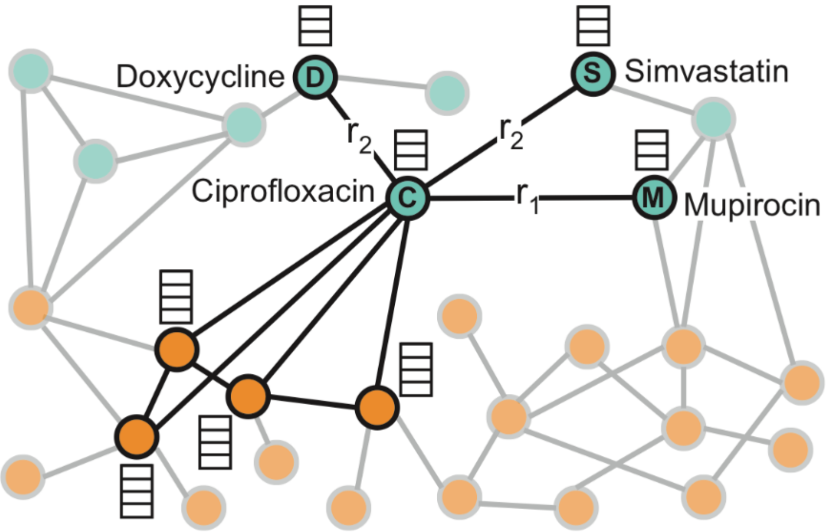
2. **Heterogeneous GNN (Decagon)** ☜

3. **Goal-directed generation (GCPN)**

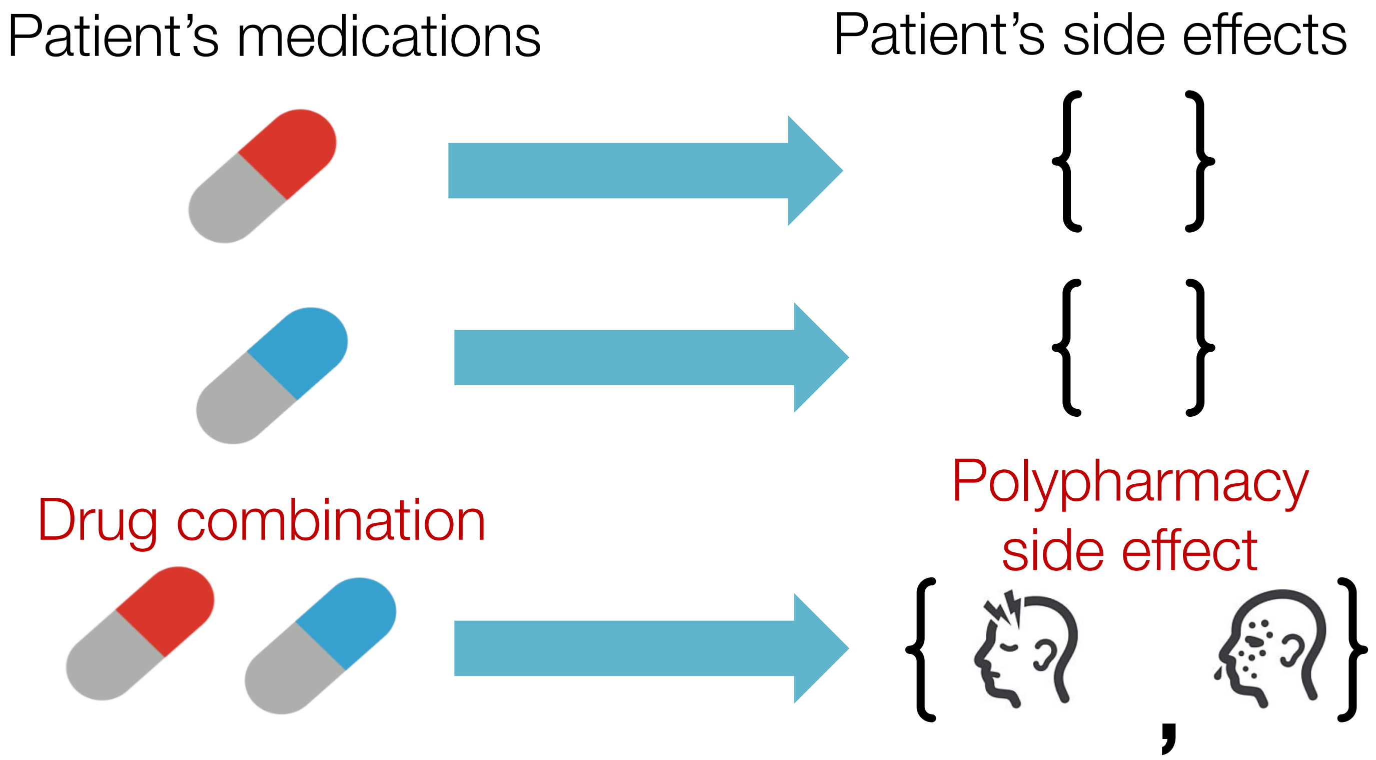
# DECAGON: Heterogeneous GNN

# Challenge

- So far we only applied GNNs to simple graphs
  - GNNs do not explicitly use node and edge type information
- Real networks are often **heterogeneous**
- How to use GNN for heterogeneous graphs?



# Polypharmacy Side Effects



Patient’s medications

Patient’s side effects

Drug combination

Polypharmacy side effect

Polypharmacy: use multiple drugs for a disease

# 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

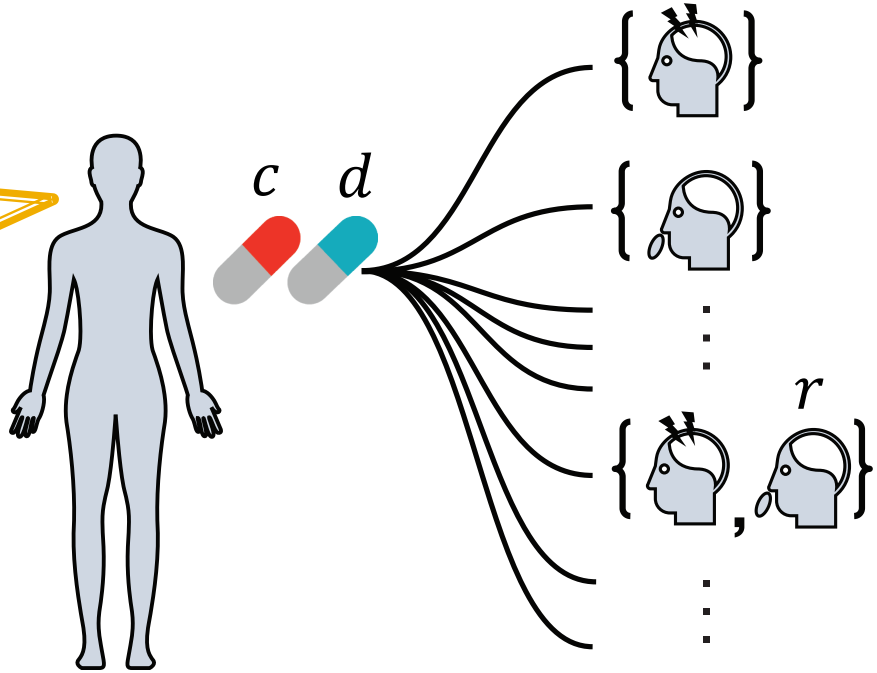
# Modeling Polypharmacy

- Systematic experimental screening of drug interactions is challenging

- **Idea:** Computationally screen/predict polypharmacy side effects
  - Use molecular, pharmacological and patient population data
  - Guide translational strategies for combination treatments in patients

# This Work

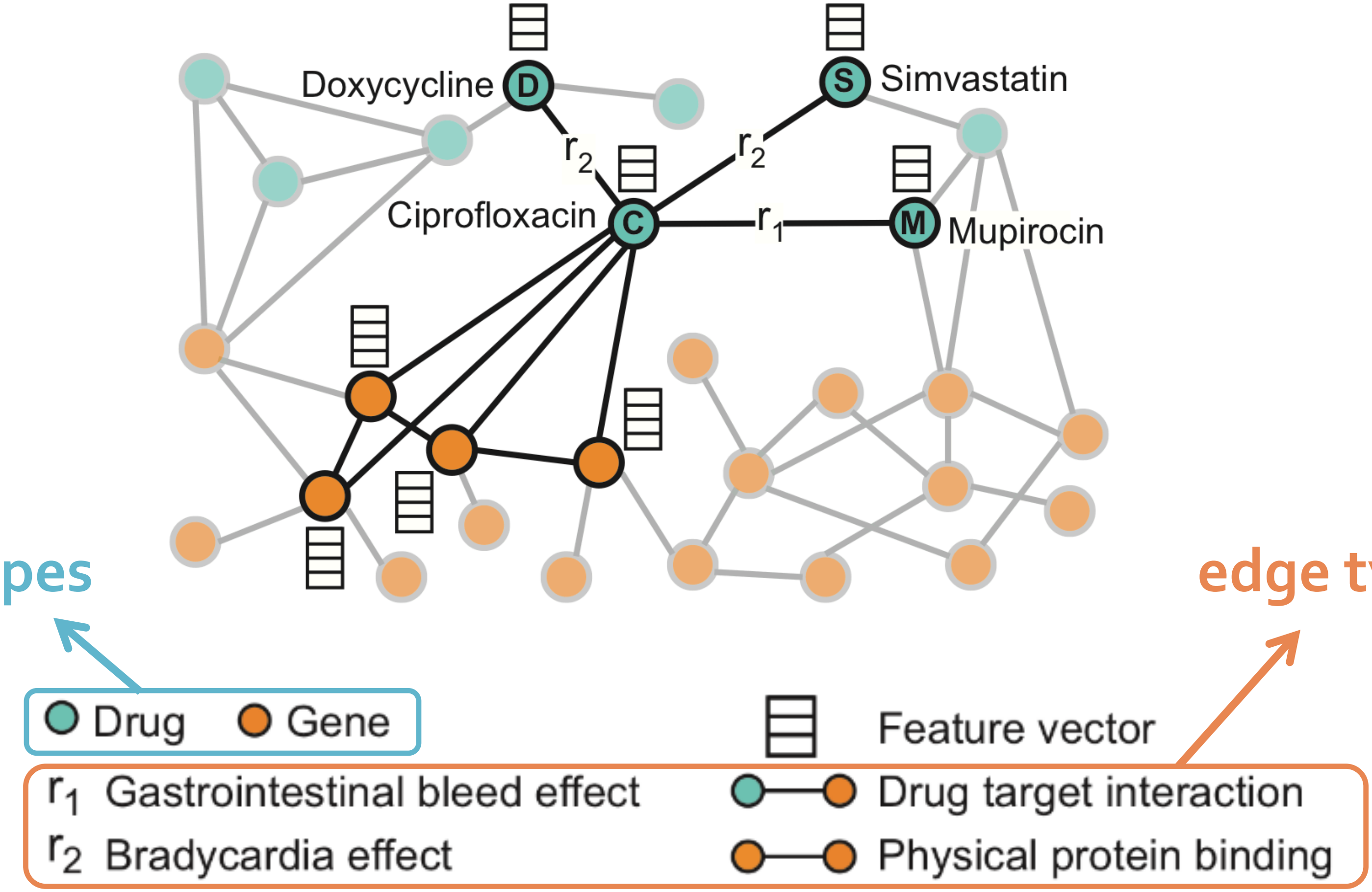
How likely with a pair of drugs $c, d$ lead to side effect $r$?



