

Graph Neural Networks: Benchmarks and Future Directions

Vijay Dwivedi*, **Chaitanya K. Joshi***,
Thomas Laurent, Yoshua Bengio, Xavier Bresson

ArXiv paper, March 2020: arxiv.org/abs/2003.00982

Software toolkit on GitHub: github.com/graphdeeplearning/benchmarking-gnns

About Me

- **Research Engineer** at I2R since August.
- Graduated from **NTU Comp Science** in 2019, previously RA with Prof. Xavier Bresson.
- Interests: **Natural Language Processing**, Dialog Systems, **Graph Neural Networks**, Combinatorial Optimization.
- Profile: <http://chaitjo.github.io/>

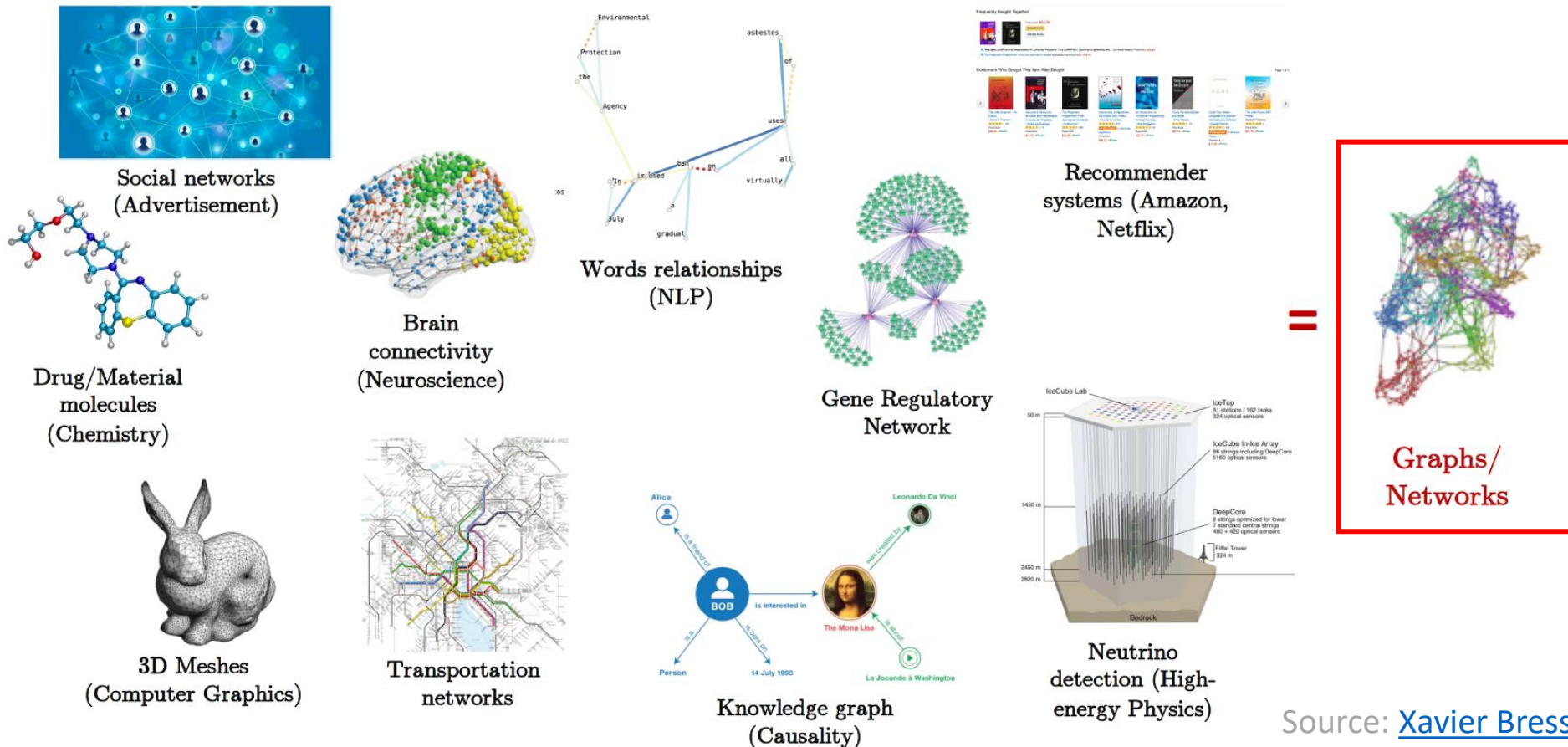


Outline of this Talk

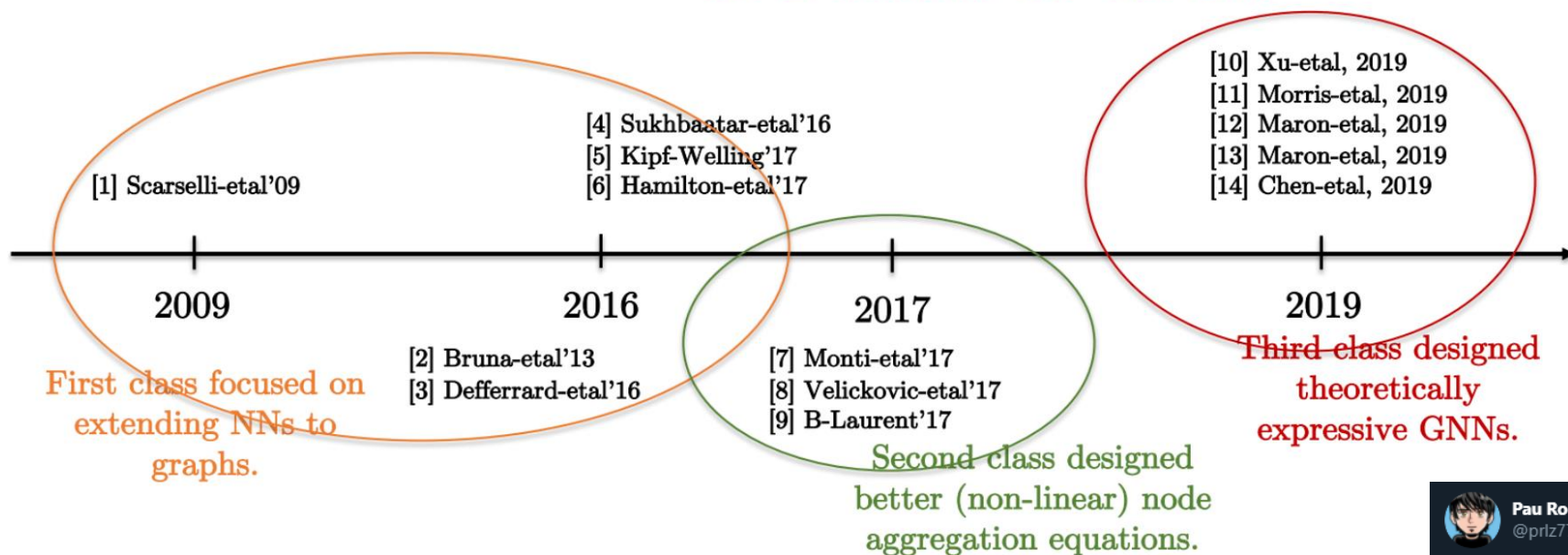
- 1. What are Graph Neural Networks?**
2. Why do we need new benchmarks?
3. Our proposed benchmark
4. Our insights from benchmarking + future directions

