# A Hands-on Introduction to Graph Deep Learning, with Examples in PyTorch Geometric - IIII

Machine Learning and Dynamical Systems Seminar

November 23, 2023

Gabriele Santin (gabrielesantin.github.io)
Antonio Longa (antoniolonga.github.io)
Steve Azzolin (steveazzolin.github.io)
**Francesco Ferrini** (francescoferrini.github.io)

# Introduction
# About us

Gabriele Santin
Assistant professor at University of Venice (Venice, Italy)
gabriele.santin@unive.it
https://gabrielesantin.github.io/

Antonio Longa
Assistant professor University of Trento (Trento, Italy)
antonio.longa@unitn.it
https://antoniolonga.github.io/

Steve Azzolin
ELLIS PhD Student at FBK and University of Trento (Trento, Italy)
steve.azzolin@unitn.it
https://steveazzolin.github.io/

**Francesco Ferrini**
PhD Student at University of Trento, (Trento, Italy)
francesco.ferrini@unitn.it
https://francescoferrini.github.io/

# Introduction
## Organization and material

Tutorial in four parts (slides + Jupyter notebooks available at link):

- **Part I:** November 2, Presenter: **GS**
  Goals: Motivations, Intro of basic concepts, definition of GNNs
- **Part II:** November 9, Presenter: **AL**
  Goals: Implementation of GNNs: How to implement a full GNN pipeline in PyTorch Geometric.
- **Part III:** November 16, Presenter: **SA**
  Goals: Explainability of GNNs: How to inspect a model to try to understand the learned decision pattern.
- **Part IV:** November 23, Presenter: **FF**
  Goals: Heterogeneity in GNNs: How can GNNs effectively model and incorporate a diversity of nodes and edges with different types.

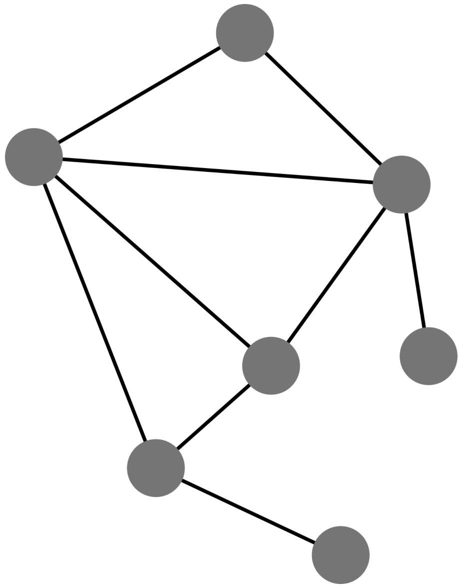
# Introduction
## Outline

1. Heterogeneous graphs
2. Graph Neural Networks
3. Relational (Heterogeneous) Graph Convolution
4. Meta-paths
5. Knowledge Graphs

# Introduction
## Homogeneous graphs

So far graphs with a single edge type (a.k.a homogeneous graph)

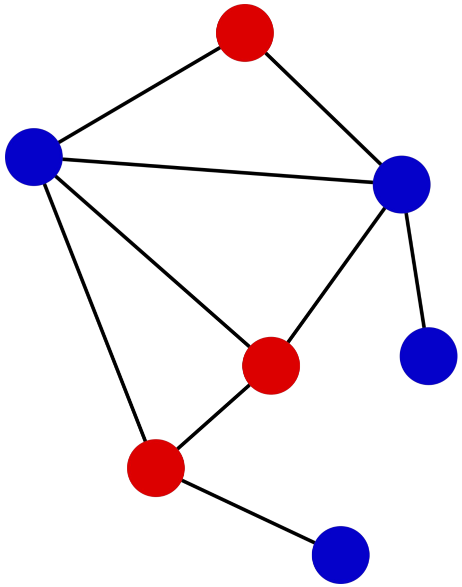
But are all graphs of this type?



# Heterogeneous graphs
## Node types

Graphs might have multiple node types:

- Node type A: Paper nodes
- Node type B: Author nodes



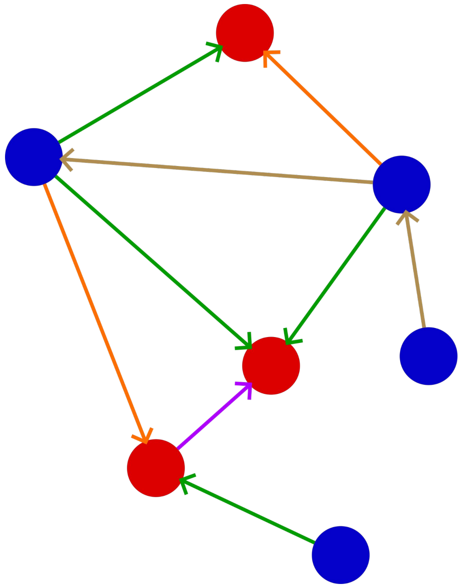
# Heterogeneous graphs
## Node and edge types

Graphs might have multiple node types:

- Node type A: Paper nodes
- Node type B: Author nodes

And multiple relation types:

- (Author, writes, Paper)
- (Author, likes, Paper)
- (Author, collaborates_with, Author)
- (Paper, cites, Paper)



# Heterogeneous graphs
## Definition

A Heterogeneous graph is defined as:

$$\mathcal{G} = (\mathcal{V}, \mathcal{E}, \tau, \phi)$$

Where:

- $\mathcal{V}$ is the set of nodes
- $\mathcal{E}$ is the set of edges

→ Homogeneous

- $\mathcal{T}$ is a mapping function from a node to its specific type
- $\phi$ is a mapping function from an edge to its specific type

→ Heterogeneous

# Heterogeneous graphs
## Real world heterogeneous graphs

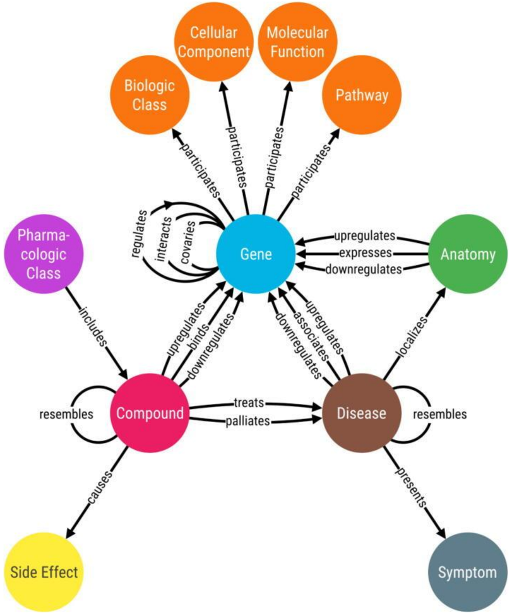



Image from "Constructing knowledge graphs and their biomedical applications"