Model and predict side effects of drug pairs

# Problem Formulation: Graphs

- **Heterogeneous (multimodal) graphs:** graphs with different node types and/or edge types



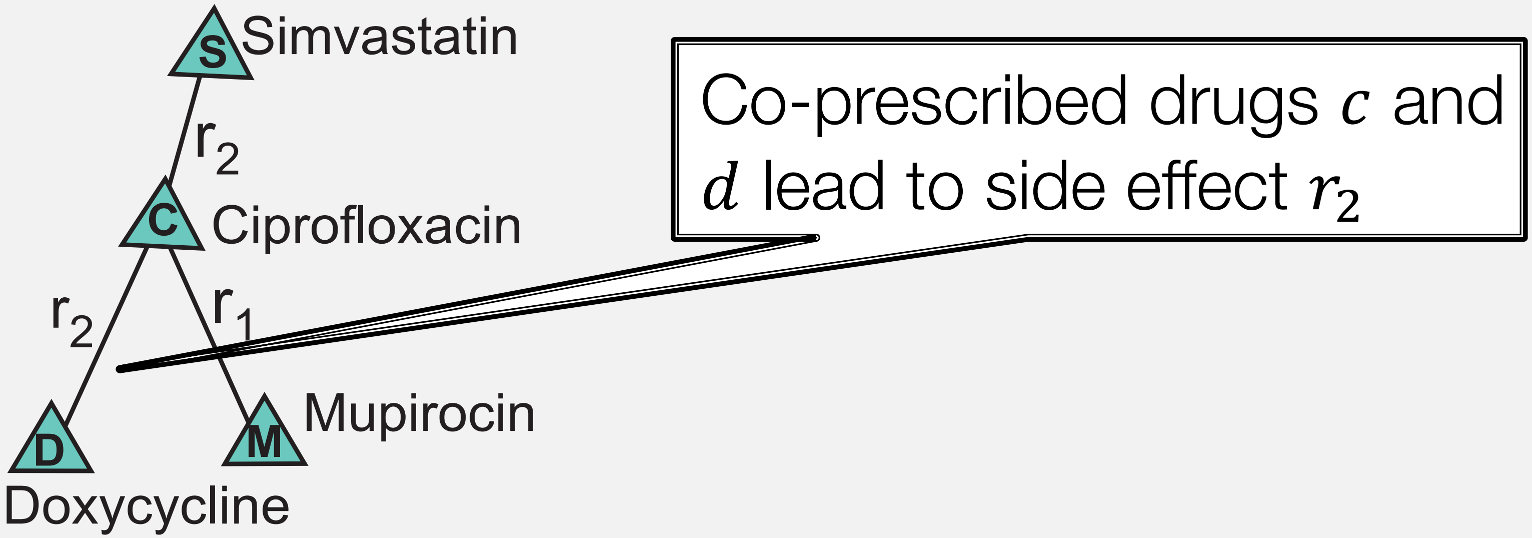
2 node types

edge types

# Problem Formulation: Predict

**Goal:** Given a partially observed graph, predict labeled edges between drug nodes

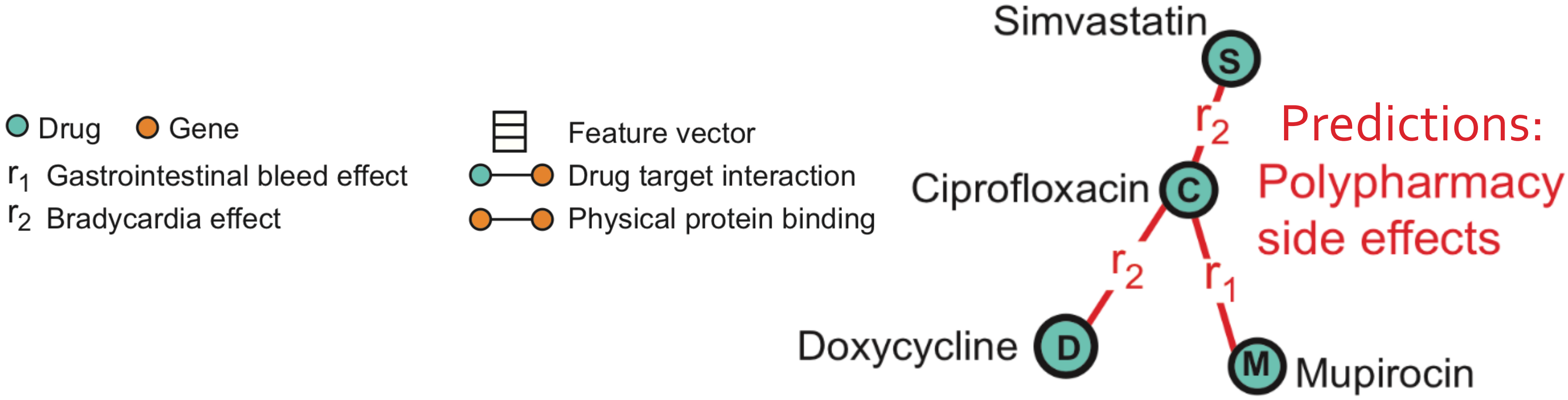
**Query:** Given a drug pair $c, d$, how likely does an edge $(c, r_2, d)$ exist?



Co-prescribed drugs $c$ and $d$ lead to side effect $r_2$

# Task Description

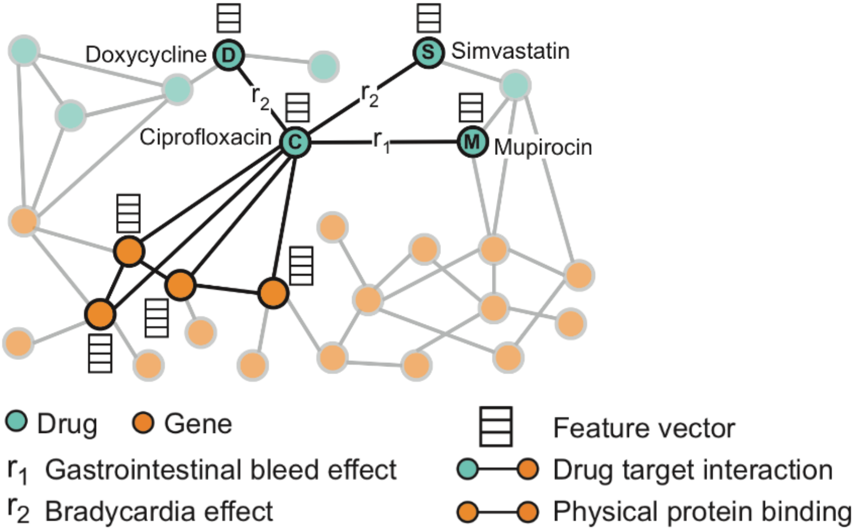
- Predict labeled edges between drugs nodes
  - i.e., predict the likelihood that an edge $(c, r_2, s)$ exists between drug nodes $c$ and $s$
  - Meaning: Drug combination $(c, s)$ leads to polypharmacy side effect $r_2$



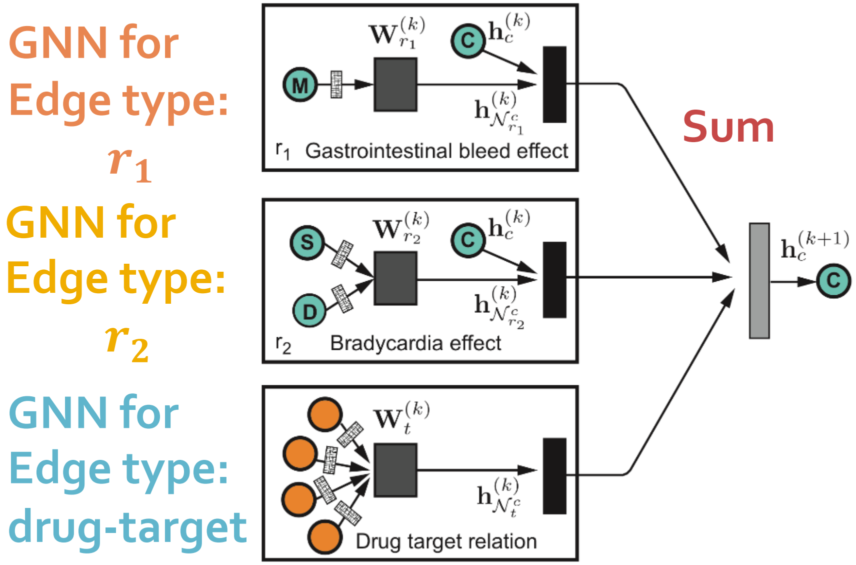
# Model: Heterogenous GNN

- **Key Insight:** Compute GNN messages from each edge type, then aggregate across different edge types

