11-741/11-441:

Machine Learning with Graphs

Introduction

Yiming Yang

1

Outline

- Administrative Stuff
- Course Contents Overview

2

Instructor and Teaching Assistants

- Yiming Yang (LTI & MLD)
- Office hours: Tue 12:45pm to 1:45pm, GHC 5703 or via zoom

- Shengyu Feng (PhD student in LTI)
- Office hours: See piazza

- Ruohong Zhang (PhD student in LTI)
- Office hours: See piazza

Background

- **Prerequisites**
  - CS courses like data structures, algorithms, programming (e.g., 15-213)
  - Linear algebra (e.g., 21-241 or 21-341), introductory probability (e.g., 21-325)
- **Preferred but not required**
  - Introductory Machine Learning (e.g., 10-701 or 10-601)
  - Neural network courses
  - This course is mostly self-contained on ML background

Sections

- **11-741** (graduate level, 12 units)
  - Students need to do 100% homework (5 HWs) & 100% of the Exam Questions (midterm exam and final exam)
- **11-441** (undergraduate level, 9 units)
  - Students need to do 80% (4 out of of the total 5) of the HW assignments by your own choices; if you do all the 5, then the top-4 scores will be used in grading.
  - Students need to do 70% of the total exam questions by your own choices; if you choose to do more, only the 70% of the best answered questions will be used in grading.

5

Grading Policies

	11-741	**11-441**
Mid-term Exam	15%	15%
Final Exam	15%	15%
Homework Assignments	14% x 5 =70%	17.5% x 4 = 70%

6

Homework

- Programming assignments in Python
- Submission: Due by 11:59pm of the due date.
  - It must be submitted by Gradescope. If Gradescope is down, it must be submitted by email to the TA.
  - A 10% penalty is applied for each day beyond the deadline.
- Grace Days for HW Submissions
  - Each student will have 5 grace days in total over the semester
  - Grace days cannot be applied to the last homework HW5) due to the tight window for grading.

Cheating, Copying, Plagiarism, Etc

- You must be the author of everything that you submit for a grade
- Revising or modifying someone else's work does not make you the author
- It is okay to discuss homework with other students, share ideas, experience, and lessons learned
- Sign the cheating policy form (as the condition to be graded)

Exams

- Exams will be on paper, in classroom; no electronic devices will be allowed (e.g., using ChatGPT is not allowed); paper nodes are allowed.
- Open book, with a set of questions (about 10) and a list of possible answers to choose from per question.
- Mid-term exam will cover the 1st half of the lecture contents of the semester, and the final exam will cover the 2nd half.
- The exams will not focus on the contents of the HW assignments.
- **No arrangement**: if you cannot attend the exams, you will just lose the points

Course Materials Online

- **Syllabus** (publicly available)

  http://la.lti.cs.cmu.edu/classes/11-741/f24/Syllabus.pdf
- **Lecture Slides** (password protected)
  - URL in the syllabus web page (above of the lecture schedule)
  - Login information will be announced via piazza
- **Piazza** (listed at Canvas)
- **Recorded Lectures**
  - Not provided in general, to encourage in-person classes
  - Exceptions (if you catch COVID) can be arranged via the TA

Outline

✓ Administrative Stuff

- Course Contents Overview

Part I. Fundamental Building Blocks

- ❑ Deep Learning (6 lectures), HW1. CNN & RNN classifiers
  - Word2vec Embedding Methods
  - Recurrent Neural Networks (RNN)
  - Convolution Neural Networks (CNN)
  - Attention Models
  - LLM Architectures
  - Scalable Alignment of LLMs
- ❑ Classification Fundamentals (3 lectures), HW2, Soft-max & SGD