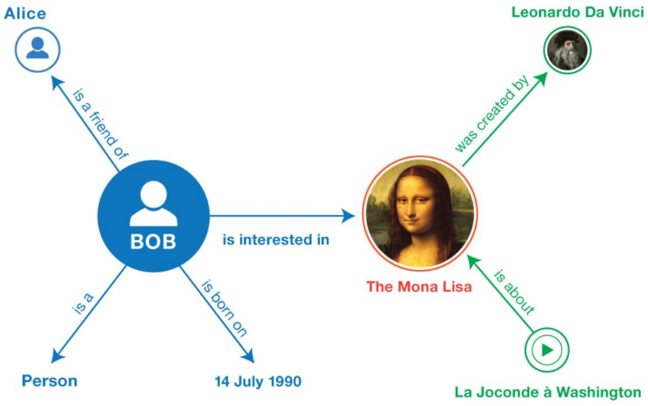



Image from "RDF Primer"

# Heterogeneous graphs
## Real world heterogeneous graphs

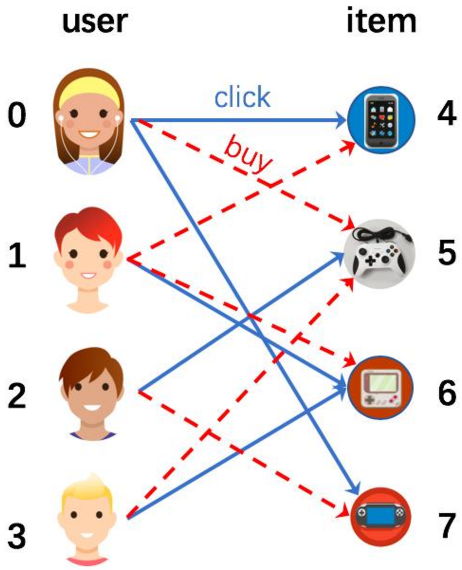



Image from Paddle Graph Learning library

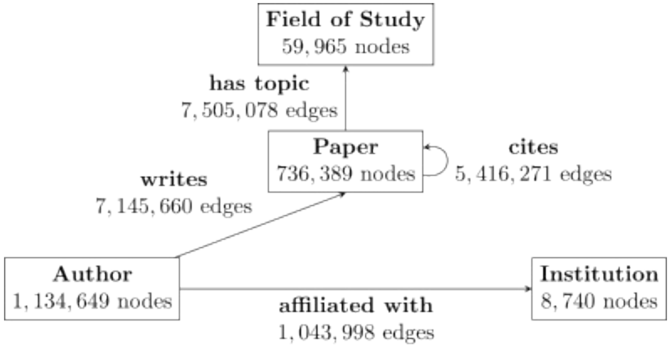
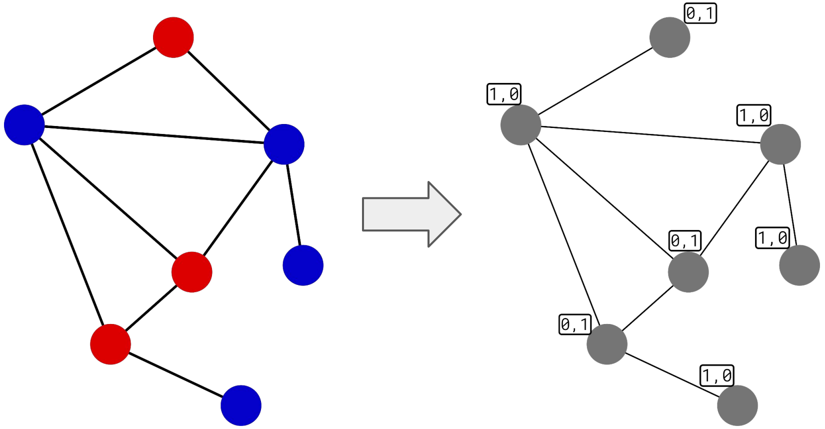



Image from Pytorch Geometric Library

# Heterogeneous graphs
## Type vs. feature

Can we treat node and edge types as features? Yes




We have transformed an heterogeneous graph in an homogeneous one where each node has a feature vector indicating the node type

# Heterogeneous graphs
## Type vs. feature

When is it useful to consider an heterogeneous graph?

- Different node types have different feature vector length (e.g. author node has a feature vector of length 10 while paper node has a feature vector of length 20)
- Different relation types represent different type of interactions (we need different models)

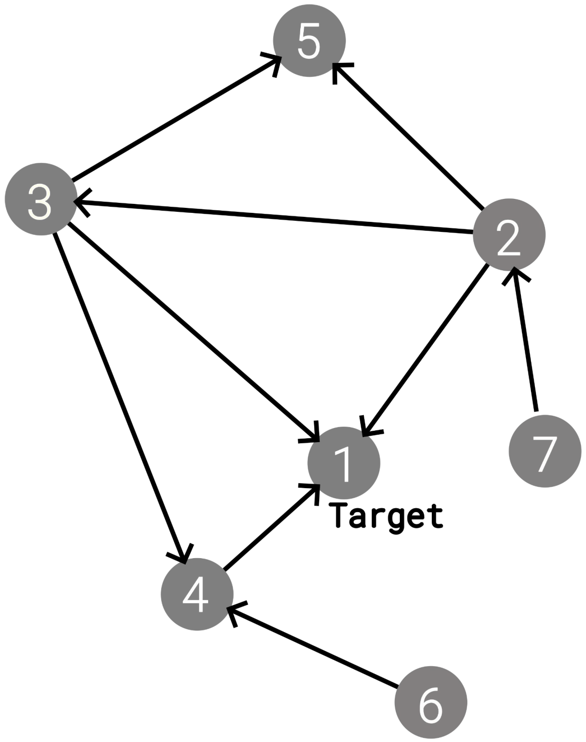
# Heterogeneous graphs
## Type vs. feature

When is it useful to consider an heterogeneous graph?

- Different node types have different feature vector length (e.g. author node has a feature vector of length 10 while paper node has a feature vector of length 20)
- Different relation types represent different type of interactions (we need different models)

Heterogeneous graphs are more expressive!