Graph Neural Networks (GNNs)

GNNs → deep learning on **graph-structured data** beyond images and text:



A Decade of GNNs

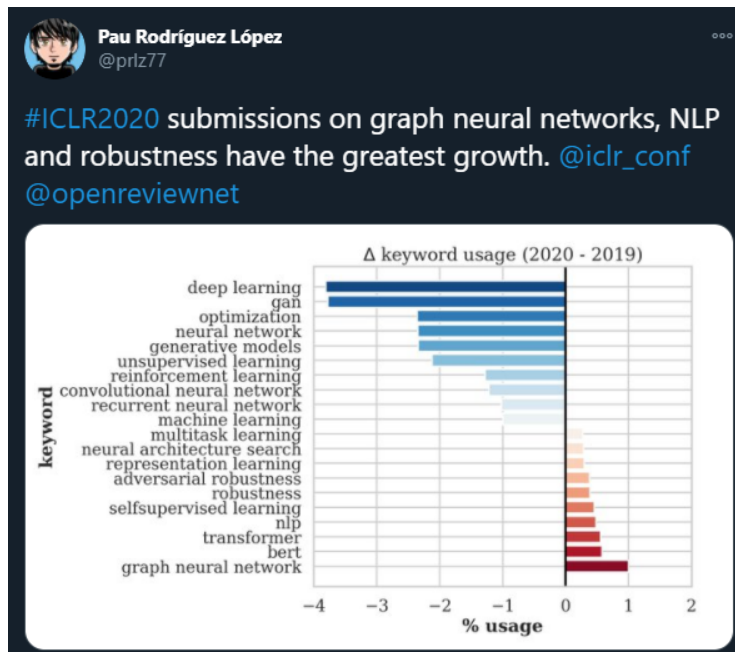


Developing powerful GNNs for real-world adoption of graph deep learning.

- [1] Scarselli, Gori, Tsoi, Hagenbuchner, Monfardini, The Graph Neural Network Model, 2009
- [2] Bruna, Zaremba, Szlam, LeCun, Spectral networks and locally connected networks on graphs, 2013
- [3] Defferrard, Bresson, Vandergheynst, Convolutional neural networks on graphs with fast localized spectral filtering, 2016
- [4] Sukhbaatar, Szlam, Fergus, Learning multiagent communication with backpropagation, 2016
- [5] Kipf, Welling, Semi-supervised classification with graph convolutional networks, 2017
- [6] Hamilton, Ying, Leskovec, Inductive representation learning on large graphs, 2017
- [7] Monti, Boscaini, Masci, Rodola, Svoboda, Bronstein, Geometric deep learning on graphs using mixture model cnns, 2017
- [8] Velickovic, Cucurull, Casanova, Romero, Lio, Bengio, Graph attention networks, 2017
- [9] Bresson, Laurent, Residual gated graph convnets, 2017
- [10] Xu, Hu, Leskovec, Jegelka, How powerful are graph neural networks?, 2019
- [11] Morris, Ritzert, Fey, Hamilton, Lenssen, Rattan, Grohe, Weisfeiler and leman go neural: Higher-order graph networks, 2019
- [12] Maron, Ben-Hamu, Shamir, Lipman, Invariant and equivariant graph networks, 2019
- [13] Maron, Ben-Hamu, Serviansky, Lipman, Provably powerful graph networks, 2019
- [14] Chen, Villar, Chen, Bruna, On the equivalence graph isomorphism testing and function approximation with gnns, 2019

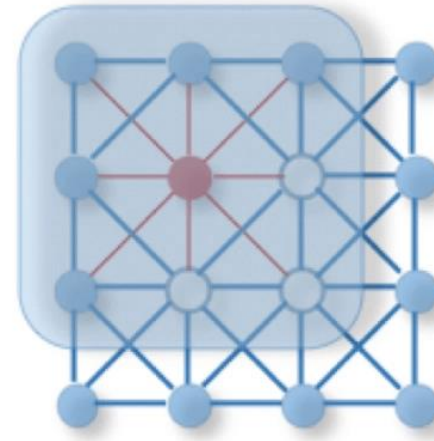
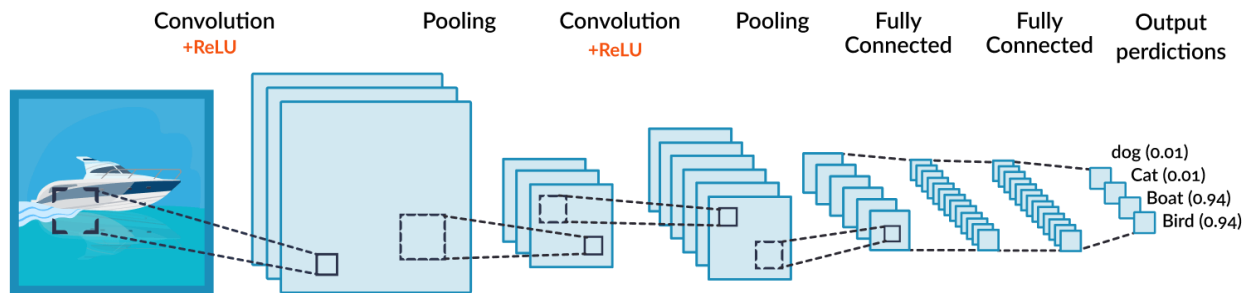
Apologize for not citing more works (4000+ GNN papers).

Xavier Bresson



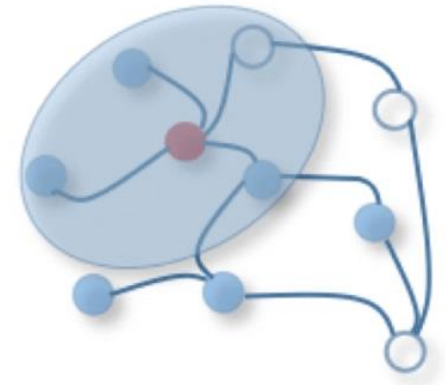
Graph Convolutions a.k.a. Message Passing

- Images → **2D Grid graphs** where each pixel is connected to 8 neighbors.
- CNN → **Sliding filter** over pixel grid.
- GNN → Sliding filter over **any arbitrary graph structure**.



Pixel grids: Fixed # neighbors, and neighbors are ordered (up/down/left/right).

=> **2D Convolution:** weighted average of pixel values.