- **Input:** heterogenous graph
- **Output:** node embeddings



## One layer of Heterogeneous GNN



GNN for Edge type: $r_1$

GNN for Edge type: $r_2$

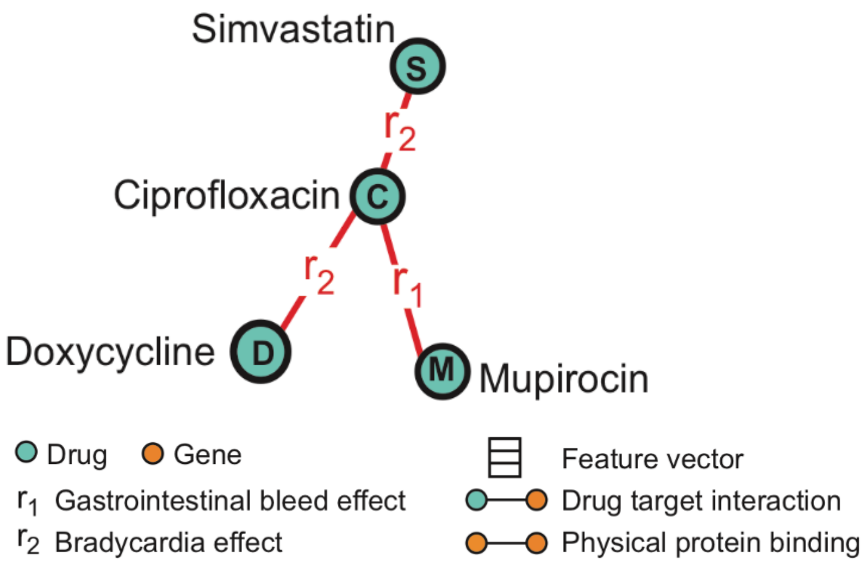
GNN for Edge type: drug-target

Sum

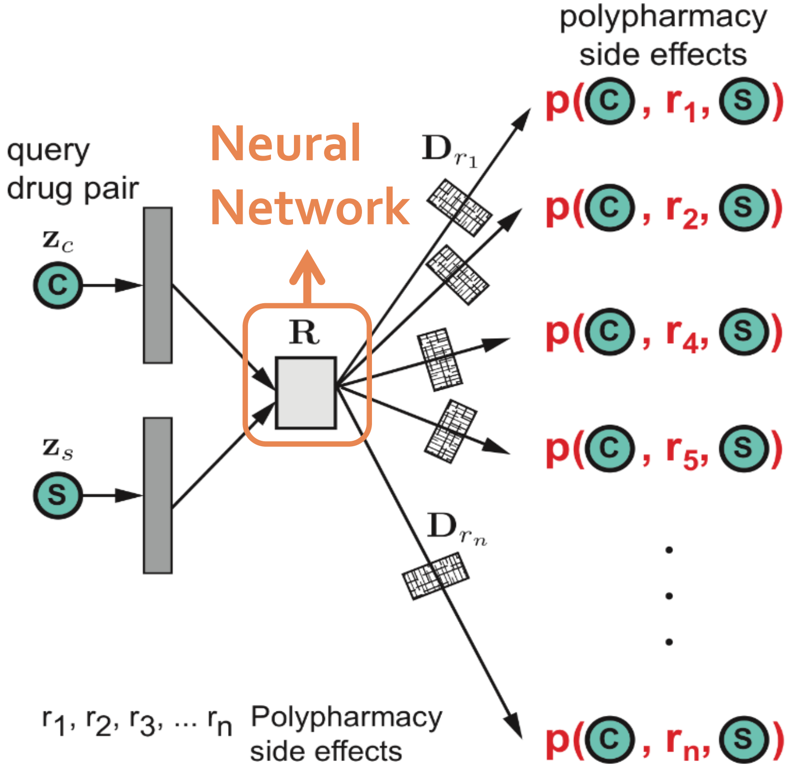
# Making Edge Predictions

- **Key Insight:** Use pair of computed node embeddings to make edge predictions

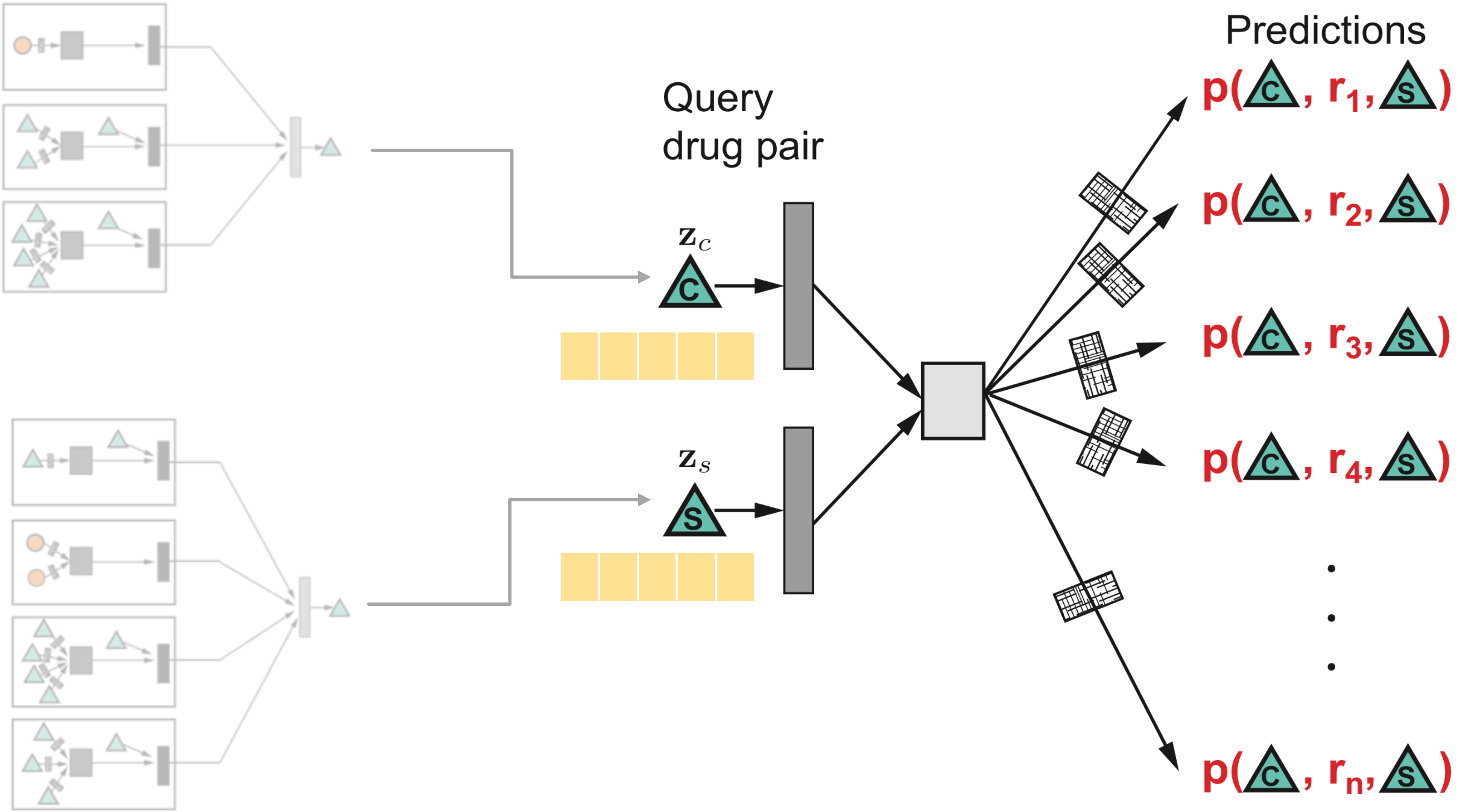
- **Input:** Node embeddings of query drug pairs
- **Output:** predicted edges



**Predict possible edges with NN**



# Decoder: Link Prediction



Query drug pair

$\mathbf{z}_C$

$\mathbf{z}_S$

Predictions

$p(C, r_1, S)$

$p(C, r_2, S)$

$p(C, r_3, S)$

$p(C, r_4, S)$

.
.
.

$p(C, r_n, S)$

p – probability

# Experiment Setup

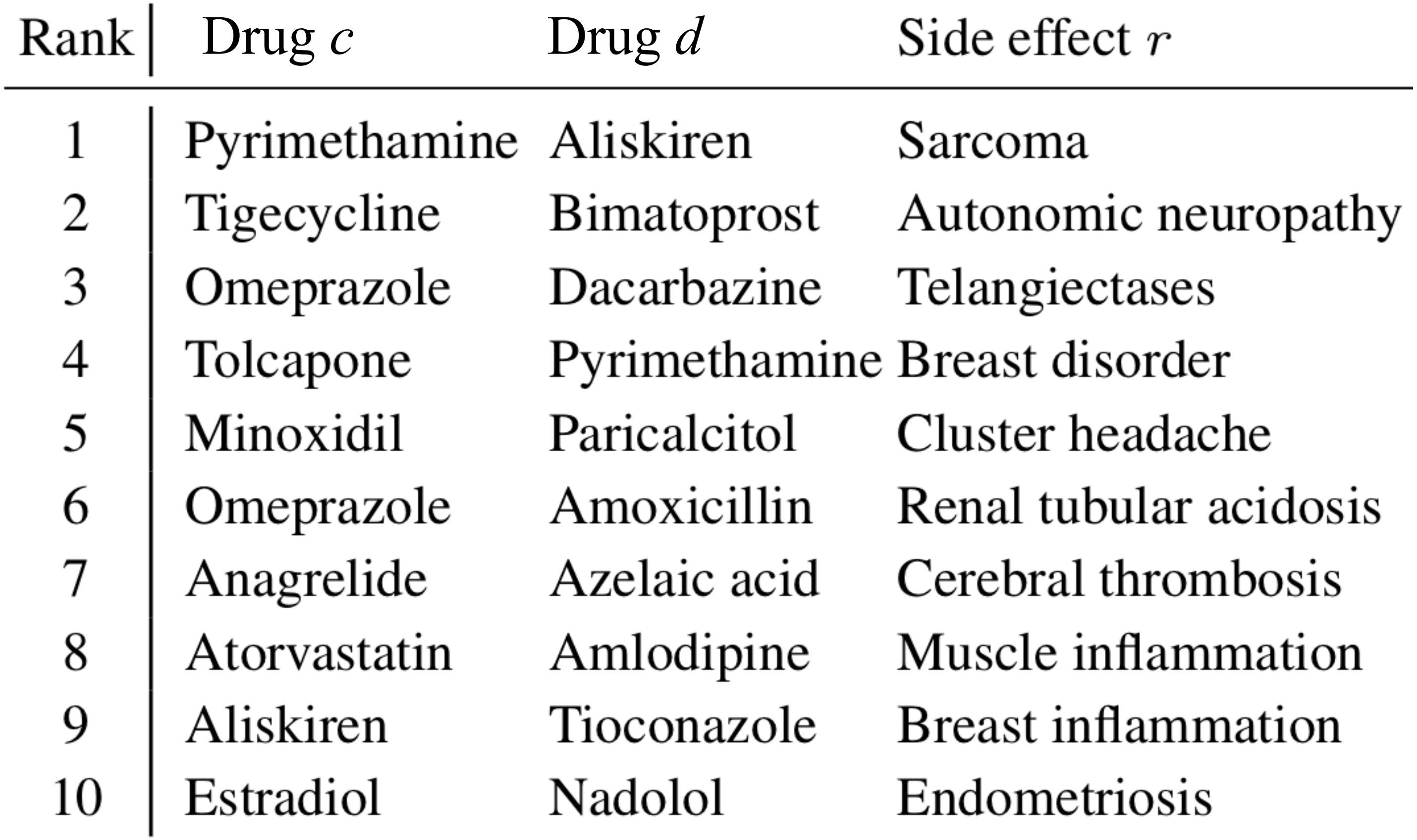
- **Data:**
  - Graph over Molecules: protein-protein interaction and drug target relationships
  - Graph over Population: Side effects of individual drugs, polypharmacy side effects of drug combinations
- **Setup:**
  - Construct a heterogeneous graph of all the data
  - Train: Fit a model to predict known associations of drug pairs and polypharmacy side effects
  - Test: Given a query drug pair, predict candidate polypharmacy side effects

# 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 | 0.693 | 0.613 | 0.476 |
| Node2vec | 0.725 | 0.708 | 0.643 |
| Drug features | 0.736 | 0.722 | 0.679 |

- Up to 54% improvement over baselines
- First opportunity to computationally flag polypharmacy side effects for follow-up analyses

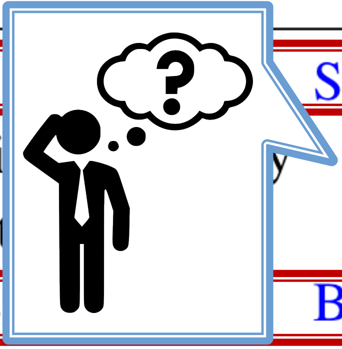
# *De novo* Predictions



| Rank | Drug $c$ | Drug $d$ | Side effect $r$ |
|---|---|---|---|
| 1 | Pyrimethamine | Aliskiren | Sarcoma |
| 2 | Tigecycline | Bimatoprost | Autonomic neuropathy |
| 3 | Omeprazole | Dacarbazine | Telangiectases |
| 4 | Tolcapone | Pyrimethamine | Breast disorder |
| 5 | Minoxidil | Paricalcitol | Cluster headache |
| 6 | Omeprazole | Amoxicillin | Renal tubular acidosis |
| 7 | Anagrelide | Azelaic acid | Cerebral thrombosis |
| 8 | Atorvastatin | Amlodipine | Muscle inflammation |
| 9 | Aliskiren | Tioconazole | Breast inflammation |
| 10 | Estradiol | Nadolol | Endometriosis |

# *De novo* Predictions

| Rank | Drug *c* | Drug *d* | Side effect *r* | Evidence found |
|---|---|---|---|---|
| 1 | Pyrimethamine | Aliskiren | Sarcoma | Stage *et al.* 2015 |
| 2 | Tigecycline | Bimatoprost | Autonomi y | |
| 3 | Omeprazole | Dacarbazine | Telangiect | |
| 4 | Tolcapone | Pyrimethamine | Breast dis | Bicker *et al.* 2017 |
| 5 | Minoxidil | Paricalcitol | Cluster headache | |
| 6 | Omeprazole | Amoxicillin | Renal tubular acidosis | Russo *et al.* 2016 |
| 7 | Anagrelide | Azelaic acid | Cerebral thrombosis | |
| 8 | Atorvastatin | Amlodipine | Muscle inflammation | Banakh *et al.* 2017 |
| 9 | Aliskiren | Tioconazole | Breast inflammation | Parving *et al.* 2012 |
| 10 | Estradiol | Nadolol | Endometriosis | |