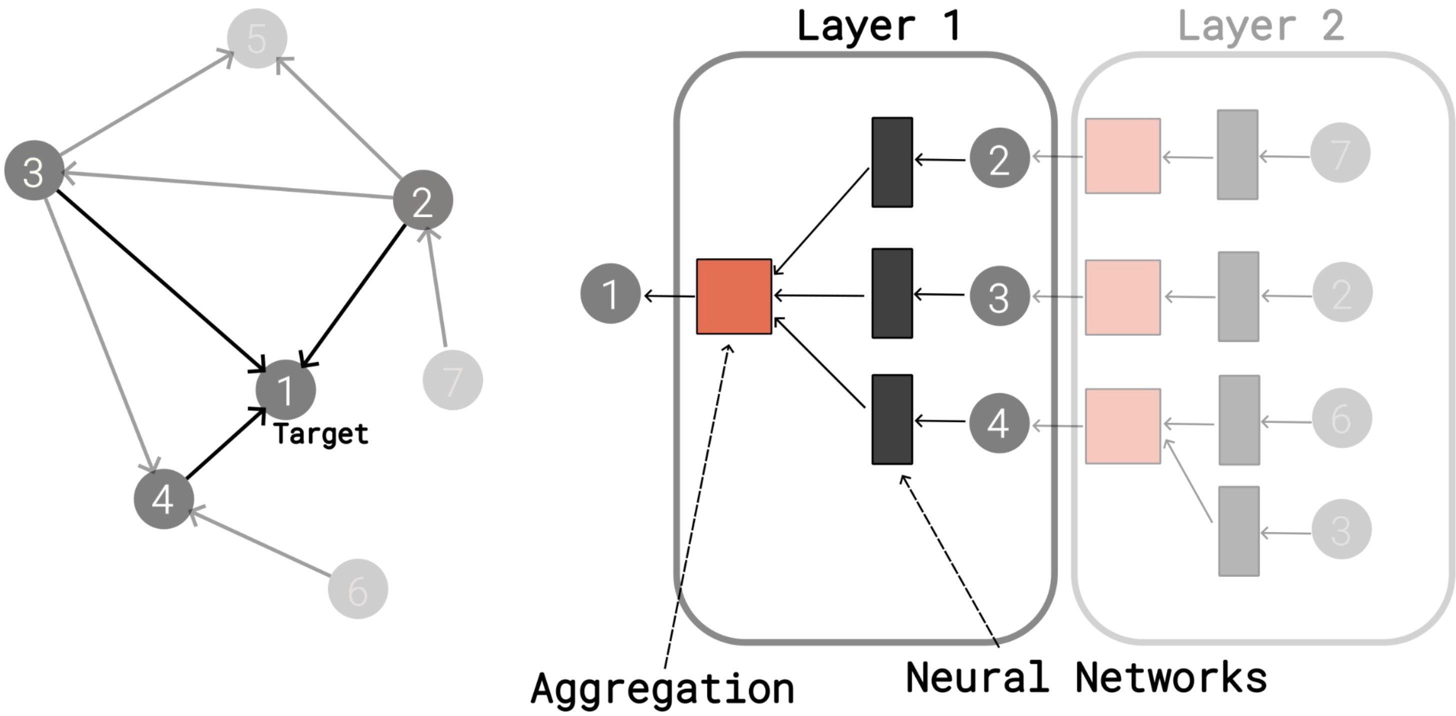
# Graph Neural Networks
## Recap: Graph Convolution