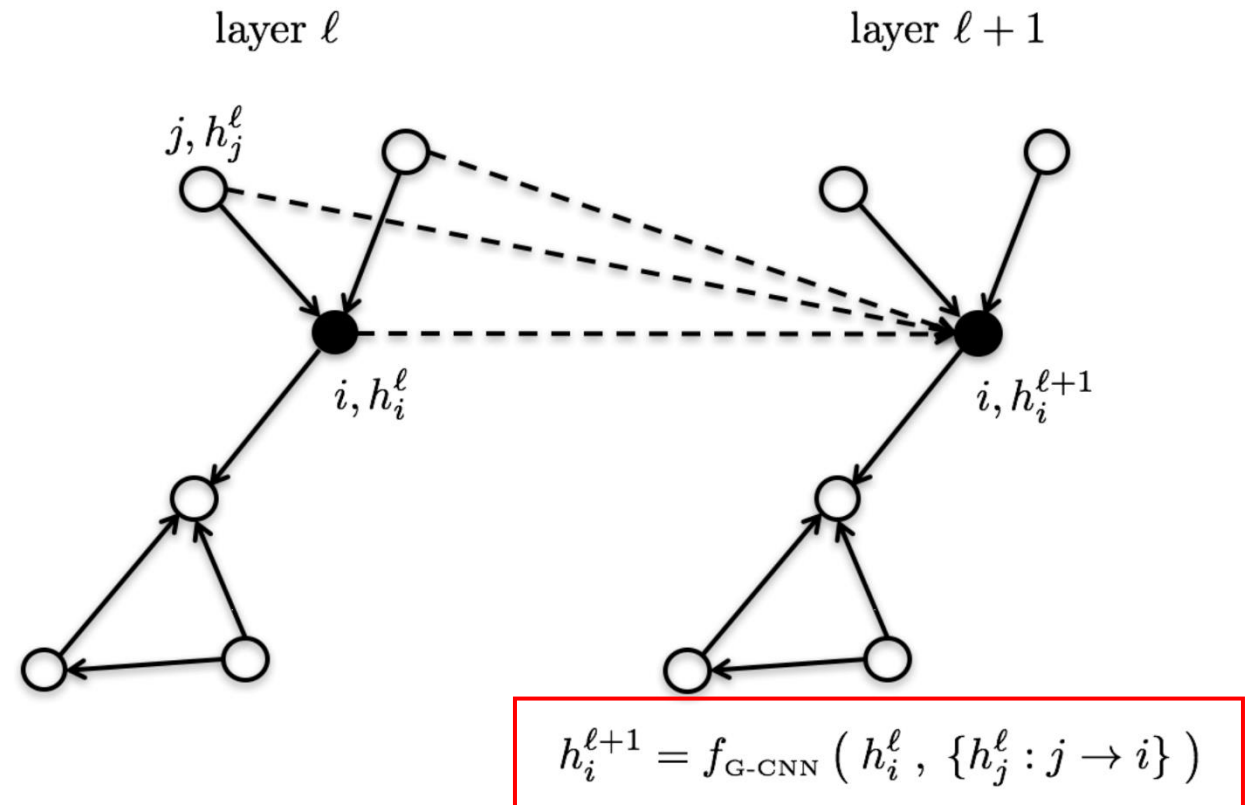
Graphs: Arbitrary # neighbors, and no canonical ordering of neighbors.

=> **Graph Convolution:** simple average of node features.

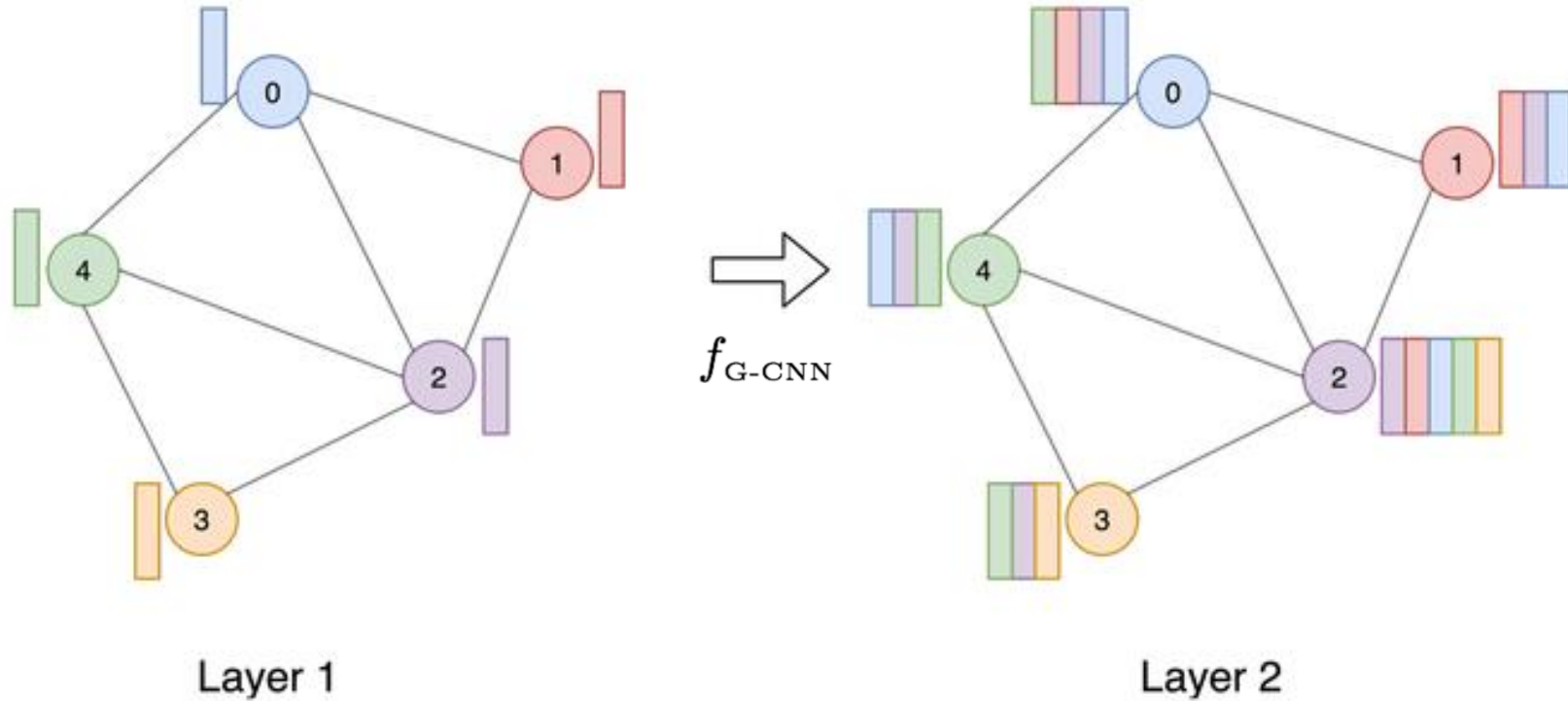
Graph Convolutions a.k.a. Message Passing

GNN filter must have the following properties:

1. **Locality**, i.e. only nodes' neighbors are convolved
2. Independent of **graph size**, i.e. weight sharing across all nodes
3. Independent of **node ordering**
4. Independent of **number of neighbors**