*Case Report*

**Severe Rhabdomyolysis due to Presumed Drug Interactions between Atorvastatin with Amlodipine and Ticagrelor**

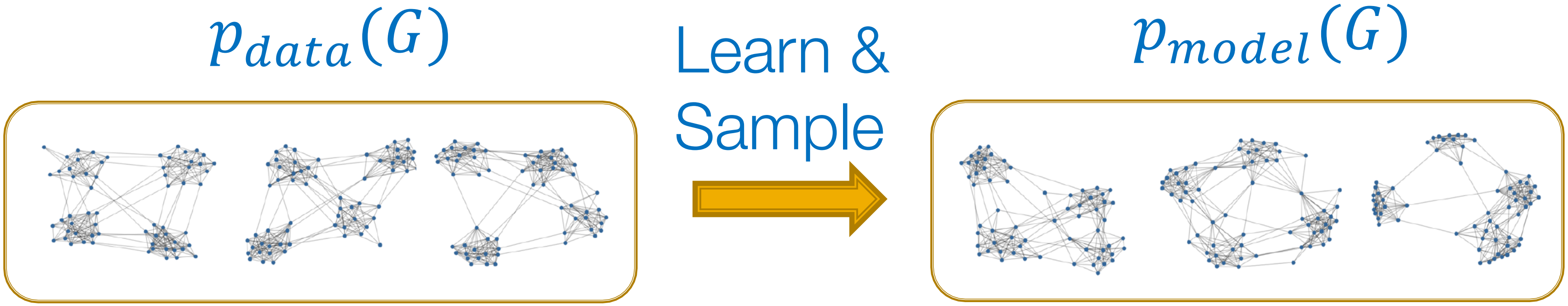
# Outline of Today's Lecture

1. **GNN recommendation (PinSage)** ✓
2. **Heterogeneous GNN (Decagon)** ✓
3. **Goal-directed generation (GCPN)** ☜

# GCPN:
# Goal-Directed Graph Generation (an extension of GraphRNN)

# Recap: Graph Generative Models

- **Given**: Graphs sampled from $p_{data}(G)$
- **Goal**:
  - Learn the distribution $p_{model}(G)$
  - Sample from $p_{model}(G)$



$p_{data}(G)$

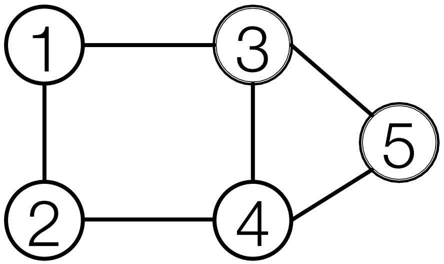
Learn & Sample

$p_{model}(G)$

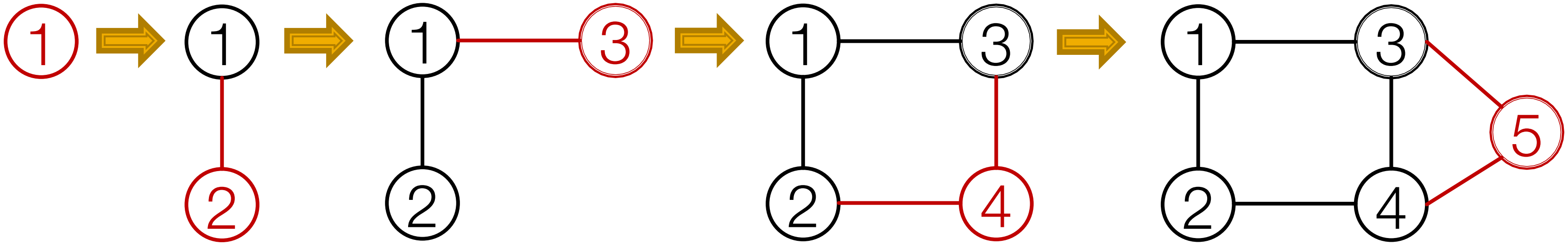
# Recap: GraphRNN Idea [You et al., ICML 2018]

## Generating graphs via sequentially adding nodes and edges

Graph $G$



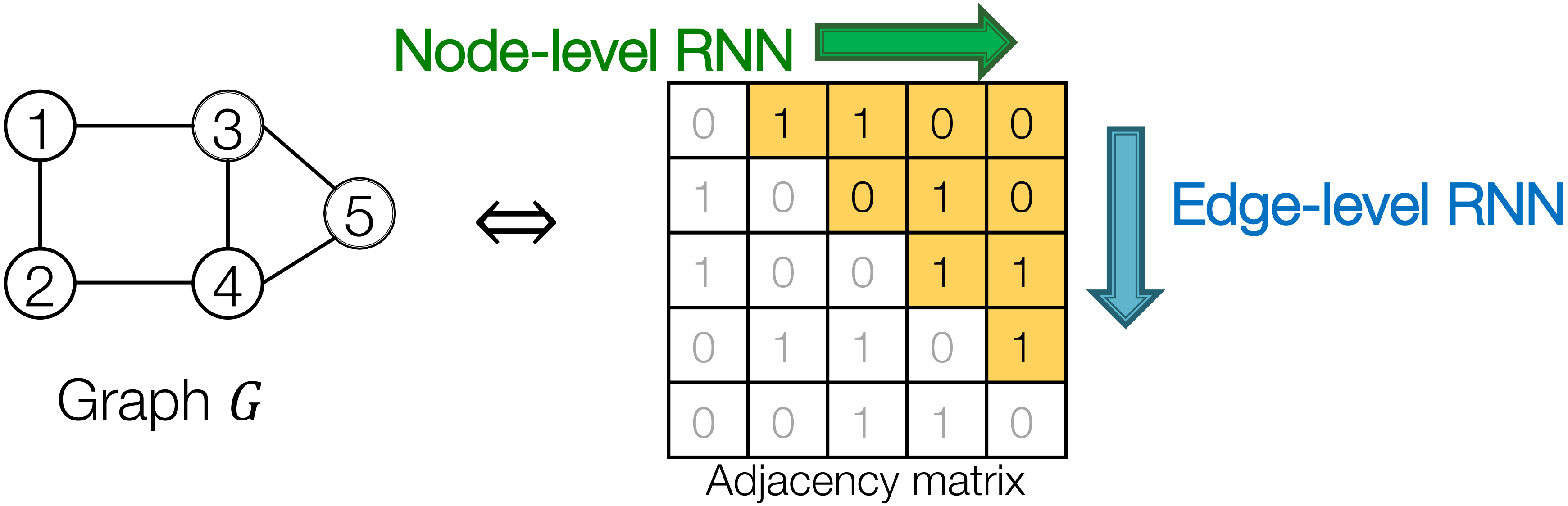
Generation process $S^{\pi}$



# GraphRNN: Two levels of RNN

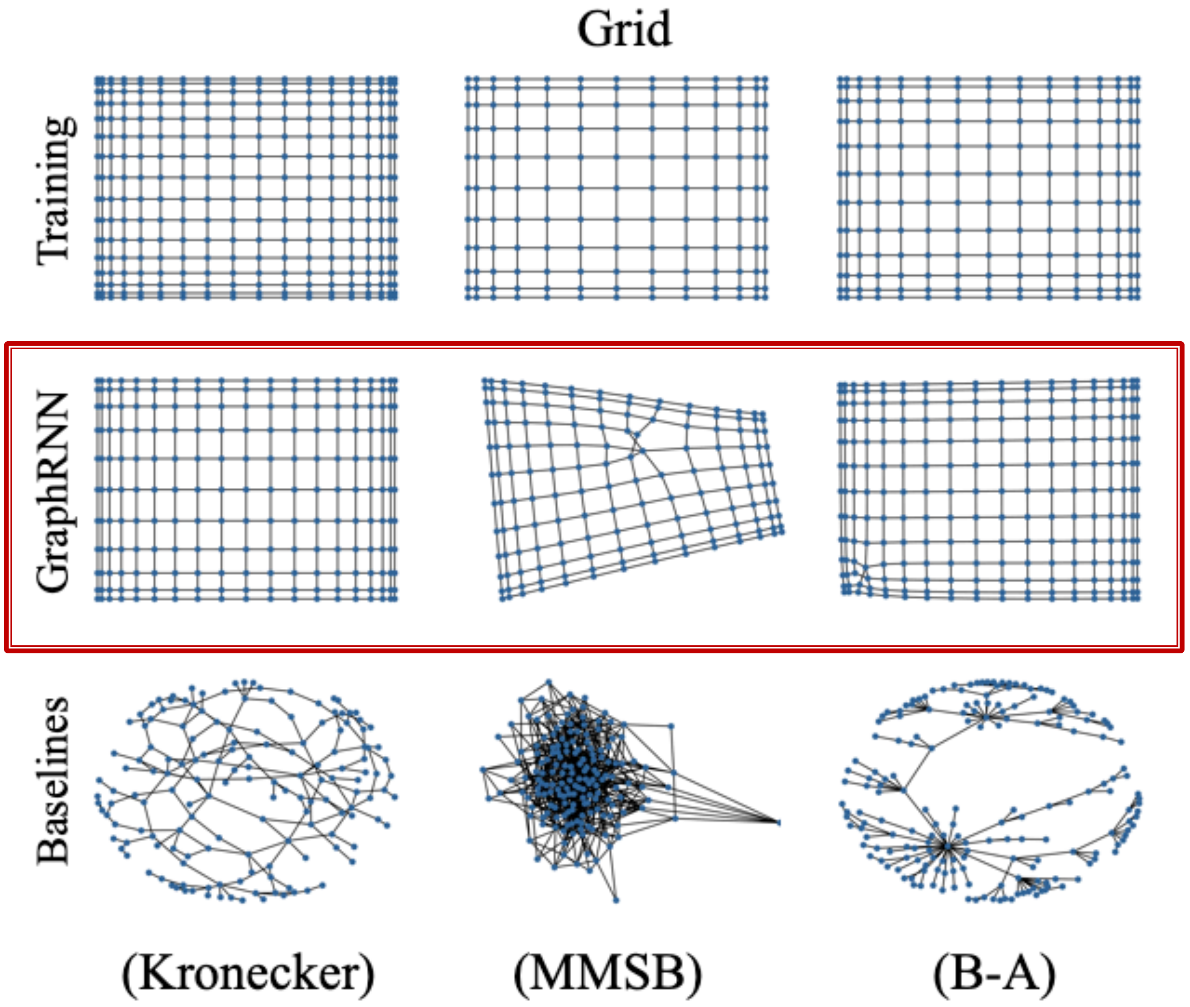
**Quick Summary of GraphRNN:**

- Generate a graph by generating a two level sequence
- Use RNN to generate the sequences