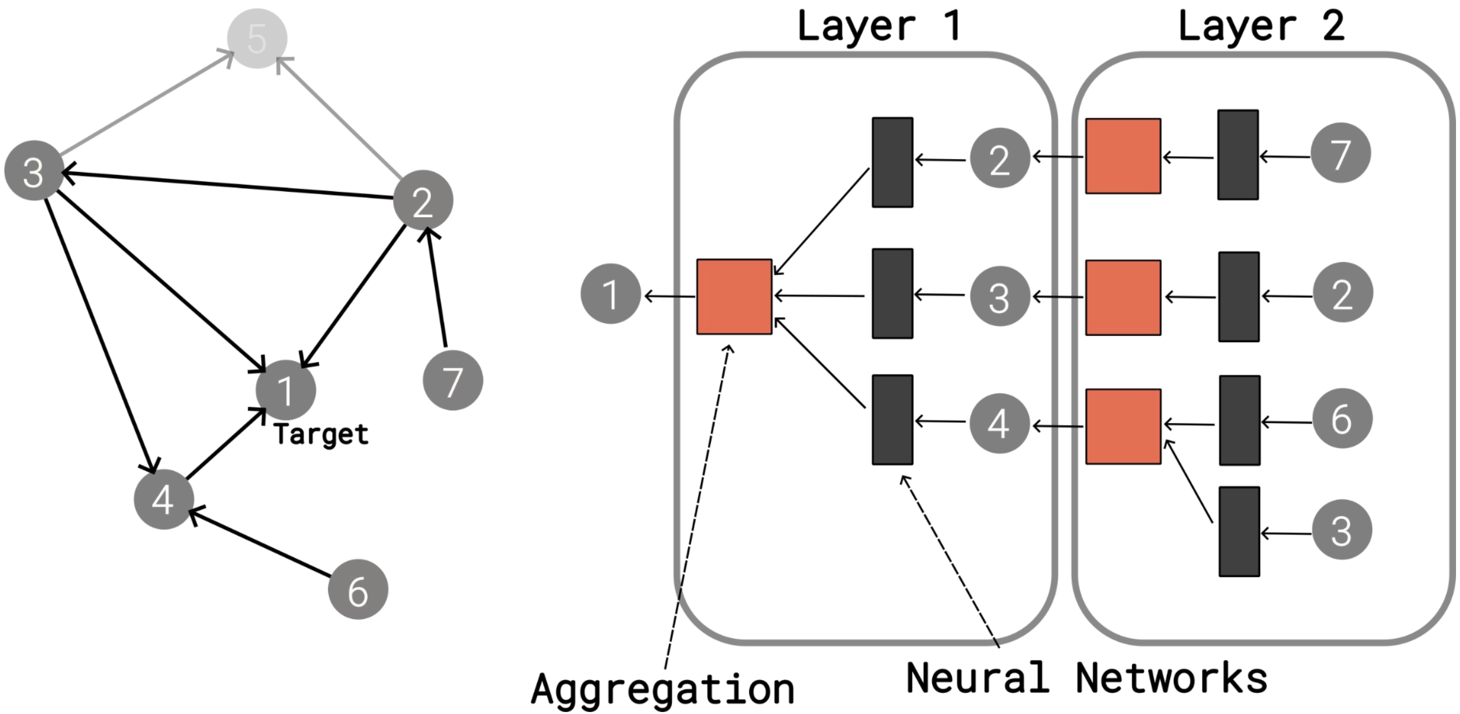

Homogeneous input graph

We want to run a GCN to update the representation of target node 1

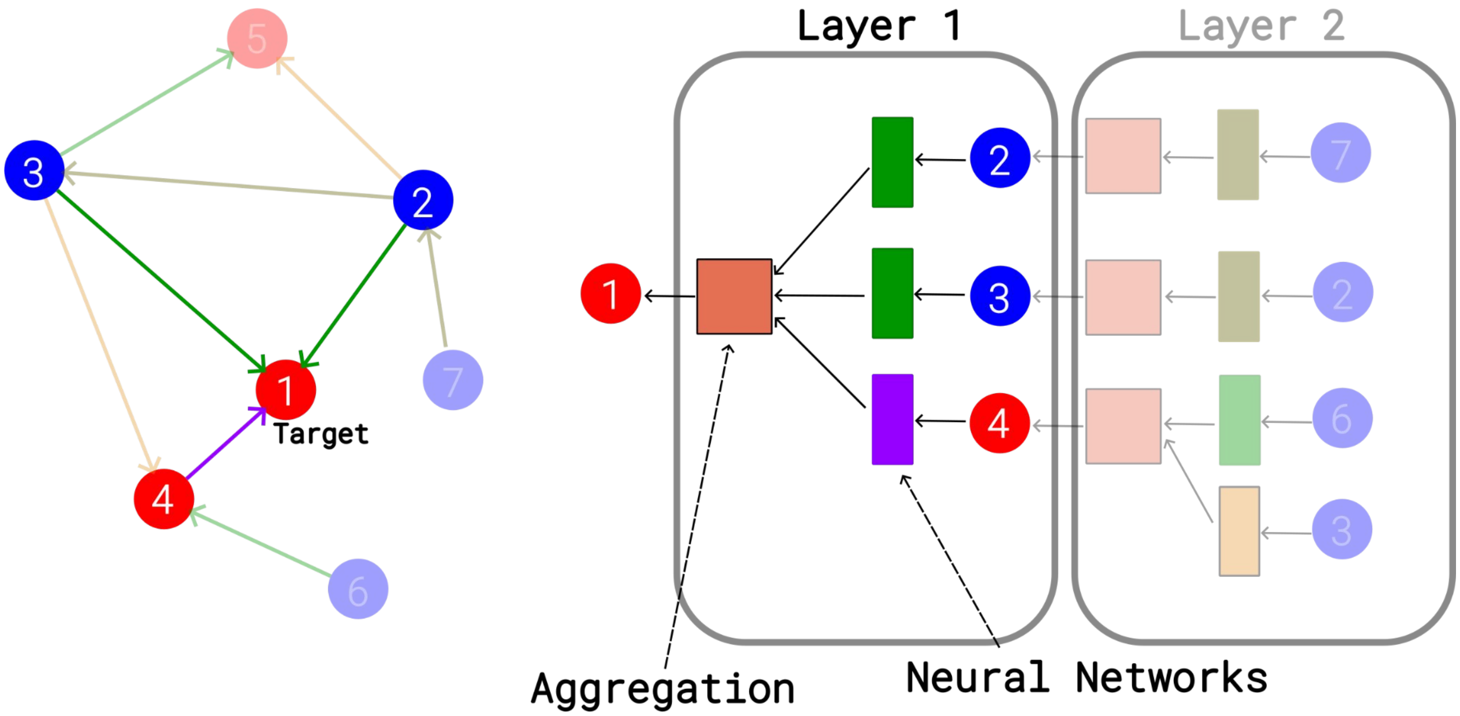
# Graph Neural Networks
## Recap: Graph Convolution



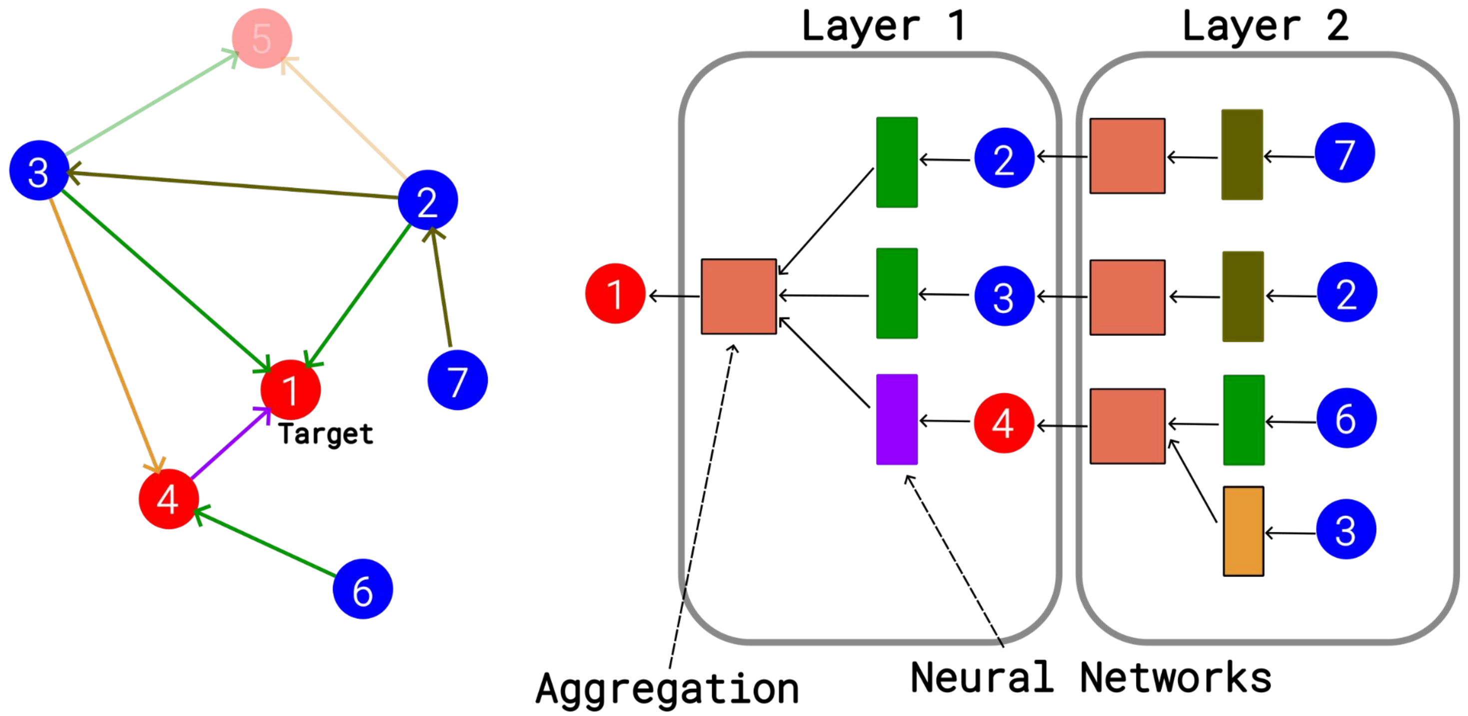
# Graph Neural Networks
## Recap: Graph Convolution



# Graph Neural Networks
## Relational (Heterogeneous) Graph Convolution



# Graph Neural Networks
## Relational (Heterogeneous) Graph Convolution



# Relational (Heterogeneous) Graph Convolution
## Introduction

At this point there is one weight matrix for each relation type

Weights for relation 1

...