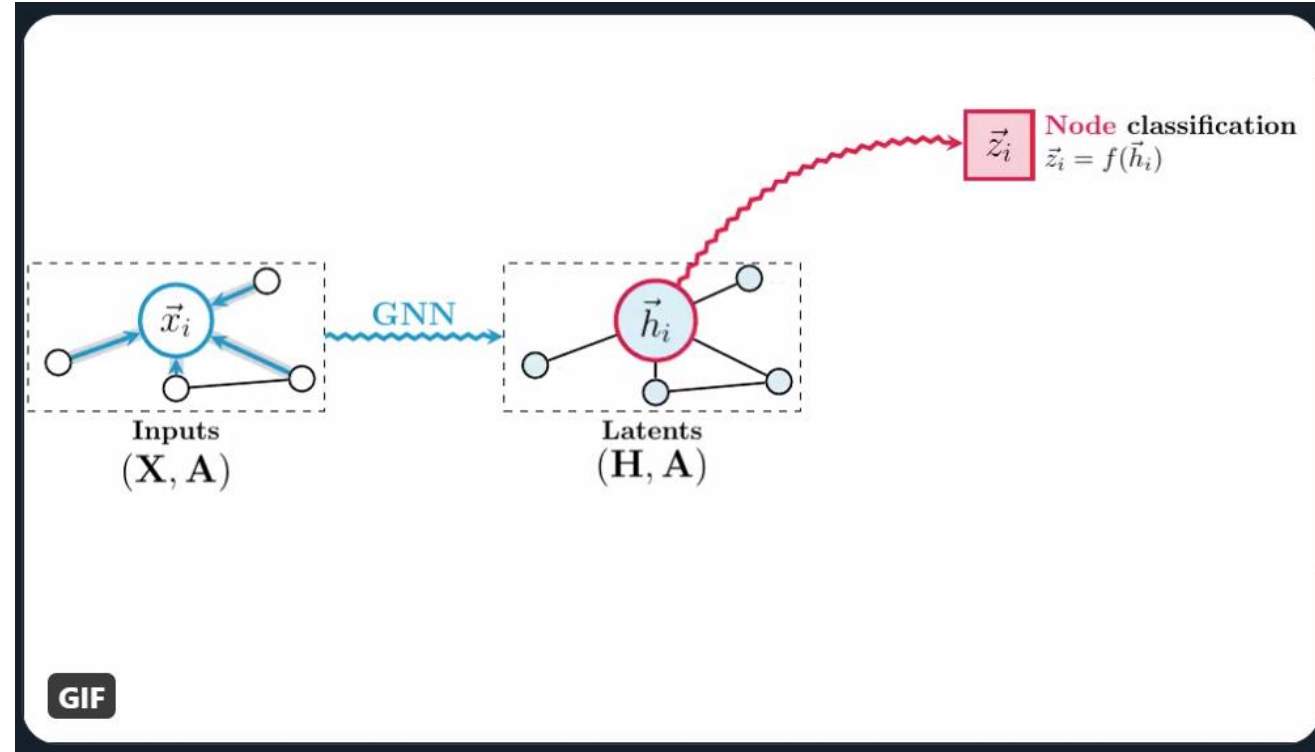
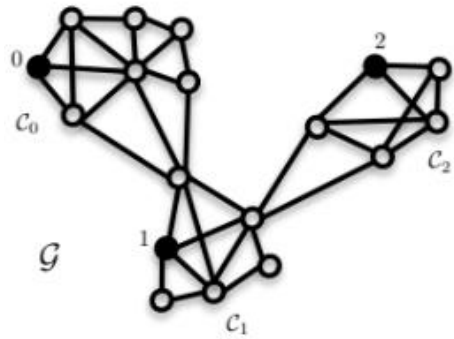
Graph Convolutions a.k.a. Message Passing



$$h_i^{\ell+1} = f_{G-CNN} (h_i^{\ell} , \{ h_j^{\ell} : j \rightarrow i \})$$

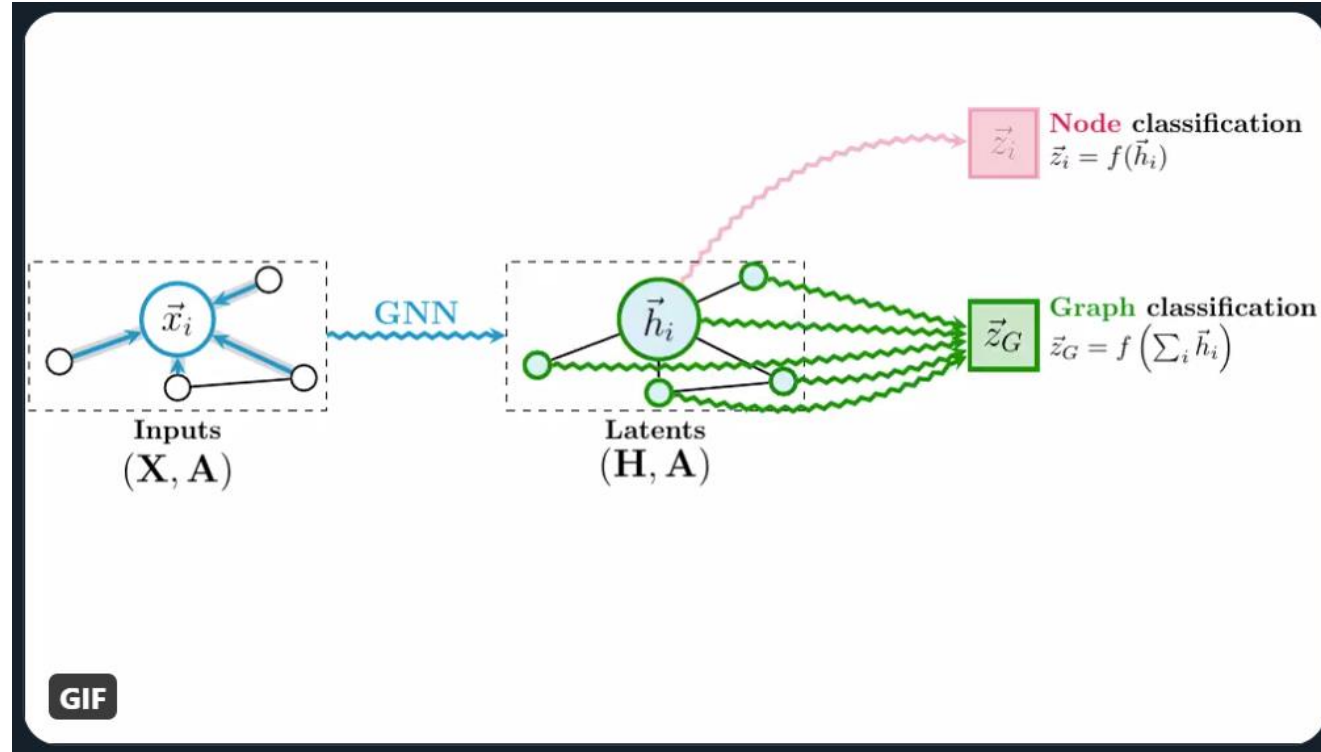
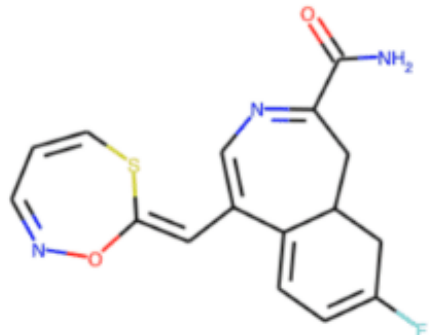
Typical Graph ML Tasks: Node-level