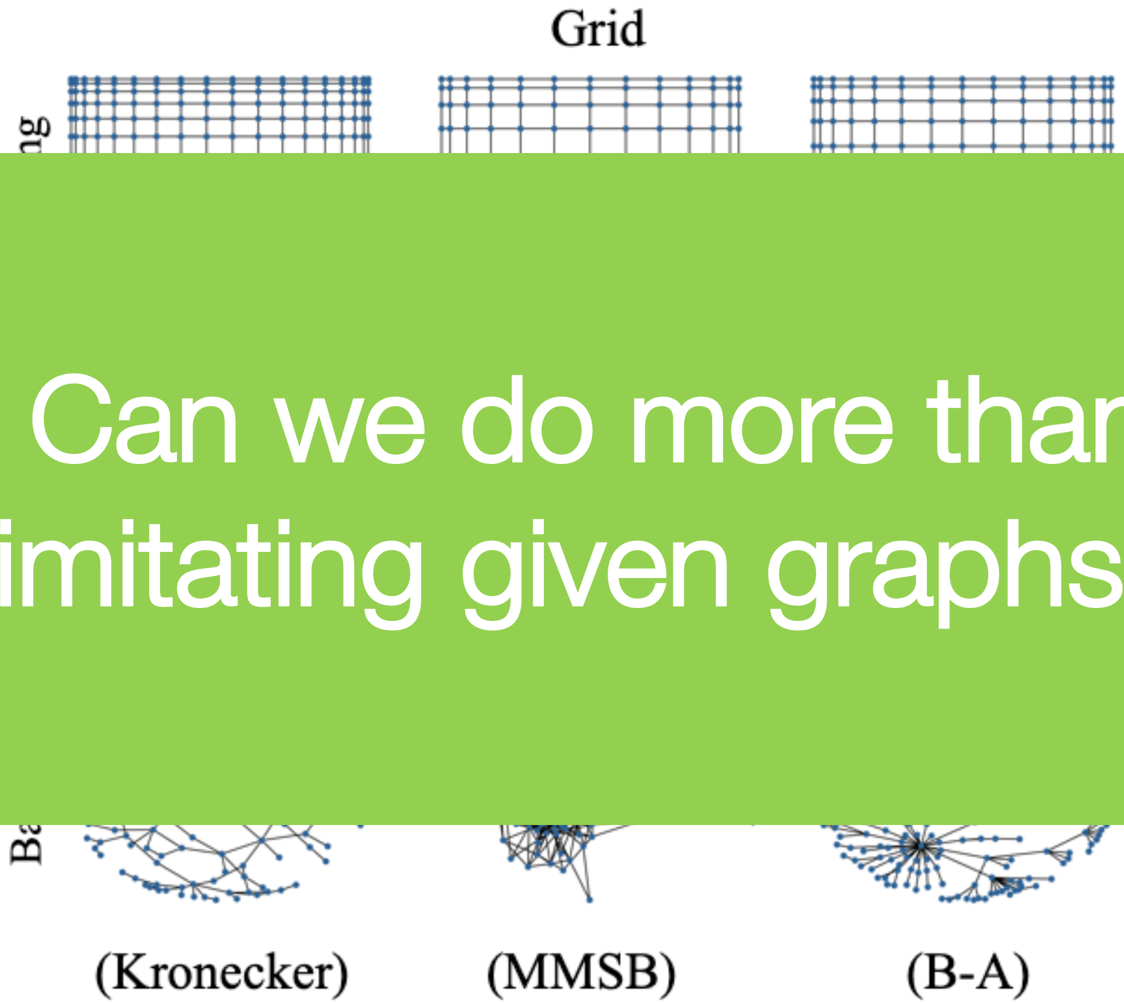


| | | | | |
|---|---|---|---|---|
| 0 | 1 | 1 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 0 | 1 | 1 |
| 0 | 1 | 1 | 0 | 1 |
| 0 | 0 | 1 | 1 | 0 |

# Imitating Given Graphs



# Imitating Given Graphs

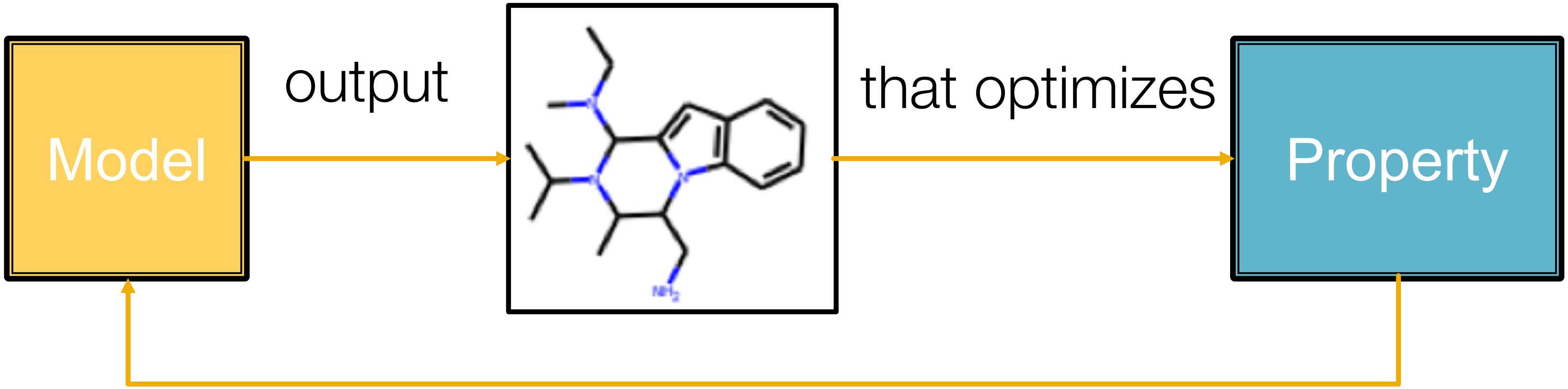


Can we do more than imitating given graphs?

# Drug Discovery

[You et al., NeurIPS 2018]

**Question:** Can we learn a model that can generate **valid** and **realistic** molecules with **high value of a given chemical property**?



Model

output

that optimizes

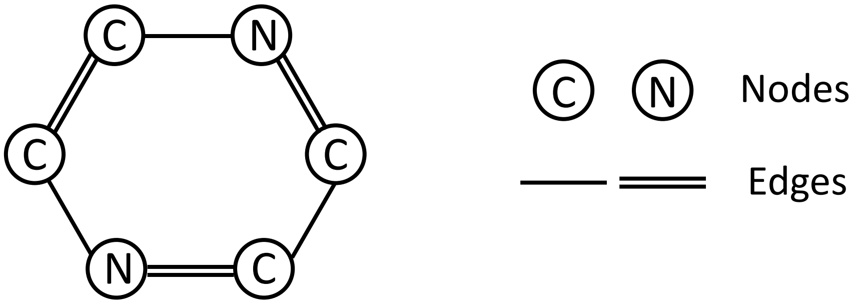
Property

e.g., `drug_likeness=0.95`

Graph Convolutional Policy Network for Goal-Directed Molecular Graph Generation. J. You, B. Liu, R. Ying, V. Pande, J. Leskovec. *Neural Information Processing Systems (NeurIPS)*, 2018.

# Molecules as Heterogenous Graphs

- **Node types:** C, N, O, …
- **Edge types:** single bond, double bond, …
- **Note:** “H”s can be automatically inferred via chemical validity rules, thus are ignored in molecular graphs



# Goal-Directed Graph Generation

**Generating graphs that:**

- **Optimize a given objective** (High scores)
  - e.g., drug-likeness
- **Obey underlying rules** (Valid)
  - e.g., chemical validity rules
- **Are learned from examples** (Realistic)
  - e.g., Imitating a molecule graph dataset

Graph Convolutional Policy Network for Goal-Directed Molecular Graph Generation. J. You, B. Liu, R. Ying, V. Pande, J. Leskovec. *Neural Information Processing Systems (NeurIPS)*, 2018.

# The Hard Part:

## Generating graphs that:

- **Optimize a given objective** (High scores)
  - e.g., drug-likeness
- **Obey underlying rules** (Valid)
  - e.g., chemical validity rules

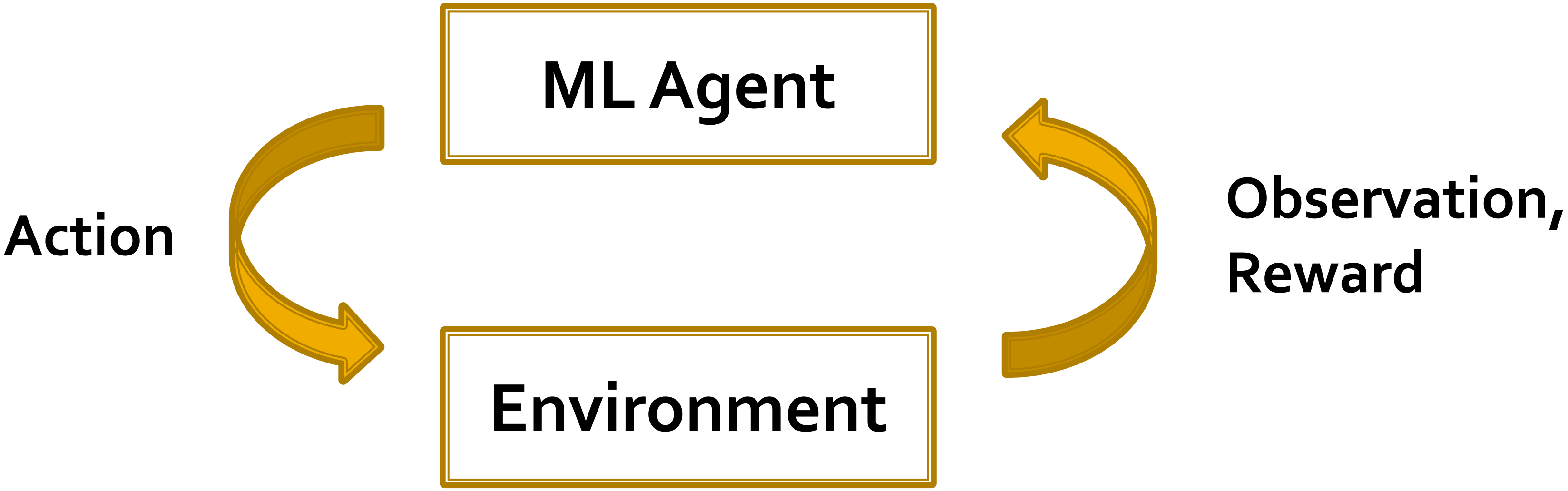
## Including “Black-box” in ML:

**Objectives like drug-likeness are governed by physical law, which are assumed to be unknown to us!**

Graph Convolutional Policy Network for Goal-Directed Molecular Graph Generation. J. You, B. Liu, R. Ying, V. Pande, J. Leskovec. *Neural Information Processing Systems (NeurIPS)*, 2018.