Weights for relation N

Weights for self-loop

Self loop

$$h_i^{(l+1)} = \sigma\left(h_i^{(l)} W_0^{(l)} + \sum_{r\in\mathcal{R}} \sum_{j\in\mathcal{N}_i^r} h_j^{(l)} W_r^{(l)}\right)$$

New representation for target node

Parameters specific for relation type

# Relational (Heterogeneous) Graph Convolution
## Scalability

Problem: Numerous types of relationships lead to a rapid growth in the number of parameters

Solution:

- Block diagonal matrices
- Basis learning

# Relational (Heterogeneous) Graph Convolution
## Scalability: basis learning

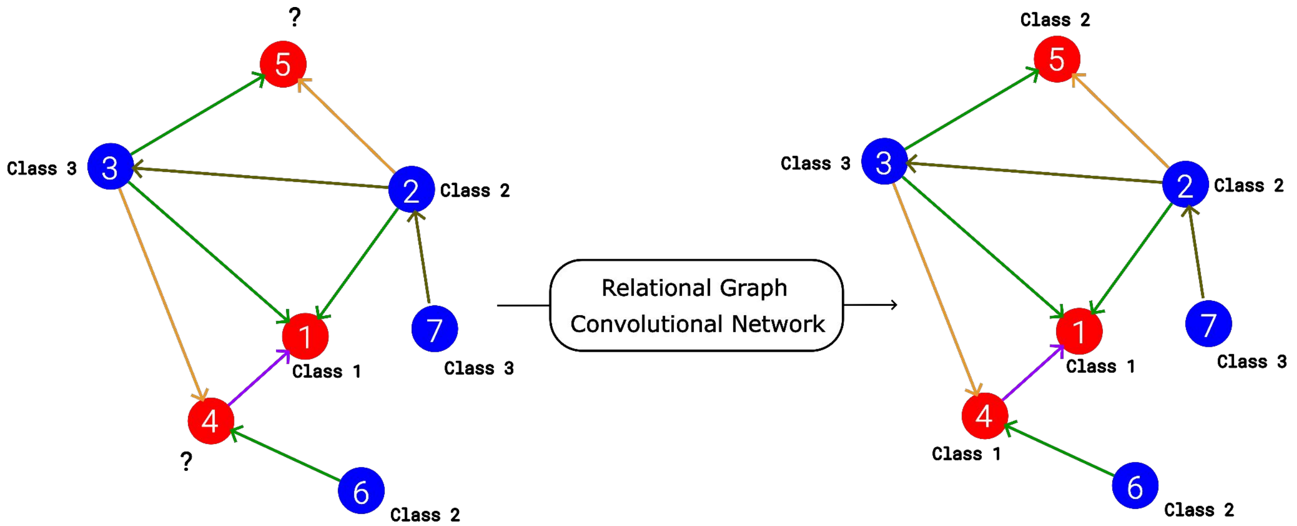
Key idea: share weights for different relations!

$$\underbrace{W_r^{(l)}}_{\text{Parameter matrix for relation r}} = \sum_{b=1}^{B} \overbrace{a_{r,b}^{(l)}}^{\text{Coefficients depending on r}} \underbrace{V_b^{(l)}}_{\text{Basis transformations}}$$

How? Sharing some parameters $V_b$ across all relations

# Relational (Heterogeneous) Graph Convolution
## Node classification

- Labels on some nodes (training nodes)
- Let’s predict labels of unlabeled nodes (test nodes)



# Code session
## Node classification on heterogeneous graphs

# Meta-paths
## New direction for heterogeneous graph learning

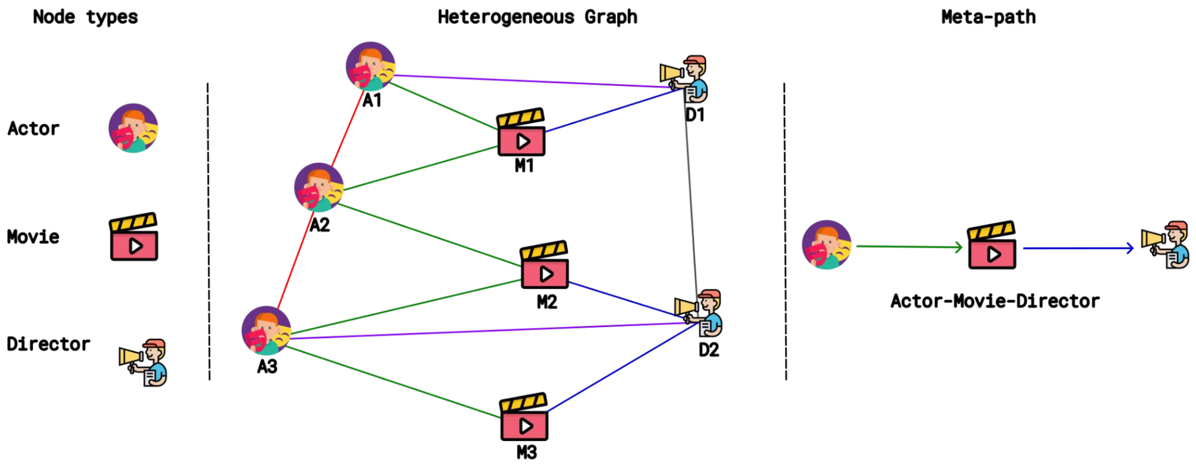
Previously: different parameters for different relation type