e.g. Community detection on social networks, Point cloud part segmentation



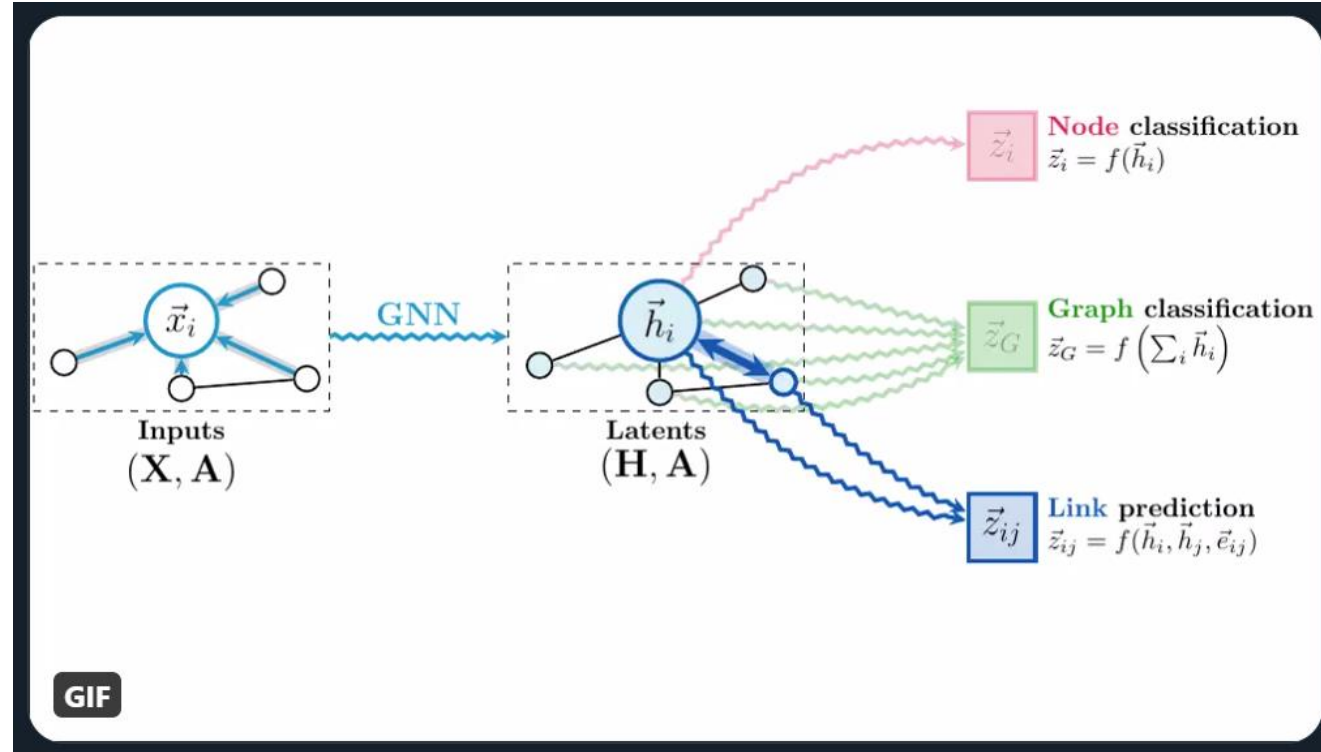
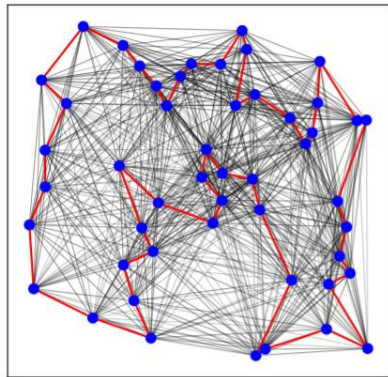
Typical Graph ML Tasks: Graph-level

e.g. Molecular property prediction, Point cloud classification



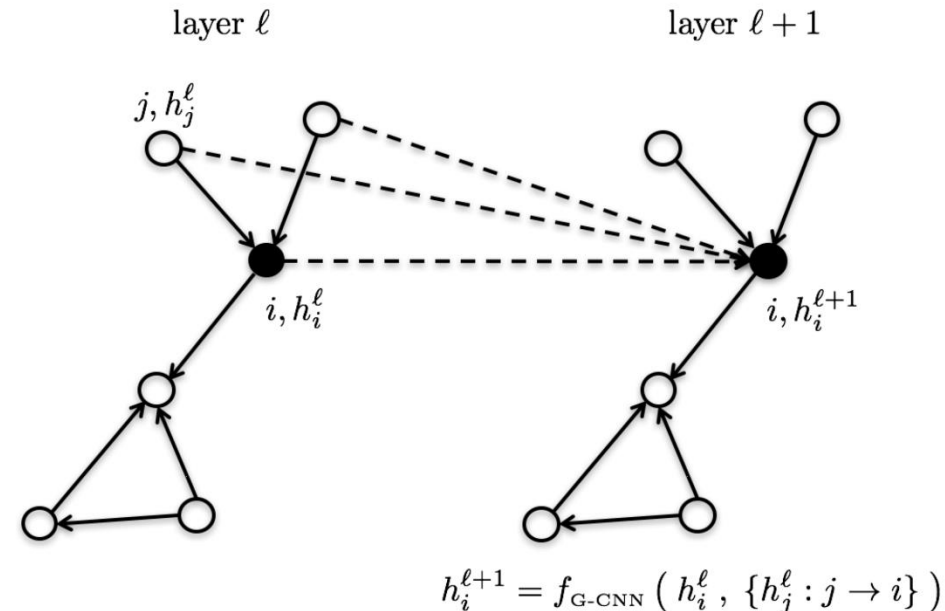
Typical Graph ML Tasks: Edge-level

e.g. Protein-protein interaction prediction



Simple, Isotropic MP-GNNs

$$h_i^{\ell+1} = \text{ReLU} \left(U^\ell h_i^\ell + \sum_{j \in \mathcal{N}_i} V^\ell h_j^\ell \right),$$



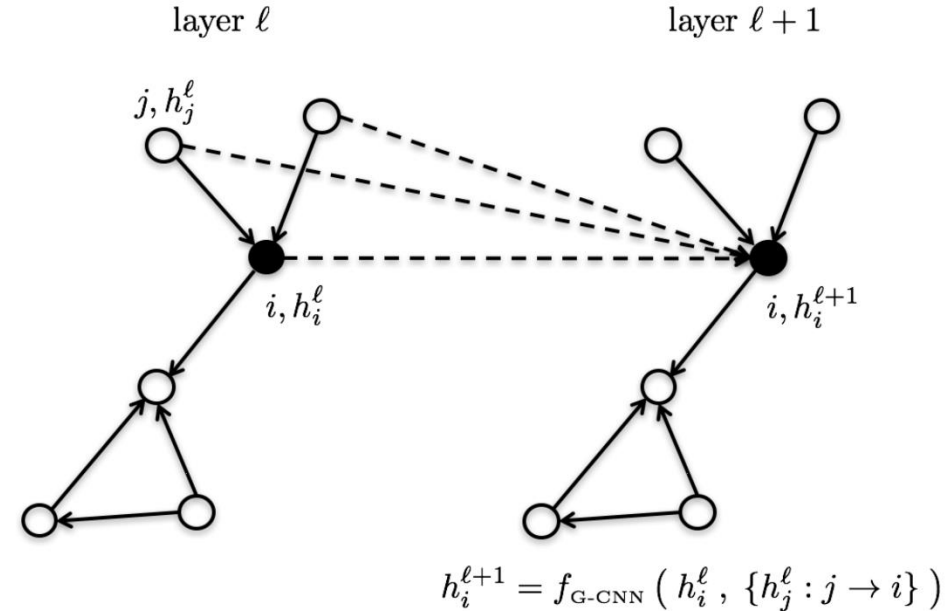
- **Isotropic** neighborhood aggregation: considers each neighbor **equally important** when **aggregating** to update the features at each node.
- Time/Space complexity: **$\mathbf{O}(n)$** , where $n \rightarrow$ number of graph nodes

Simple, Isotropic MP-GNNs

$$h_i^{\ell+1} = \text{ReLU} \left(U^\ell h_i^\ell + \sum_{j \in \mathcal{N}_i} V^\ell h_j^\ell \right),$$