Part II. Graph-based Learning Topics

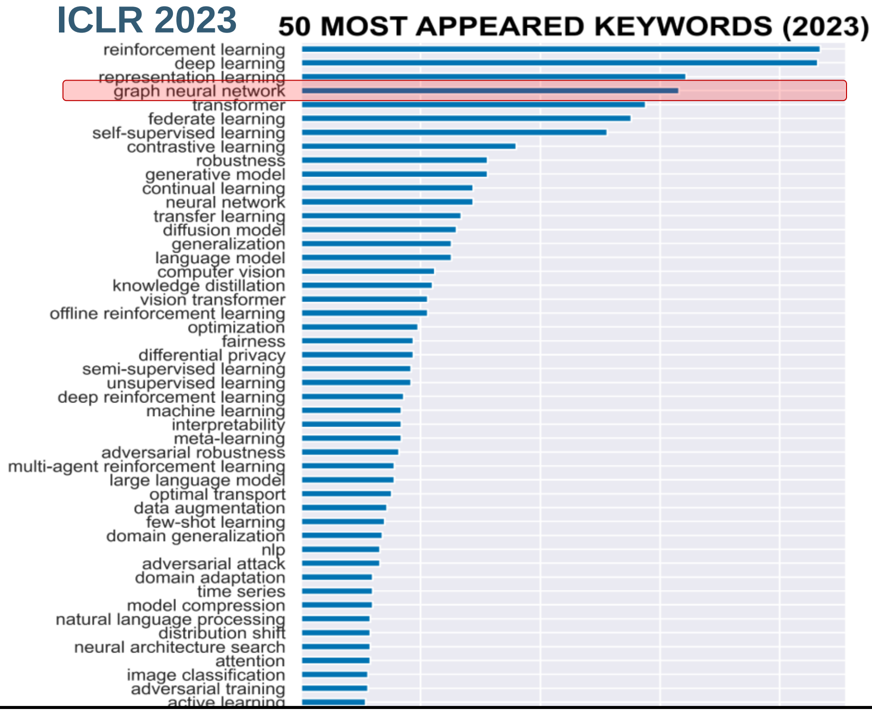
- Social Impact Analysis (3 lectures)
    - HW3. PageRank models
- Graph Neural Networks, etc. (5-6 lectures)
    - HW4. GCN models
- Knowledge Graph Embedding (2 lectures)
    - HW5. Node Embedding with TransE
- Neural Solvers for NP-Complete Problems (4 lectures)

@Yiming Yang, 11-741 F24 Introduction to ML_with_Graphs

13

13

ICLR 2023

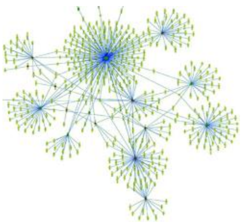


14

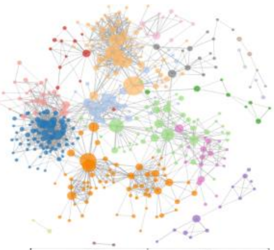
Why graphs?

A common language for representing various types of data

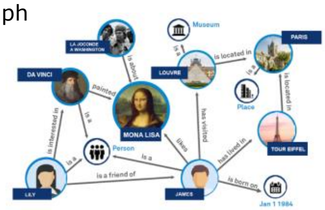
Hyperlinked Web Pages



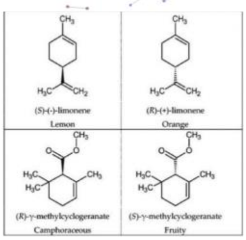
Citation Graph



Knowledge Graph

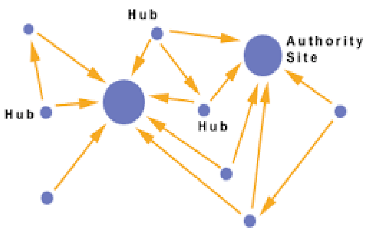


Molecular Structures



Ex 1. Social Impact Analysis

- Which web pages are more trustworthy (good authorities)?
- Which web pages are more resourceful (good hubs)?
- Naïve answer: Counting the input or output degrees of each node
- Better definition: A node is authoritive if it is pointed by many good hubs; similarly, a node is a good hub if it points to many authorities.
- **A chicken-egg problem!**



Ex 1: How to handle the chicken-egg problem?

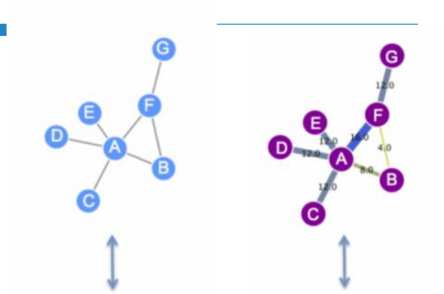
- **HITS (J. Kleinberg, 1998)**
  - Randomly assign a score to each node as its initial hub.
  - Use the current hub scores of all the nodes to calculate their authority scores;
  - Use the current authority scores of all the nodes to calculate their hub scores;
  - Repeat the above two steps until all the scores no longer change (typically, after 10-20 iterations).
- But why should the scores converge? Where do they converge? Would the final scores depend on the random initialization?