Problem: more edge types = higher number of parameters

**What's next?** Graph Neural Networks based on Meta-paths

# Meta-paths
## Introduction

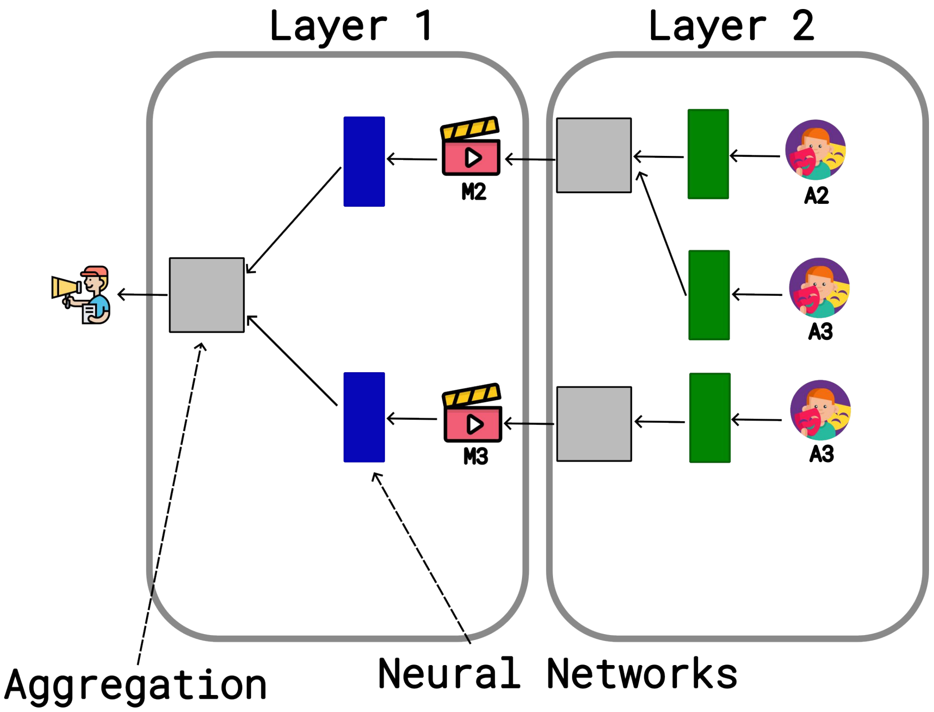
Meta-paths are chains of node and edge types in a heterogeneous graph



# Meta-paths
## Meta-paths and Graph Neural Networks

Why meta-paths are important?

Aggregation only through relations in the meta-path



# Meta-paths

## Meta-paths and Graph Neural Networks

### Graph Transformer Networks: Learning meta-path graphs to improve GNNs

Seongjun Yun, Minbyul Jeong, Sungdong Yoo, Seunghun Lee, Sean S. Yi, Raehyun Kim, Jaewoo Kang, Hyunwoo J. Kim*

*Department of Computer Science and Engineering, Korea University, Seoul, South Korea*

ARTICLE INFO

*Article history:*
Received 12 October 2021
Received in revised form 16 April 2022
Accepted 27 May 2022

ABSTRACT

Graph Neural Networks (GNNs) have been widely applied to various fields due to their powerful representations of graph-structured data. Despite the success of GNNs, most existing GNNs are designed to learn node representations on the *fixed* and *homogeneous* graphs. The limitations especially become problematic when learning representations on a misspecified graph or a *heterogeneous* graph

### GraphMSE: Efficient Meta-path Selection in Semantically Aligned Feature Space for Graph Neural Networks

**Yi Li[1], Yilun Jin[2], Guojie Song[3]*, Zihao Zhu[4], Chuan Shi[4], Yiming Wang[1]**

[1]Peking University, Beijing, China
[2]The Hong Kong University of Science and Technology, Hong Kong SAR, China
[3]Key Laboratory of Machine Perception (Ministry of Education), Peking University, Beijing, China
[4]Beijing University of Posts and Telecommunications, Beijing, China
{liyi2015, gjsong, wangyiming17}@pku.edu.cn, yilun.jin@connect.ust.hk, {zhuzihao,shichuan}@bupt.edu.cn

**Abstract**

Heterogeneous information networks (HINs) are ideal for describing real-world data with different types of entities and relationships. To carry out machine learning on HINs, meta-paths are widely utilized to extract semantics with pre-defined

Yet, the majority of existing works leveraging meta-path based GCNs assume that an indicative set of meta-paths have been given *a priori* upon which the GCNs are built, which is clearly not the case in reality. Specifically, real-world data are notoriously diverse, and it is not even re-

### MEGNN: Meta-path extracted graph neural network for heterogeneous graph representation learning

Yaomin Chang [a,b], Chuan Chen [a,b,*], Weibo Hu [a,b], Zibin Zheng [a,b], Xiaocong Zhou [a], Shouzhi Chen [c]

[a] *School of Computer Science and Engineering, Sun Yat-sen University, Guangzhou, China*
[b] *National Engineering Research Center of Digital Life, Sun Yat-sen University, Guangzhou, China*
[c] *Tencent Inc., Shenzhen, China*

ARTICLE INFO

*Article history:*
Received 3 December 2020
Received in revised form 14 October 2021
Accepted 15 October 2021
Available online 21 October 2021

*Keywords:*
Heterogeneous graph
Graph neural networks
Representation learning
Meta-paths

ABSTRACT

Heterogeneous graphs with multiple types of nodes and edges are ubiquitous in the real world and possess immense value in many graph-based downstream applications. However, the heterogeneity within nodes and edges in heterogeneous graphs has brought pressing challenges for practical node representation learning. Existing works manually define multiple meta-paths to model the semantic relationships in heterogeneous graphs. Such strategies heavily rely on the quality of domain knowledge and require extensive hand-crafted works. In this paper, we propose a novel Meta-path Extracted heterogeneous Graph Neural Network (MEGNN) that is capable of extracting meaningful meta-paths in heterogeneous graphs, providing insights about data and explainable conclusions to the model's effectiveness. Concretely, MEGNN leverages heterogeneous convolution to combine different bipartite sub-graphs corresponding to edge types into a new trainable graph structure. By adopting the message

### MAGNN: Metapath Aggregated Graph Neural Network for Heterogeneous Graph Embedding

Xinyu Fu
The Chinese University of Hong Kong
Hong Kong, China
xyfu@cse.cuhk.edu.hk

Jiani Zhang
The Chinese University of Hong Kong
Hong Kong, China
jnzhang@cse.cuhk.edu.hk

Ziqiao Meng
The Chinese University of Hong Kong
Hong Kong, China
zqmeng@cse.cuhk.edu.hk

Irwin King
The Chinese University of Hong Kong
Hong Kong, China
king@cse.cuhk.edu.hk

**ABSTRACT**

A large number of real-world graphs or networks are inherently heterogeneous, involving a diversity of node types and relation types. Heterogeneous graph embedding is to embed rich structural and semantic information of a heterogeneous graph into low-dimensional node representations. Existing models usually define multiple metapaths in a heterogeneous graph to capture the composite relations and guide neighbor selection. However, these models either omit node content features, discard intermediate nodes along the meta-

**1 INTRODUCTION**

Many real-world datasets are naturally represented in a graph data structure, where objects and the relationships among them are embodied by nodes and edges, respectively. Examples include social networks [14, 29], physical systems [2, 10], traffic networks [18, 34], citation networks [1, 14, 16], recommender systems [26, 35], knowledge graphs [3, 24], and so on. The unique non-Euclidean nature of graphs renders them difficult to be modeled by traditional machine learning models. For the neighborhood set of each node,