Trainable parameters

Aggregation Function:
Sum/Mean/Max



- **Isotropic** neighborhood aggregation: considers each neighbor **equally important** when **aggregating** to update the features at each node.
- Time/Space complexity: **$\mathcal{O}(n)$** , where $n \rightarrow$ number of graph nodes

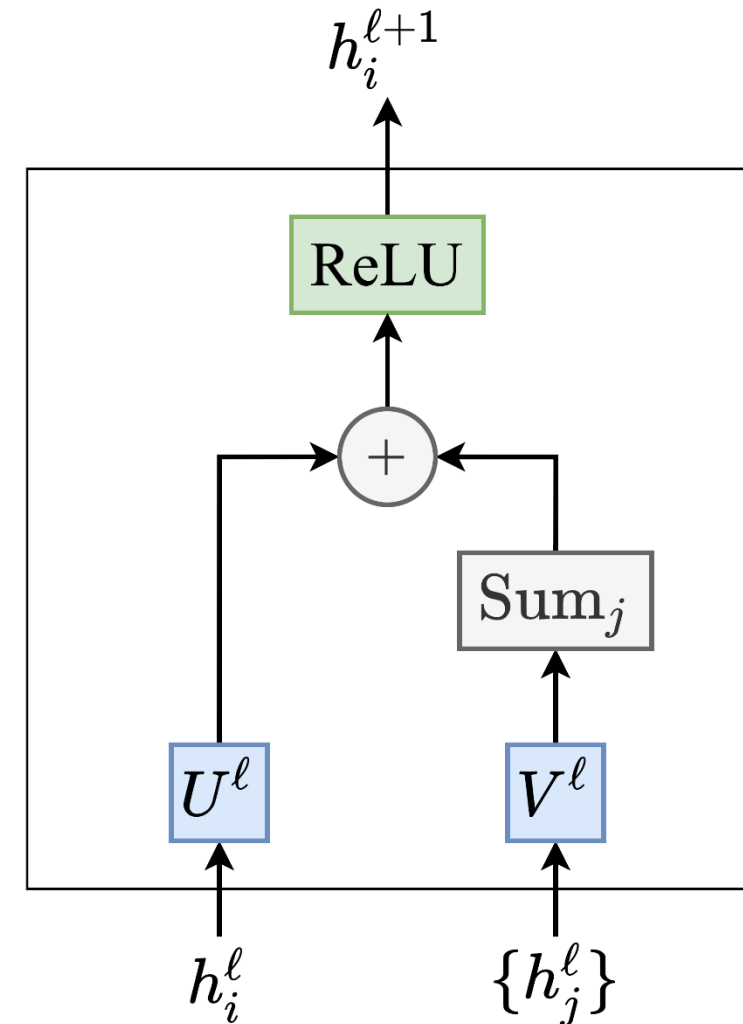
Simple, Isotropic MP-GNNs

$$h_i^{\ell+1} = \text{ReLU}\left(U^\ell h_i^\ell + \sum_{j \in \mathcal{N}_i} V^\ell h_j^\ell\right),$$

Parameter for
central node

Parameter for
each neighbor

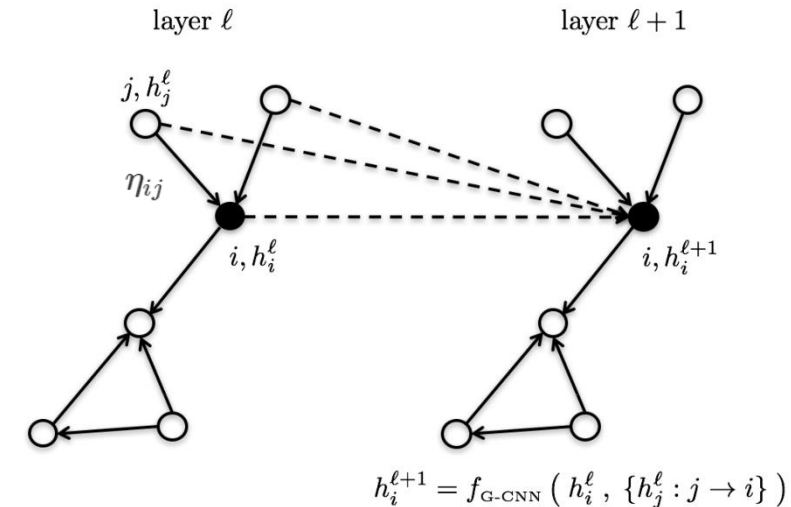
- **Original GCNs:** [Sukhbaatar-etal, 2016](#); [Kipf-Welling, 2017](#)
- **GraphSage,** [Hamilton-etal, 2017](#)
- **ChebNet,** [Defferrard-etal, 2016](#)



Anisotropic MP-GNNs

$$h_i^{\ell+1} = \text{ReLU}\left(U^\ell h_i^\ell + \sum_{j \in \mathcal{N}_i} \eta_{ij} \odot V^\ell h_j^\ell\right),$$

$$\text{where } \eta_{ij} = \sigma(A^\ell h_i^\ell + B^\ell h_j^\ell),$$



- **Graphs have no direction**, unlike images. However, some neighbors may be more important than others for given task...
- **Anisotropic mechanisms** allows GNNs to learn **weighted aggregation**.

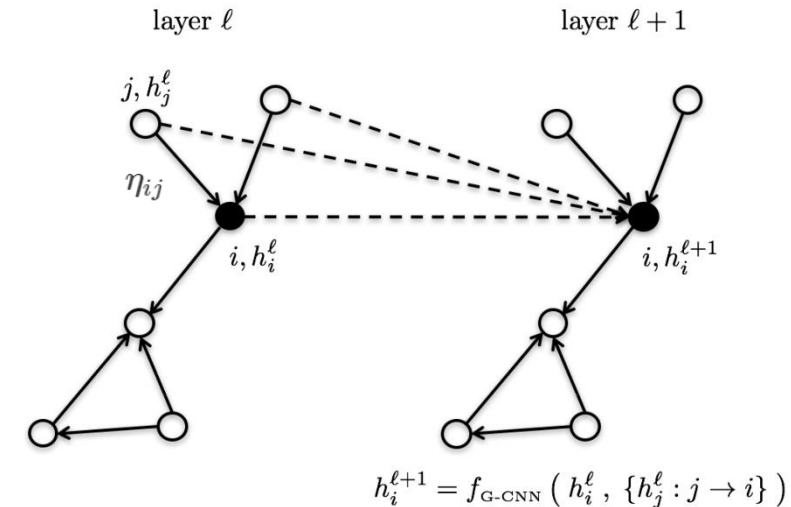
Anisotropic MP-GNNs

$$h_i^{\ell+1} = \text{ReLU}\left(U^\ell h_i^\ell + \sum_{j \in \mathcal{N}_i} \eta_{ij} \odot V^\ell h_j^\ell\right),$$

where $\eta_{ij} = \sigma(A^\ell h_i^\ell + B^\ell h_j^\ell)$,

Can also take softmax over all $j \in \text{Neighbors}(i)$

Weight for edge i - j , multiplied to feature of node j



Approach: learn **features for each edge** in addition to node features:

- Initialize as **input edge features**, if available, e.g. molecules and bond types.
- Learn edge features as **joint representations of node features** at each layer.
- Use Gating or **Attention mechanisms** to learn weighted aggregation.