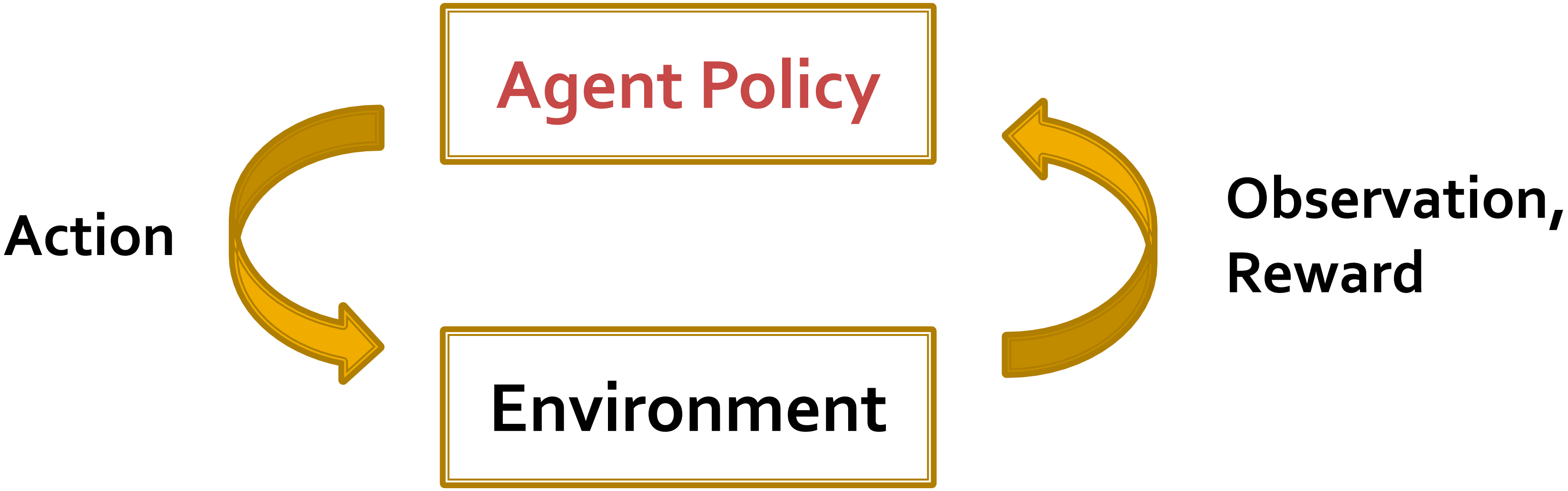
# Solution: Reinforcement Learning

- A ML agent **observes** the environment, takes an **action** to interact with the environment, and receives positive or negative **reward**
- The agent then **learns from this loop**
- **Key**: Environment is a **blackbox** to the agent



# Policy-based RL

- **Policy:** Agent behavior, which maps observation to action
- **Policy-based RL:** An agent directly learns an optimal policy from data

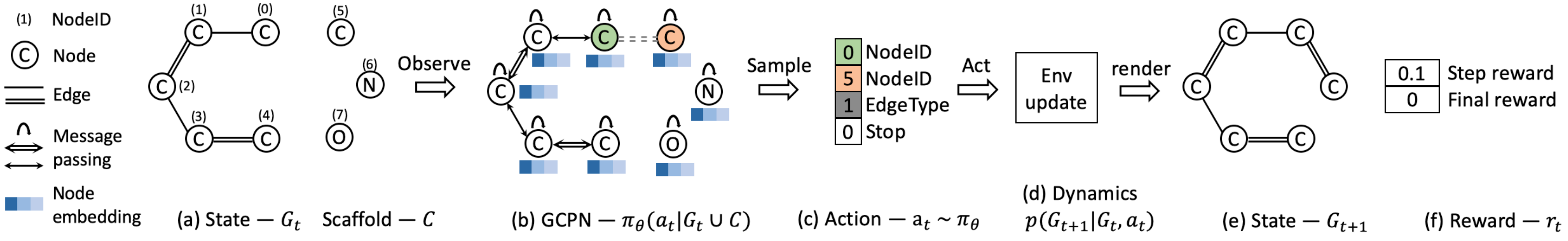


# Model: GCPN

**Graph Convolutional Policy Network** combines graph representation + RL:

- Graph Neural Network captures complex structural information, and enables validity check in each state transition (Valid)
- Reinforcement learning optimizes intermediate/final rewards (High scores)
- Adversarial training imitates examples in given datasets (Realistic)

# Overview of GCPN



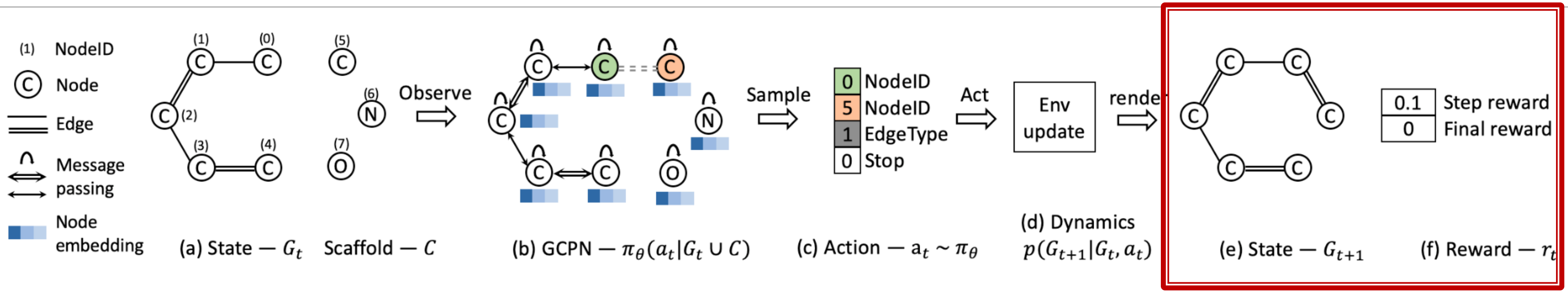
(a) State — $G_t$ Scaffold — $C$ (b) GCPN — $\pi_\theta(a_t|G_t \cup C)$ (c) Action — $a_t \sim \pi_\theta$ (d) Dynamics $p(G_{t+1}|G_t, a_t)$ (e) State — $G_{t+1}$ (f) Reward — $r_t$

- (a) Insert nodes/scaffolds
- (b) Compute state via GCN
- (c) Sample next action
- (d) Take action (check chemical validity)
- (e, f) Compute reward

# How Do We Set the Reward?

- Learn to take valid action
  - At each step, assign small positive reward for valid action
- Optimize desired properties
  - At the end, assign positive reward for high desired property
- Generate realistic graphs
  - At the end, adversarially train a GCN discriminator, compute adversarial rewards that encourage realistic molecule graphs

# How Do We Set the Reward?

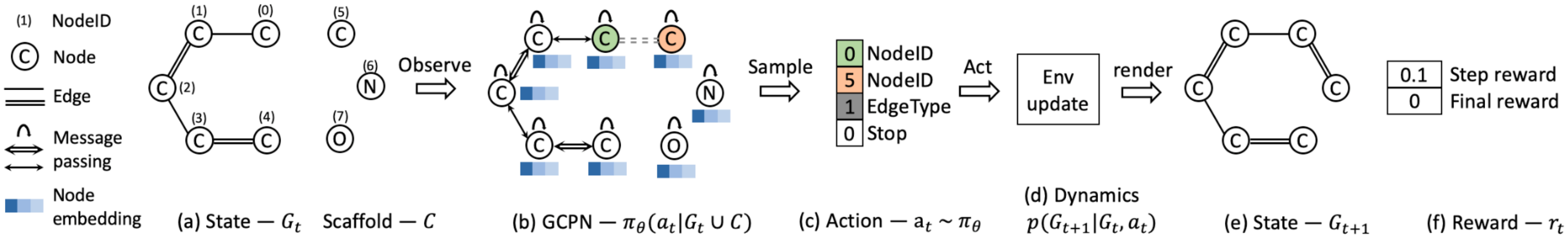


(a) State — $G_t$ Scaffold — $C$ (b) GCPN — $\pi_\theta(a_t|G_t \cup C)$ (c) Action — $a_t \sim \pi_\theta$ (d) Dynamics $p(G_{t+1}|G_t, a_t)$ (e) State — $G_{t+1}$ (f) Reward — $r_t$

Reward: $r_t$ = Final reward + Step reward

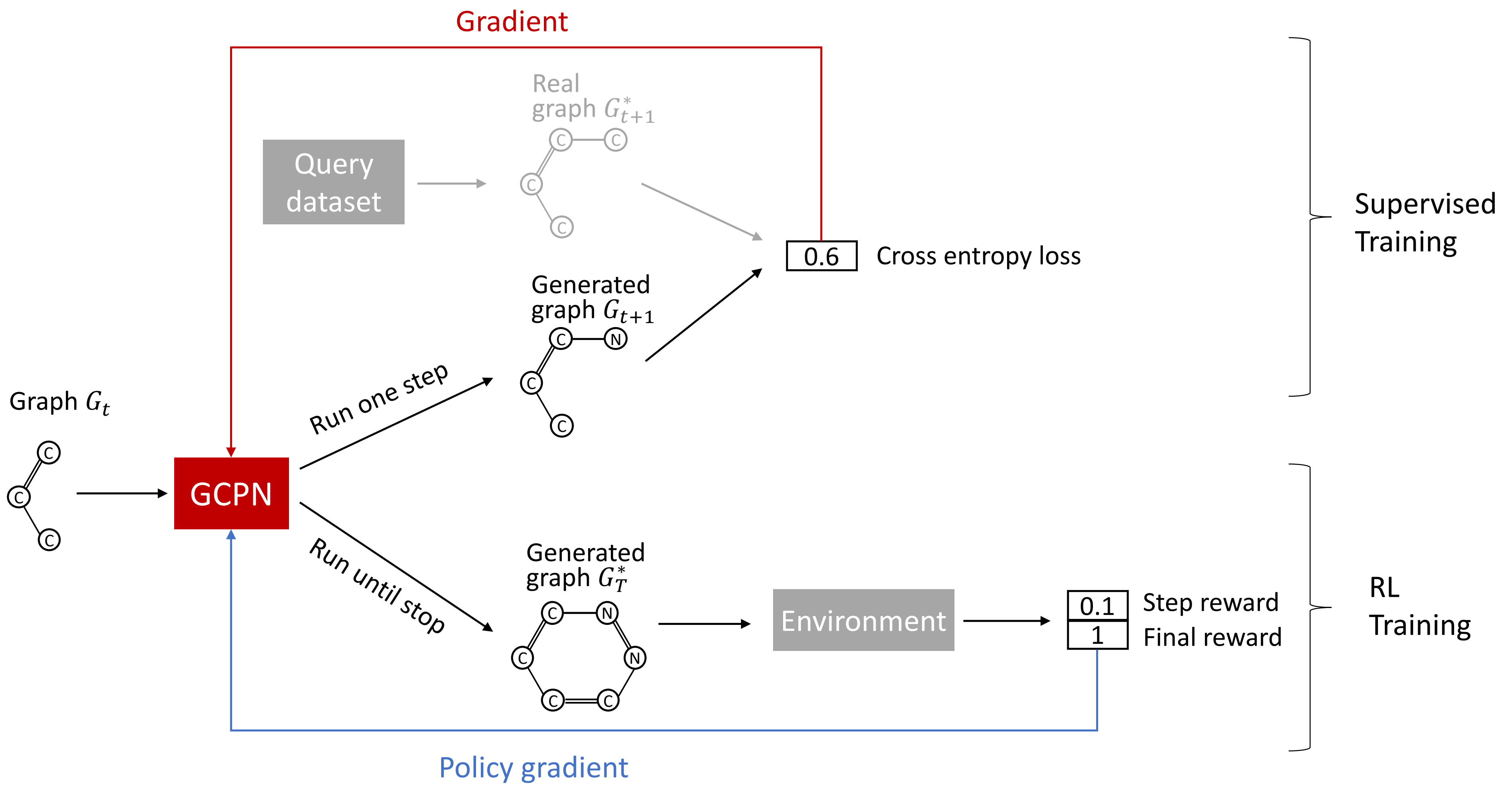
- Final reward = Domain-specific reward
- Step rewards = Step-wise validity reward