Convergence of the scores

- **HITS (J. Kleinberg, 1998)**
  - The final authority/hub scores of nodes depend on the graph structure only and the initialization has no effect on those scores.
  - Denote by $\boldsymbol{x}^{(t)} \in R^n$ the vector of the authority scores of all nodes in iteration $\boldsymbol{t}$, and by $\boldsymbol{y}^{(t)} \in R^n$ the corresponding hub-score vector.
  - It can be proven that $\boldsymbol{x}^{(t)}$and $\boldsymbol{y}^{(t)}$ converge to the 1st eigenvectors of matrices $A^T A$ and AA^T, respectively, when t is sufficiently large.
- **PageRank (S. Brin and L. Page, 1998)**
  - Conceptually similar but using some different matrix formulation.
  - We will learn more about these methods in our lectures.

Representing the Graph Structure

- **Graph** $G = (V, A)$
  - V is the set of n vertices.
  - A is the adjacency matrix with $n \times n$ elements as the edge weights, which can be eighter binary-valued (left) or weighted (right).

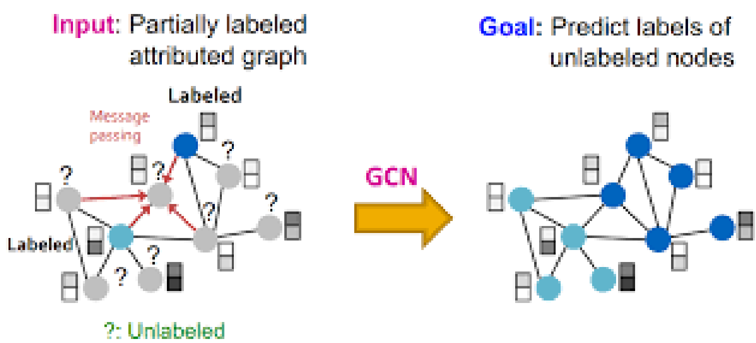


	A	B	C	D	E	F	G
A	0	1	1	1	1	1	0
B	1	0	0	0	0	1	0
C	1	0	0	0	0	0	0
D	1	0	0	0	0	0	0
E	1	0	0	0	0	0	0
F	1	1	0	0	0	0	1
G	0	0	0	0	0	1	0

	A	B	C	D	E	F	G
A	0	8	12	12	12	16	12
B	8	0	0	0	0	4	0
C	12	0	0	0	0	0	0
D	12	0	0	0	0	0	0
E	12	0	0	0	0	0	0
F	16	4	0	0	0	0	12
G	12	0	0	0	0	12	0

Adjacency matrices

Ex 2. Graph Convolutional Network (GCN)
for Semi-supervised Classification [ICLR 2017]



Consider a citation graph where the nodes are documents: some nodes are labeled, and the rest nodes need to be labeled.

Graph Convolutional Network (GCN)

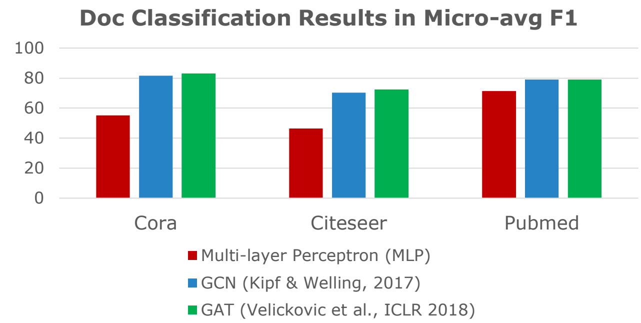
- GCN uses a multi-layer neural network to optimize node embedding with respect to semi-supervised node classification.
- Graph convolution means to aggregate node features or embeddings from the direct neighborhood of each node.
- With multiple layers together, GCN enables higher order propagation from the multi-hop neighborhood of each node.
- Therefore, GCN produces smooth changes in node embedding if connected, which in tern yields smooth label propagation over the graph.