Anisotropic MP-GNNs

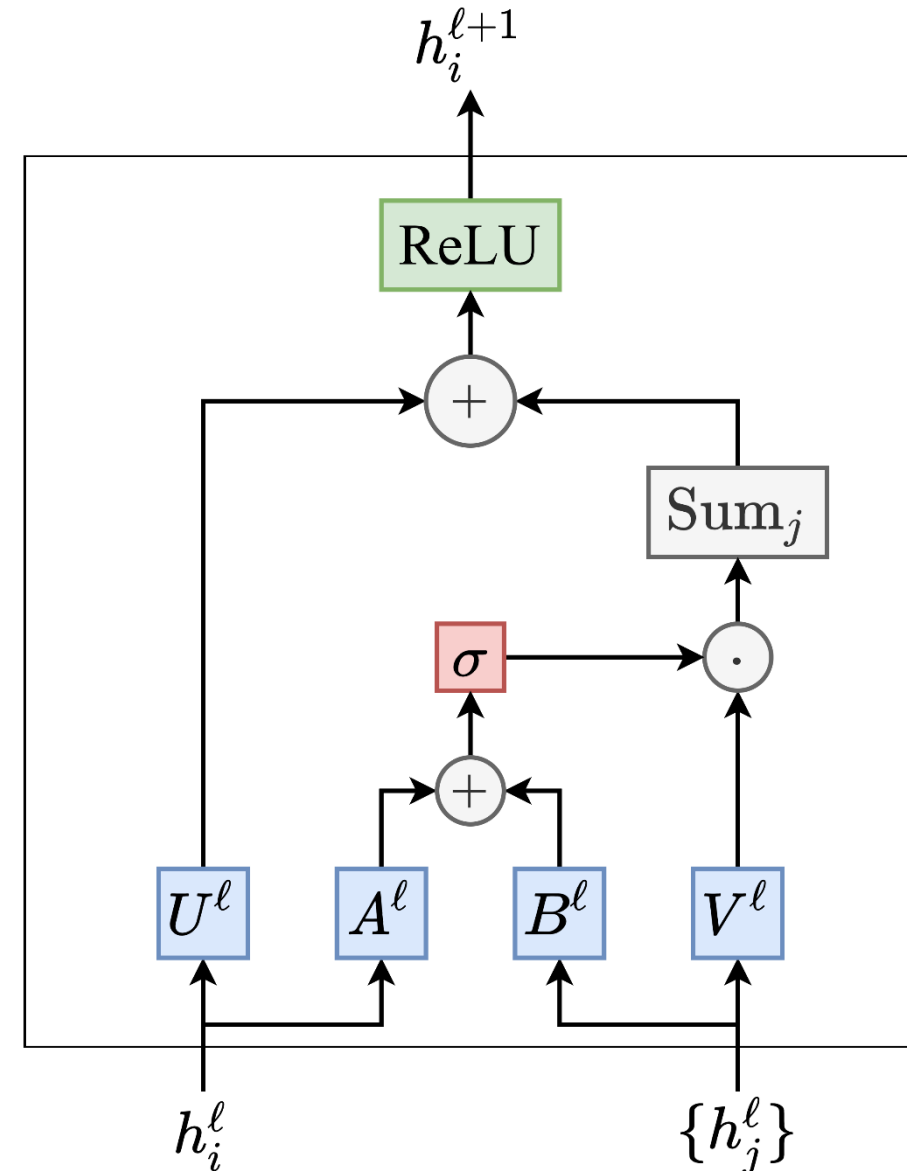
$$h_i^{\ell+1} = \text{ReLU}\left(U^\ell h_i^\ell + \sum_{j \in \mathcal{N}_i} \eta_{ij} \odot V^\ell h_j^\ell\right),$$

where $\eta_{ij} = \sigma(A^\ell h_i^\ell + B^\ell h_j^\ell)$,

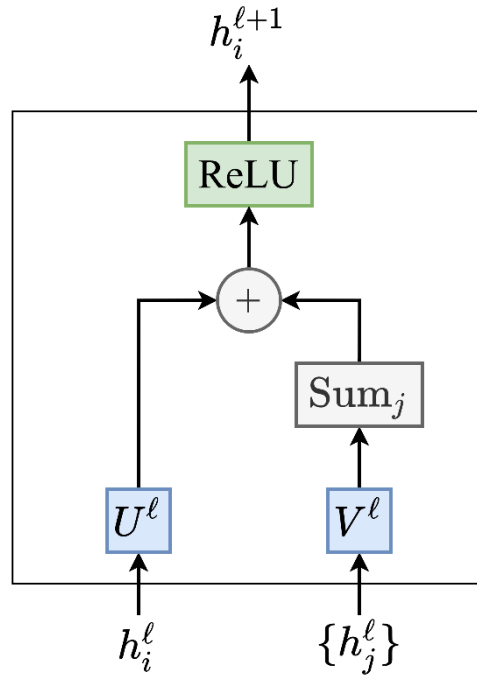
Can also take softmax over all $j \in \text{Neighbors}(i)$

Weight for edge i - j , multiplied to feature of node j

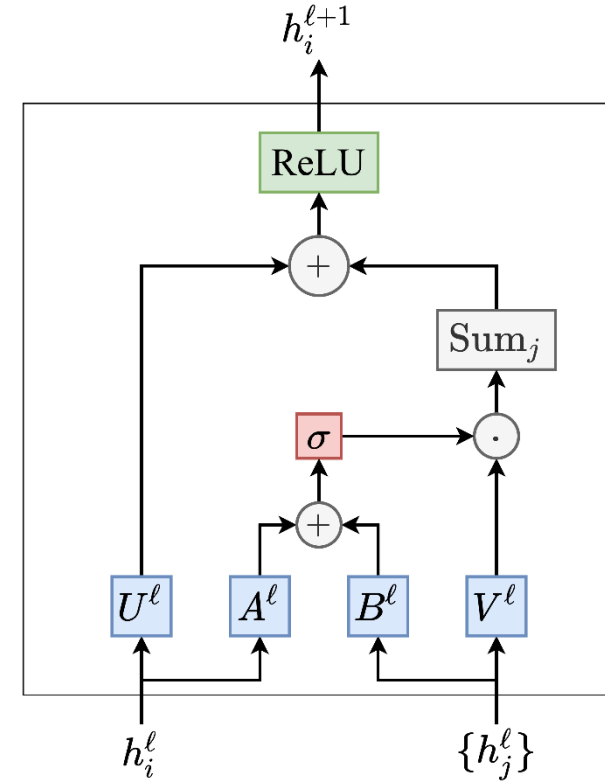
- **Graph Attention Network**, [Velickovic-etal, 2018](#)
- **Gated GCNs**: [Marchegiani-Titov, 2017](#); [Bresson-Laurent, 2018](#)
- **MoNet**, [Monti-etal, 2016](#)