# How Do We Train?

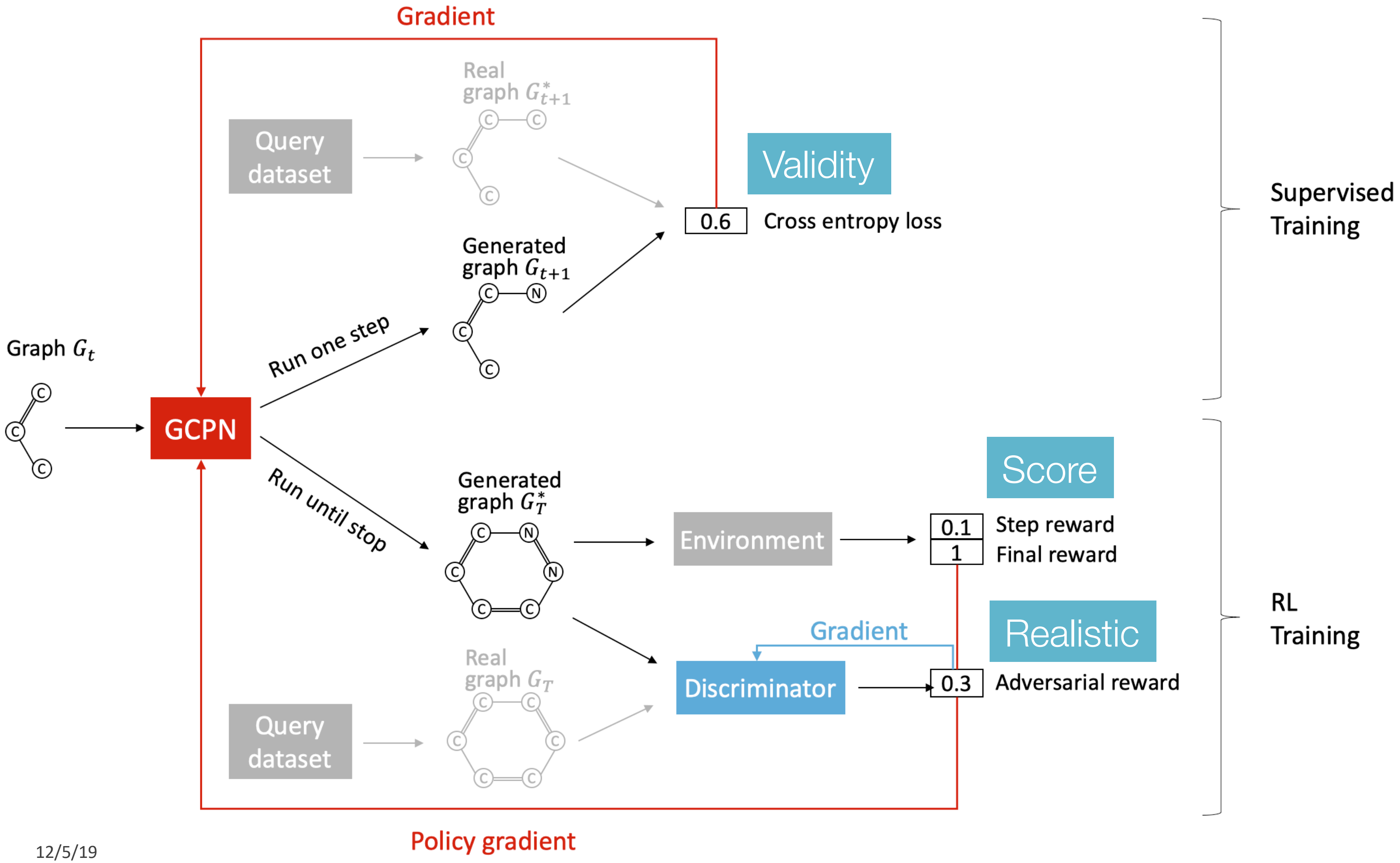


- **Two parts:**
- **(1) Supervised training:** Train policy by imitating the action given by real observed graphs. Use gradient.
- **(2) RL training:** Train policy to optimize rewards. Use standard policy gradient algorithm (refer to any RL course, e.g., CS234).

# GCPN Architecture



# GCPN Architecture



12/5/19

# GCPN: Tasks

- Property optimization
  - Generate molecules with high specified property score
- Property targeting
  - Generate molecules whose specified property score falls within given range
- Constrained property optimization
  - Edit a given molecule for a few steps to achieve higher specified property score

# Data and Baselines

- ZINC250k dataset
  - 250,000 drug like molecules whose maximum atom number is 38
- Baselines:
  - ORGAN: String representation + RL [Guimaraes et al., 2017]
  - JT-VAE: VAE-based vector representation + Bayesian optimization [Jin et al., 2018]

# Quantitative Results

## Property optimization

- +60% higher property scores

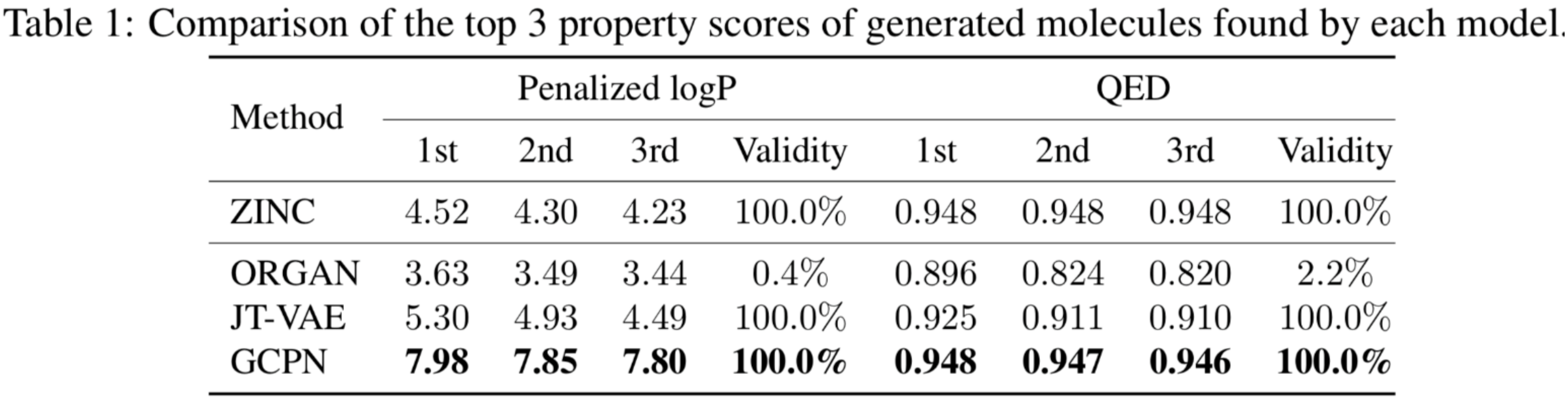


Table 1: Comparison of the top 3 property scores of generated molecules found by each model.

| Method | Penalized logP | | | | QED | | | |
|---|---|---|---|---|---|---|---|---|
| | 1st | 2nd | 3rd | Validity | 1st | 2nd | 3rd | Validity |
| ZINC | 4.52 | 4.30 | 4.23 | 100.0% | 0.948 | 0.948 | 0.948 | 100.0% |
| ORGAN | 3.63 | 3.49 | 3.44 | 0.4% | 0.896 | 0.824 | 0.820 | 2.2% |
| JT-VAE | 5.30 | 4.93 | 4.49 | 100.0% | 0.925 | 0.911 | 0.910 | 100.0% |
| GCPN | **7.98** | **7.85** | **7.80** | **100.0%** | **0.948** | **0.947** | **0.946** | **100.0%** |

**logP:** octanol-water partition coef., indicates **solubility**

**QED:** indicator of **drug-likeness**

# Quantitative Results

## Property targeting

- 7x higher success rate than JT-VAE, 10% less diversity

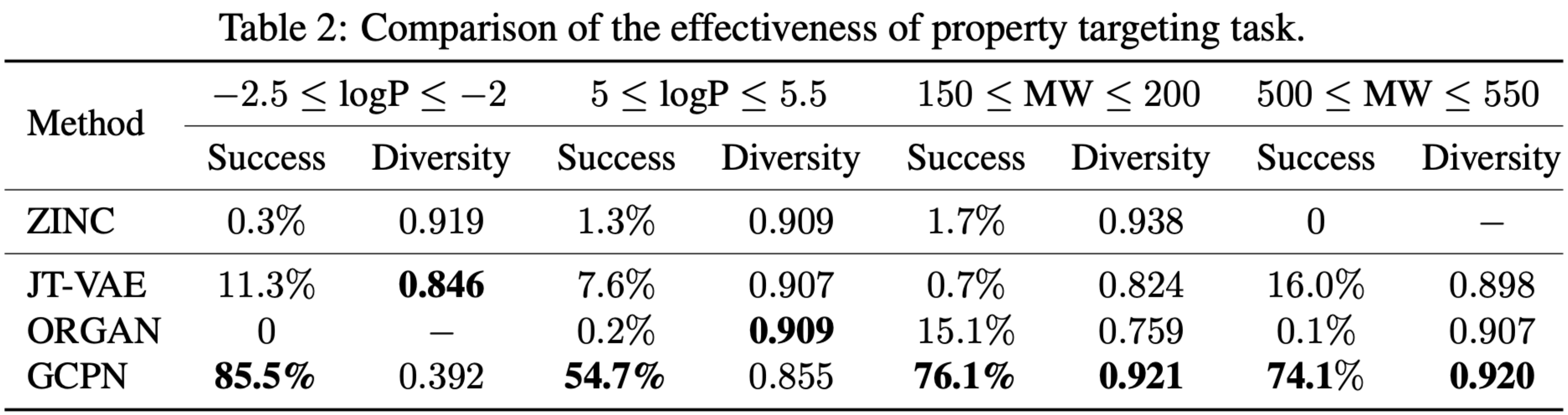


Table 2: Comparison of the effectiveness of property targeting task.

| Method | $-2.5 \leq \text{logP} \leq -2$ | | $5 \leq \text{logP} \leq 5.5$ | | $150 \leq \text{MW} \leq 200$ | | $500 \leq \text{MW} \leq 550$ | |
|---|---|---|---|---|---|---|---|---|
| | Success | Diversity | Success | Diversity | Success | Diversity | Success | Diversity |
| ZINC | 0.3% | 0.919 | 1.3% | 0.909 | 1.7% | 0.938 | 0 | – |
| JT-VAE | 11.3% | **0.846** | 7.6% | 0.907 | 0.7% | 0.824 | 16.0% | 0.898 |
| ORGAN | 0 | – | 0.2% | **0.909** | 15.1% | 0.759 | 0.1% | 0.907 |
| GCPN | **85.5%** | 0.392 | **54.7%** | 0.855 | **76.1%** | **0.921** | 74.1% | **0.920** |

**logP:** octanol-water partition coef., indicates **solubility**

**MW**: molecular weight an indicator of **drug-likeness**

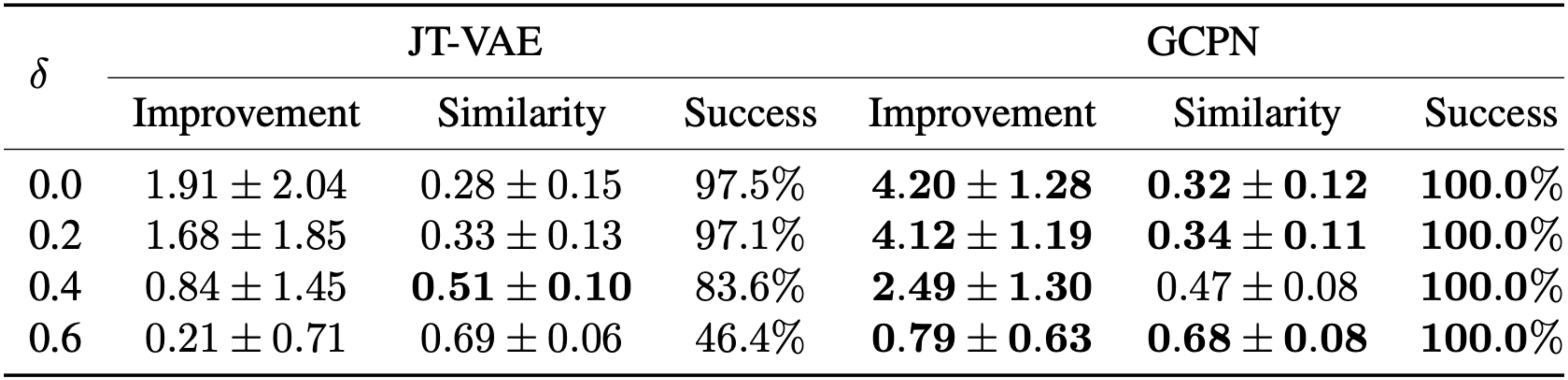
**Diversity**: avg. pairwise Tanimoto distance between Morgan fingerprints of molecules