Various Graph Neural Networks

- Graph Convolution Network (GCN) [ICLR 2017]
- Graph Attention Network (GAT) [ICLR 2018)
- Graph Isomorphism Network (GIN) [ICLR 2019]
- Graphormer [NeurIPS 2021]
- Neural Graph Collaborative Filtering (NGCF) [SIGIR 2019]

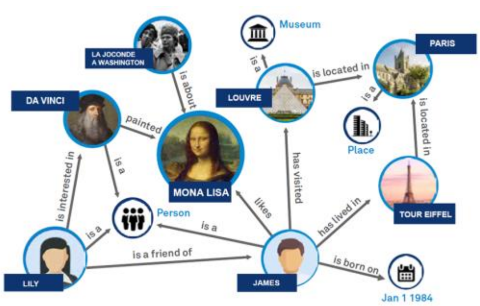
All the methods are rooted in recently deep learning techniques.

Results of MLP, GCN and GAT
(Velickovic et al., ICLR 2018)



Method	Cora	Citeseer	Pubmed	Word Embedding	Citation Links	Link Weighting
Multi-layer Perceptron (MLP)	55.1	46.5	71.4	✓		
GCN (Kipf & Welling, 2017)	81.5	70.3	79	✓	✓	
GAT (Velickovic et al., ICLR 2018)	83	72.5	79	✓	✓	✓

Ex 3. Knowledge Graph (KG) based Reasoning



- KG consists of heterogeneous entities & Relations
- Naïve belief aggregation could be misleading.
- E.g., we may obtain similar profiles (embeddings) for Da Vinci and Mona Lisa, which is rather silly.
- E.g., our system may predict Mona Lisa being born in 1984 because James is a direct neighbor.

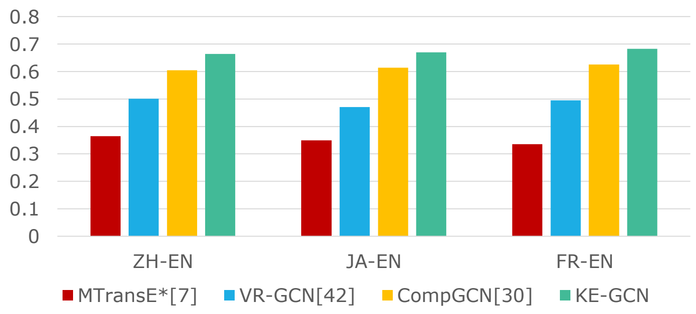
- We must discriminate edge differences for effective belief propagation.

Knowledge Embedding based GCN (WWW 2021)

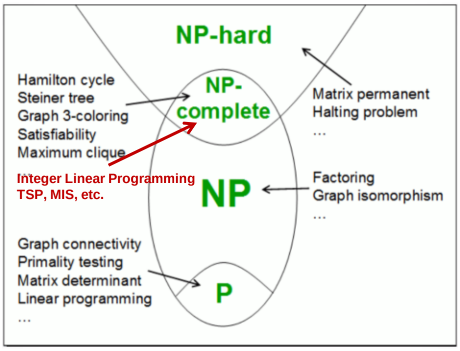
- **Main Observation**
  - Conventional GCNs (focus on node embedding only) well on homogeneous graphs (e.g., citation graphs) but may not be good enough for heterogeneous graphs (e.g., knowledge graphs)
- **Remedy**
  - Introducing edge embedding in addition to node embedding
  - How: Multiple ways to do it (in our lectures)

Results in Cross-language Entity Alignment

- **Task:** To align KG entities across English (EN), Japanese (JA), French (FR) and Chinese (ZH), e.g., Biden ←→ 拜登
- **Metric**: Mean Reciprocal Rank (MRR: higher is better)

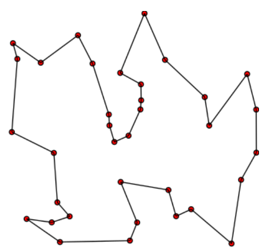


Ex4. Graph-based Learning for NP-Complete Problems

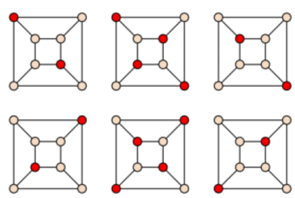


- **NP Complete** is the class of hardest problems in CS**.**
- **We are not trying** to prove if NP == P.
- **We interested in using** cutting-edge neural networks to better solve NP-complete problems through approximation.