### Heterogeneous Graph Attention Network

Xiao Wang, Houye Ji
Beijing University of Posts and Telecommunications
Beijing, China
{xiaowang,jhy1993}@bupt.edu.cn

Chuan Shi*, Bai Wang
Beijing University of Posts and Telecommunications
Beijing, China
{shichuan,wangbai}@bupt.edu.cn

Peng Cui, P. Yu
Tsinghua University
Beijing, China
{cuip,psyu}@tsinghua.edu.cn

Yanfang Ye
West Virginia University
WV, USA
yanfang.ye@mail.wvu.edu

**ABSTRACT**

Graph neural network, as a powerful graph representation technique based on deep learning, has shown superior performance and attracted considerable research interest. However, it has not been fully considered in graph neural network for heterogeneous graph which contains different types of nodes and links. The heterogeneity and rich semantic information bring great challenges for designing a

web. Graph neural network (GNN), as a powerful deep representation learning method for such graph data, has shown superior performance on network analysis and aroused considerable research interest. For example, [10, 20, 24] leverage deep neural network to learn node representations based on node features and the graph structure. Some works [6, 14, 18] propose the graph convolutional networks by generalizing the convolutional operation to graph. A

# Meta-paths Learning for Multi-relational GNNs
## Introduction

Limitations of previous approaches:

- Predefined meta-paths
- Small number of relation types in the graph

Solution: learn meta-paths over graphs where the number of relations is high

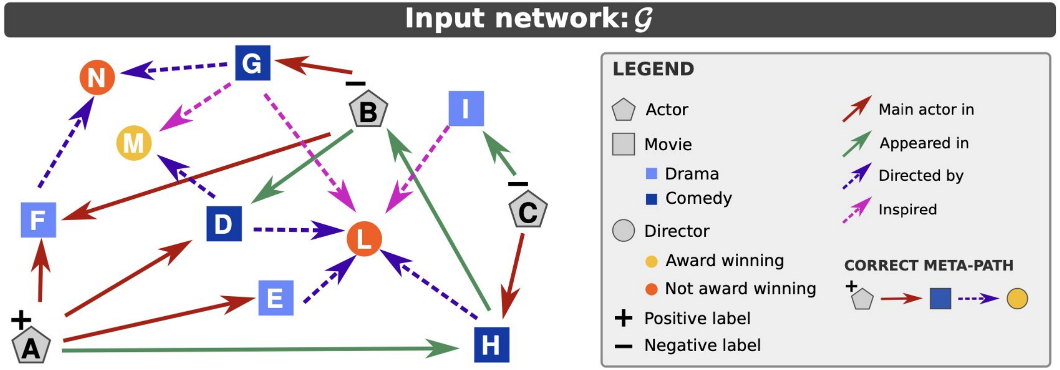
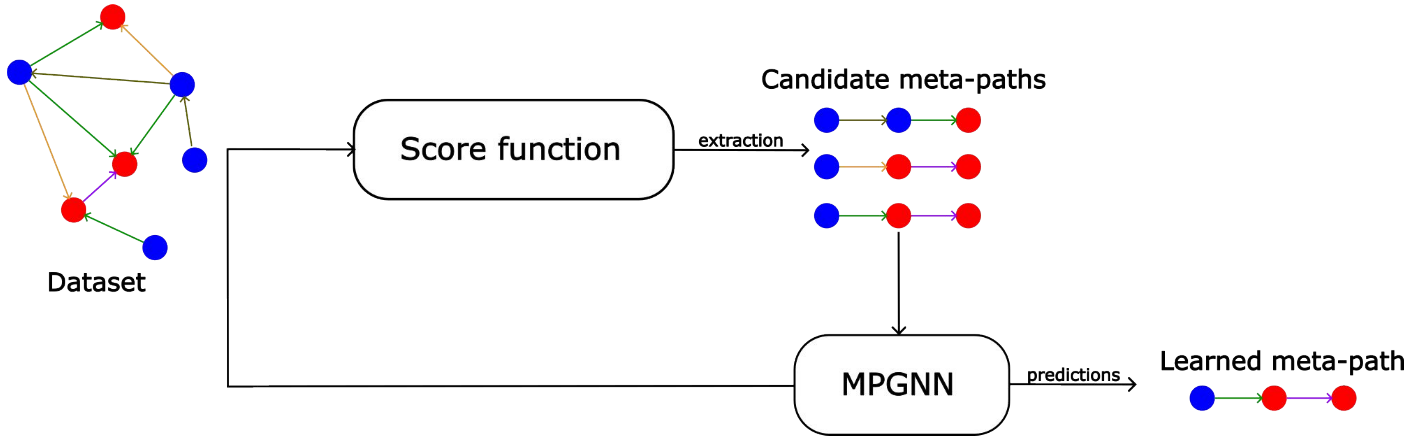



Image from Meta-path Learning for Multi-relational Graph Neural Networks, F. Ferrini et al., LOG 2023 [paper]

# Meta-paths Learning for Multi-relational GNNs
## Method

Method:

- A score function extracts candidate meta-paths
- Meta-path Graph Neural Network make predictions only with learnt meta-paths



# Meta-paths Learning for Multi-relational GNNs
## Results

Results:

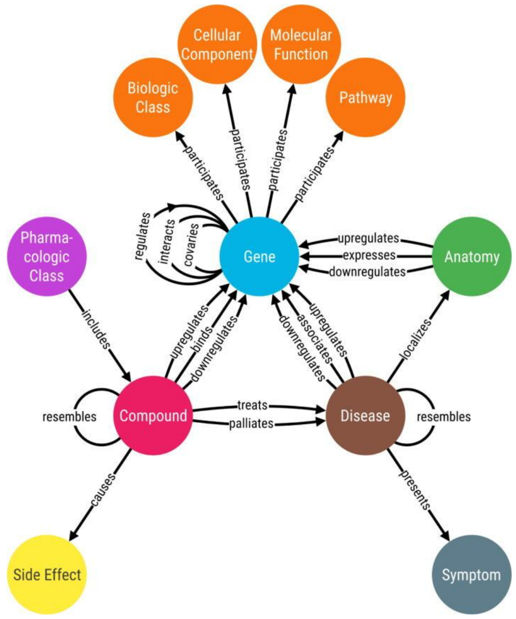
- Better performances on synthetic datasets with increasing complexity
- Similar performances with SOTA approaches in real datasets with few relations
- Better performances in real world dataset with high number of relations (Freebase)

# Extra session
## Knowledge Graphs

# Knowledge Graphs
## Introduction

Knowledge Graphs represents real-world knowledge through structured entities and relationships.