# Quantitative Results

## Constrained property optimization

- +180% higher scores than JT-VAE

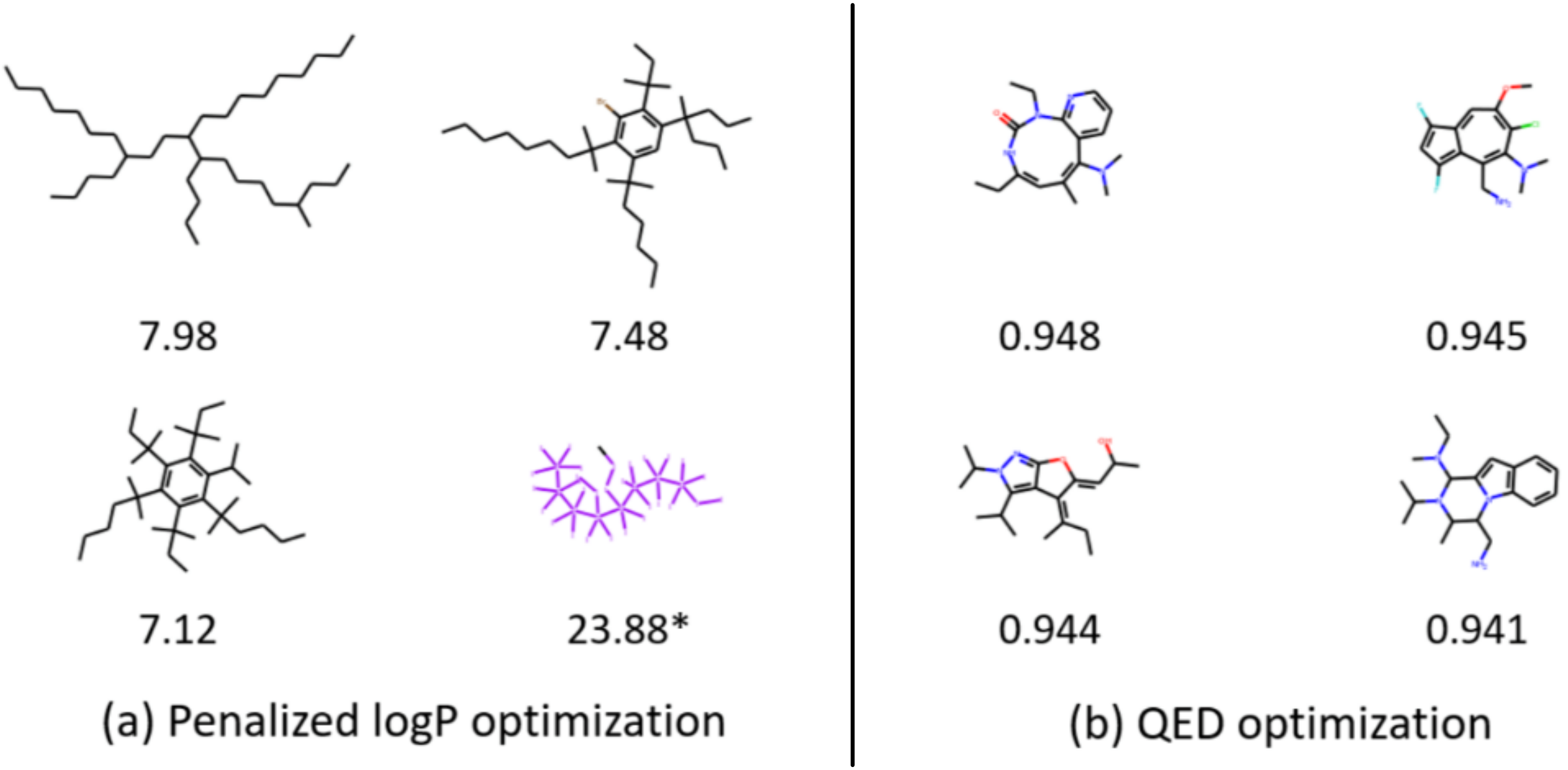
Table 3: Comparison of the performance in the constrained optimization task.



| $\delta$ | JT-VAE | | | GCPN | | |
|---|---|---|---|---|---|---|
| | Improvement | Similarity | Success | Improvement | Similarity | Success |
| 0.0 | 1.91 ± 2.04 | 0.28 ± 0.15 | 97.5% | **4.20 ± 1.28** | **0.32 ± 0.12** | **100.0%** |
| 0.2 | 1.68 ± 1.85 | 0.33 ± 0.13 | 97.1% | **4.12 ± 1.19** | **0.34 ± 0.11** | **100.0%** |
| 0.4 | 0.84 ± 1.45 | **0.51 ± 0.10** | 83.6% | **2.49 ± 1.30** | 0.47 ± 0.08 | **100.0%** |
| 0.6 | 0.21 ± 0.71 | 0.69 ± 0.06 | 46.4% | **0.79 ± 0.63** | **0.68 ± 0.08** | **100.0%** |

# Qualitative Results

## Visualization of GCPN graphs: Property optimization



(a) Penalized logP optimization

(b) QED optimization

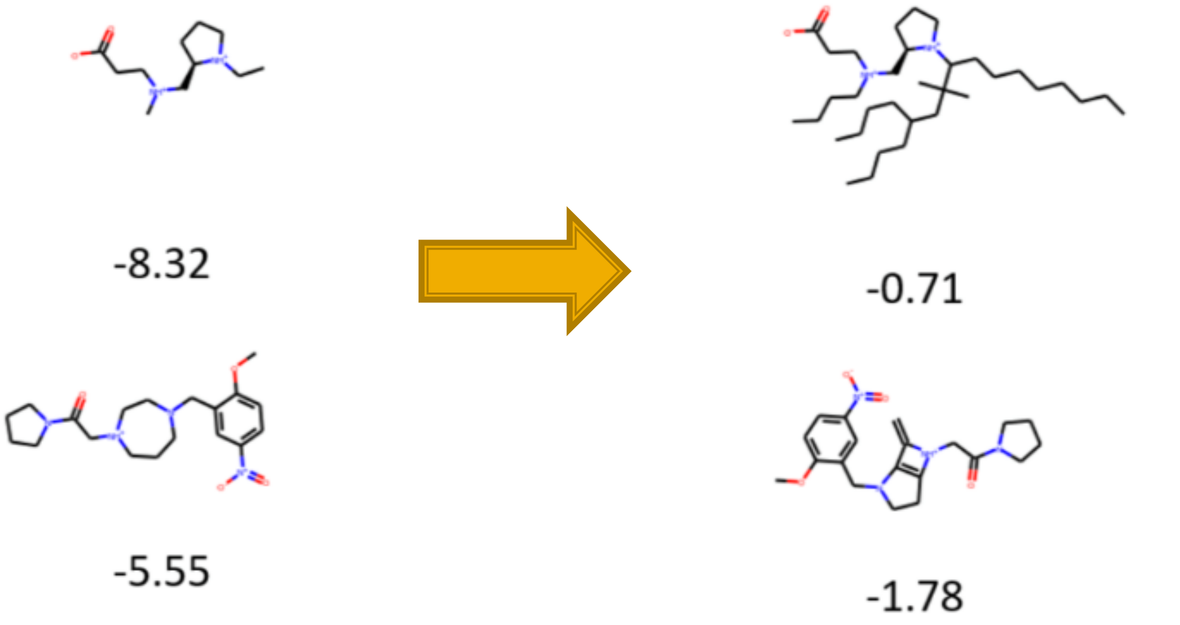
# Qualitative Results

## Visualization of GCPN graphs:

constrained optimization

Starting structure

Finished structure



(c) Constrained optimization of penalized logP

# Summary of Graph Generation

- Complex graphs can be successfully generated via sequential generation
- Each step a decision is made based on hidden state, which can be
  - Explicit: intermediate generated graphs, decode with GCN
  - Implicit: vector representation, decode with RNN
- Possible tasks:
  - Imitating a set of given graphs
  - Optimizing graphs towards given goals

# References

**PinSage:**

- Graph convolutional neural networks for web-scale recommender systems. R. Ying, R. He, K. Chen, P. Eksombatchai, W. Hamilton, J. Leskovec. *KDD 2018.*

**Decagon:**

- Modeling polypharmacy side effects with graph convolutional networks. Z., Marinka, M. Agrawal, J. Leskovec. *Bioinformatics* 2018.
- Website: http://snap.stanford.edu/decagon/

**GCPN:**

- Graph Convolutional Policy Network for Goal-Directed Molecular Graph Generation. J. You, B. Liu, R. Ying, V. Pande, J. Leskovec. *NeurIPS 2018.*
- Code: https://github.com/bowenliu16/rl_graph_generation

# What Next?

- **Project write-ups:**
  - Tue Dec 10 **(11:59PM)** Pacific Time
    - **1 team member uploads PDF to Gradescope**
    - **Don’t forget to tag your other team members!**
- **Poster session:**
  - Thu Dec 12, 12:15 – 3:15 pm in **Huang Foyer**
    - All groups with at least one non-SCPD member must present
    - There should be 1 person at the poster at all times
    - **Prepare a 2-minute elevator pitch of your poster**
    - **More instructions on Piazza**

No late days!

# What Next? Our Courses

- **CS246: Mining Massive Datasets (Winter 2020)**
  - Data Mining & Machine Learning for Big Data
    - (big==doesn't fit in memory/single machine), SPARK

- **CS341: Project in Data Mining (Spring 2020)**
  - Groups do a research project on Big Data
  - We provide interesting data, projects and **access to the Google Cloud infrastructure**
  - Nice way to finish up CS224W project & **publish it**!

# What Next?

- **Conferences / Journals:**
  - **KDD**: Conf. on Knowledge Discovery & Data Mining
  - **ICML:** Intl. Conf. on Machine Learning
  - **NeurIPS:** Neural Information Processing Systems
  - **ICLR:** Intl. Conf. on Learning Representations
  - **WWW**: ACM World Wide Web Conference
  - **WSDM**: ACM Web search and Data Mining
  - **ICWSM**: AAAI Int. Conf. on Web-blogs & Social Media
  - **Journal of Network Science**
  - **Journal of Complex Networks**

# What Next? Other Courses

- **Other relevant courses:**
  - **CS229**: Machine Learning
  - **CS230**: Deep Learning
  - **MSE231:** Computational Social Science
  - **MSE334:** The Structure of Social Data
  - **CS276**: Information Retrieval and Web Search
  - **CS245**: Database System Principles
  - **CS347**: Transaction Processing & Databases

# Thank you Michele and TAs!!

**Teaching Assistants**

Christina Yuan
Head TA

Lingzi (Liz) Guo

Benjamin (Ben) Hannel

Kuangcong (Cecilia) Liu

Vasco Portilheiro

Andrew Wang

Alexis Goh Weiying

Zhitao (Rex) Ying

**Co-Instructor**

Michele Catasta

**Thank You**

# In Closing...

- You Have Done a Lot!!!
- And (hopefully) learned a lot!!!
  - Answered questions and proved many interesting results
  - Implemented a number of methods
  - **And are doing excellently on the class project!**

**Thank You for the Hard Work!!!**

Eve Online: Exodus developer CCP publisher CCP
© GameWallpapers.com hosted by JTLnet.com