Examples of NPC problems: TSP & MIS



Traveling Salesman Problem (TSP) : Find a tour in the graph which visits each node once and only ones and return to the starting node at the end.



Maximal Independent Set (MIS):
Find the maximum subset of mutually independent nodes.

How can cutting-edge neural learning help?

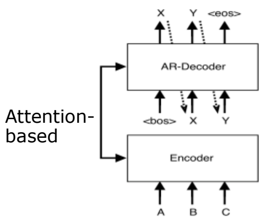
- Most NPC problems can be defined as to search over a graph for an optimal solution.
- A candidate solution can be generated by sequentially selecting a variable (node or edge) one-at-a-time under certain constraints.
- This reminds us about the generative process in LMs and Graphical Neural Networks (GNNs).
- We have successfully developed the SOTA neural solvers for several NPC problems in both prediction accuracy and scalability.

SOTA Neural NPC Solvers

- Before 2022, GNN-based solvers only scaled to TSP graphs with $n = 100$ nodes.
- **Our DIMES** [NearIPS 2022a], a DRL-based GNN solver, scaled to TSP graphs of $n = 10{,}000$ nodes with the best results on evaluation benchmarks.
- **Our DIFUSCO** [ICML 2023], the first graph-based diffusion model for NPC, outperforming DIMES both in scalability and accuracy.

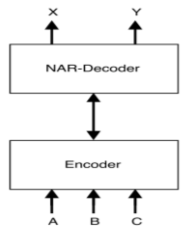
Two types of language models: AR vs. NAR

Autoregressive (AR) LM



- Encoder-Decoder Neural Networks
- Scaled to NPC graphs with n=100 nodes (ICLR 2017, ICLR 2019)

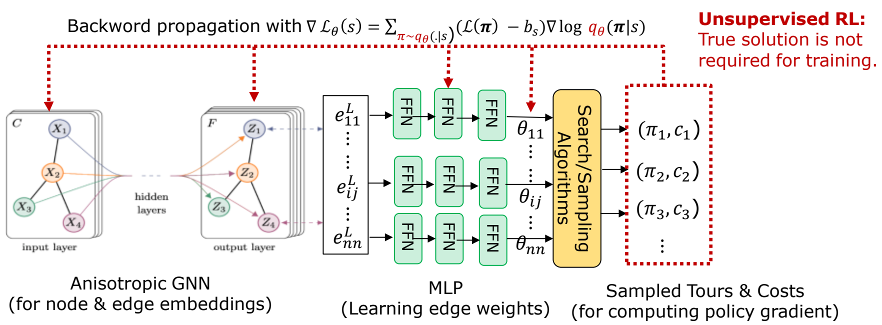
Non-autoregressive (NAR) LM



- Encoder + Active Search (not NNet)
- Scaled to NPC graphs with n=10000 nodes plus better results (NeurIPS 2022, ICML 2023)

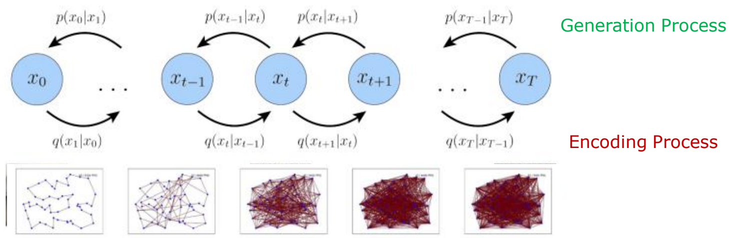
31

DIMES: Solving NPC with GNN+MLP+Sampling



32

Diffusion modeling for generating good TSP tours



Continuous modeling (Gaussian) for images → Discrete modeling Bernoulli) for graphs

Concluding Remarks

- **Why graphs?**
  - A common language for representing many types of entities, relations and human knowledge, supporting graph-based reasoning beyond bag/sequence of tokens
  - Broad applications (social impact analysis, classification, regression, recommendation, combinatorial optimization, etc.)
- **Connections to recent deep learning**
  - Novel adaptation and enrichment of popular LLMs, Deep Reinforcement Learning (DRL) and Diffusion Models