- Entities: real-world concepts (i.e. people, objects)
- Relations: connect entities
- Facts: triplets (head entity, relation, tail entity)



# Knowledge Graphs
## Introduction

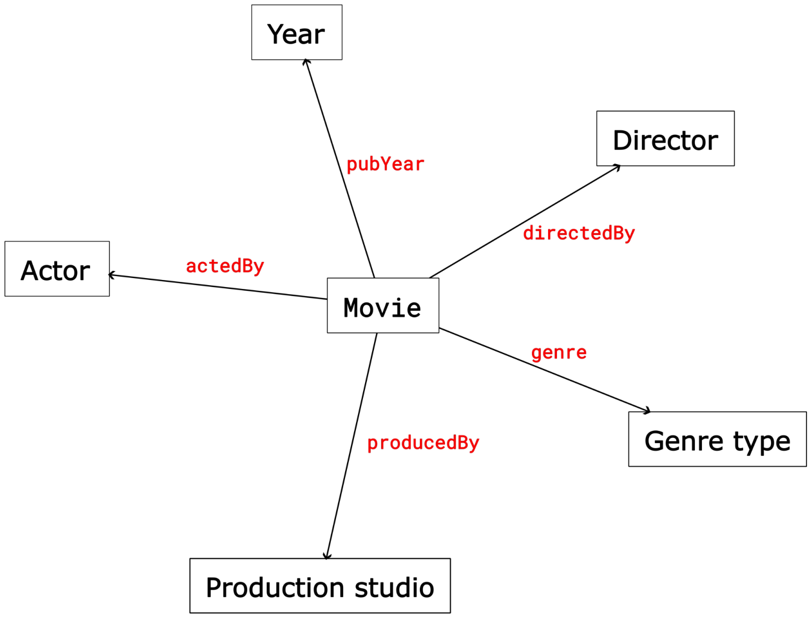
- Wikidata: [source]
- DBPedia: [source]
- Yago: [source]
- WordNet: [source]
- Microsoft Academic Graph (MAG): [source]
- Google Knowledge Graph: [source]
- Freebase: incorporated in Google Knowledge Graph

# Knowledge Graphs
## Example: Freebase

Node types: Movie, Director, Year, Production Studio, Actor, Genre type

Relation types: directedBy, pubYear, productedBy actedBy, genre

Triplet example: (Inception, pubYear, 2010)



# Knowledge Graphs
## Tasks

Fact: Real Knowledge Graphs have millions of entities and thousand of relations

| | DBpedia | Freebase | OpenCyc | Wikidata | YAGO |
|---|---|---|---|---|---|
| Number of triples | 411 885 960 | 3 124 791 156 | 2 412 520 | 748 530 833 | 1 001 461 792 |
| Number of classes | 736 | 53 092 | 116 822 | 302 280 | 569 751 |
| Number of relations | 2819 | 70 902 | 18 028 | 1874 | 106 |
| No. of unique predicates | 60 231 | 784 977 | 165 | 4839 | 88 736 |
| Number of entities | 4 298 433 | 49 947 799 | 41 029 | 18 697 897 | 5 130 031 |

Table from Which Knowledge Graph is Best for me? M. Färber et al. 2018

# Knowledge Graphs
## Opportunities

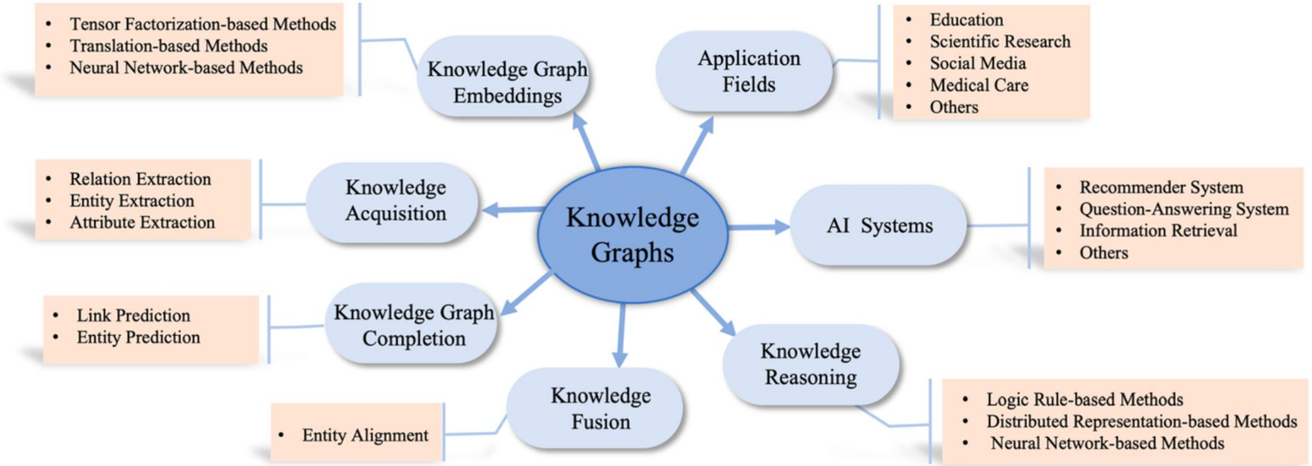



Figure from Knowledge Graph: Opportunities and Challenges C. Peng et al. 2023

# Knowledge Graphs
## Resources

Surveys:

- Knowledge Graph: Opportunities and Challenges [source]
- Knowledge Graph Embedding: A Survey from the Perspective of Representation Spaces [source]
- A Survey on Knowledge Graph Embeddings for Link Prediction [source]
- Unifying Large Language Models and Knowledge Graphs: A Roadmap [source]

# Conclusions

- Different ways of dealing with heterogeneity in graphs
- Some approaches struggle because of the high number of parameters
- Meta-path Graph Neural Networks focus on relevant paths in the heterogeneous graph
- Challenges in Knowledge Graphs (Code for Knowledge Graph Embedding available soon at github.com/steveazzolin/gdl_tutorial_turinginst)

# Introduction
## Organization and material

Tutorial in four parts (slides + Jupyter notebooks available at link):

- ~~**Part I:** November 2, Presenter: **GS**~~
  Goals: Motivations, Intro of basic concepts, definition of GNNs
- ~~**Part II:** November 9, Presenter: **AL**~~
  Goals: Implementation of GNNs: How to implement a full GNN pipeline in PyTorch Geometric.
- ~~**Part III:** November 16, Presenter: **SA**~~
  Goals: Explainability of GNNs: How to inspect a model to try to understand the learned decision pattern.
- ~~**Part IV:** November 23, Presenter: **FF**~~
  Goals: Heterogeneity in GNNs: How can GNNs effectively model and incorporate a diversity of nodes and edges with different types.