Simple GCN vs. Gated GCN



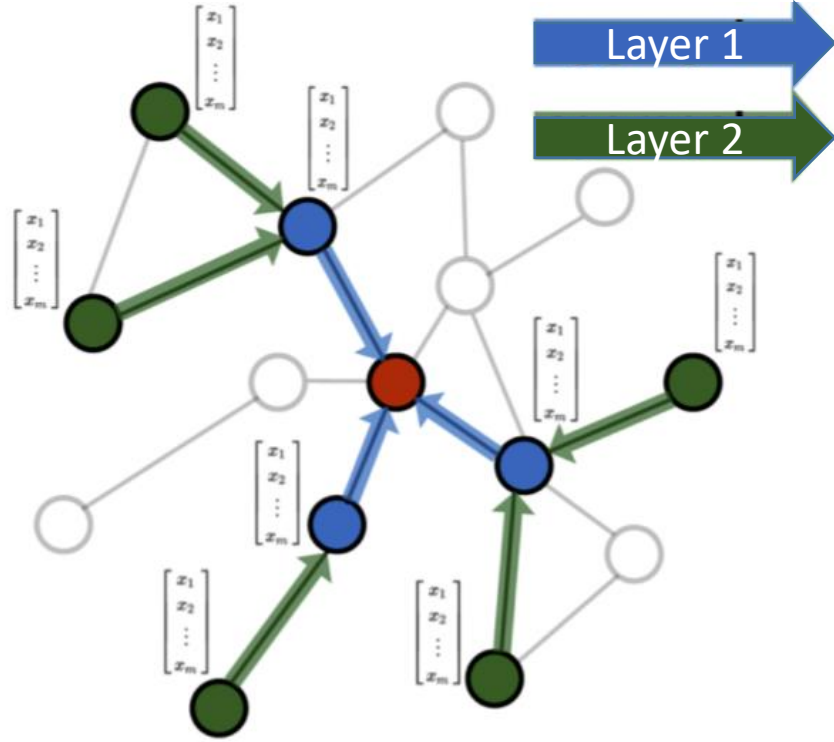
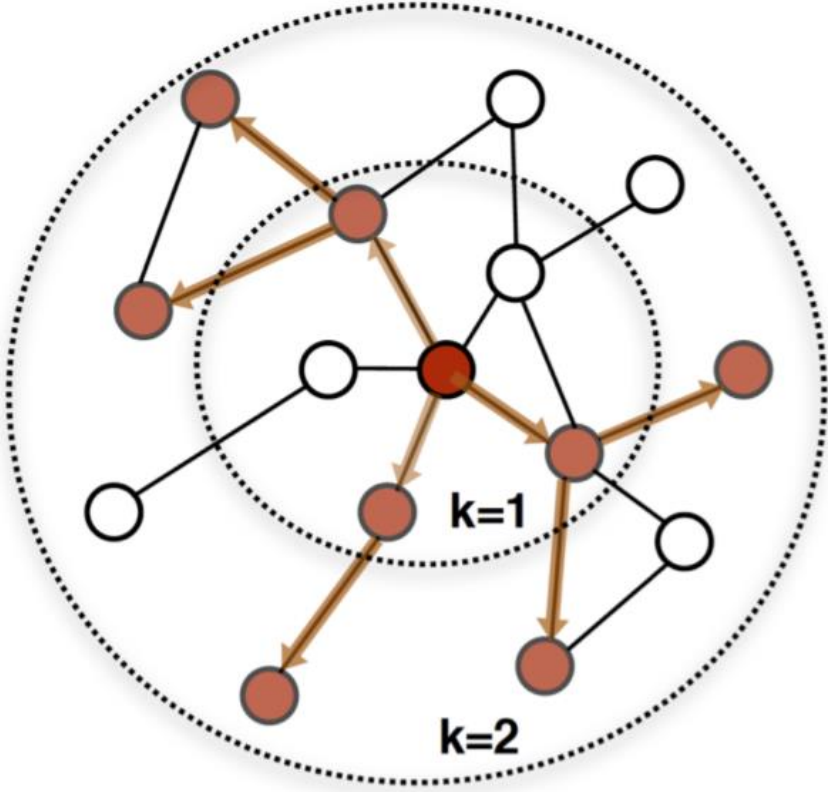
vs.



More complex architectures =>

Tradeoff between **computational efficiency** vs. **performance**

Message Passing and k-Hop Neighborhoods



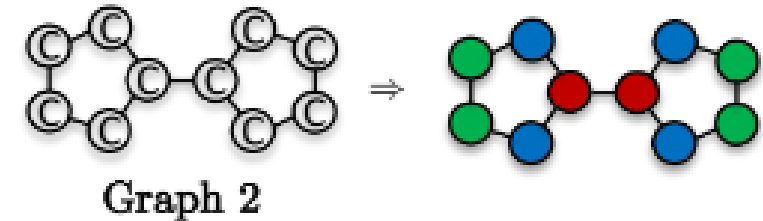
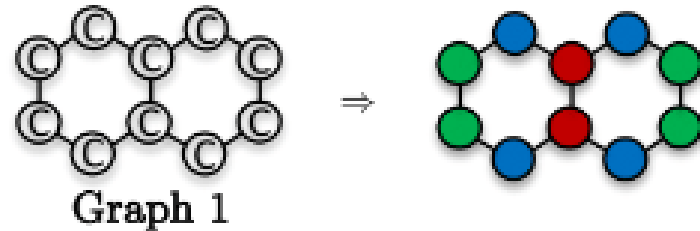
At layer k : Message (i.e. features) from each node are sent to its k -hop neighbors.



Each node receives information from its k -hop neighborhood

Theoretical Limits of MP-GNNs: Graph Isomorphism

Node features after message-passing



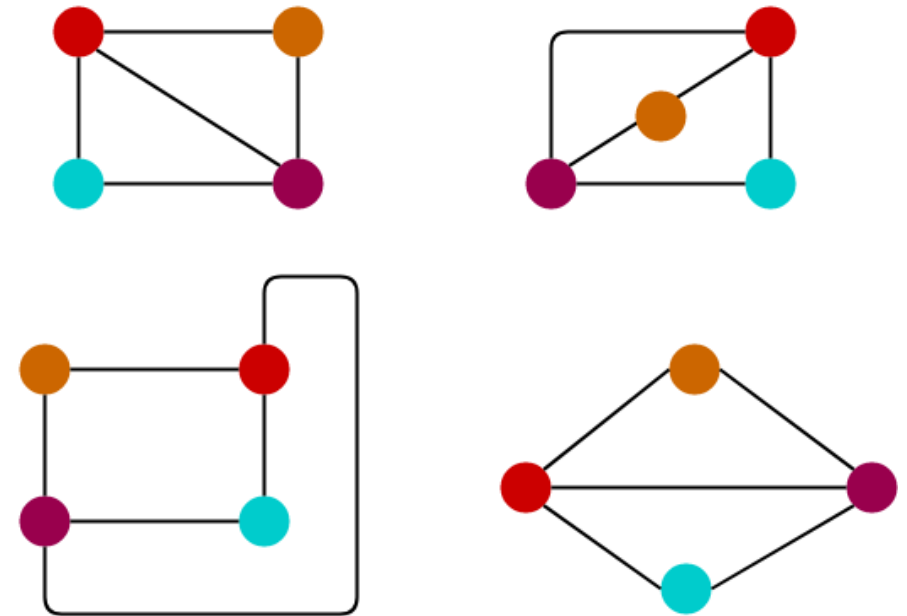
Final graph features
= sum all node features



MP-GNNs will fail to distinguish G1 and G2!

Theoretical Limits of MP-GNNs: Graph Isomorphism

- Two graphs are **isomorphic** \rightarrow there exists an index permutation between the nodes that **preserves node adjacencies**.
- Graph theory x GNNs \rightarrow characterize **GNNs' expressive power** via graph isomorphism testing

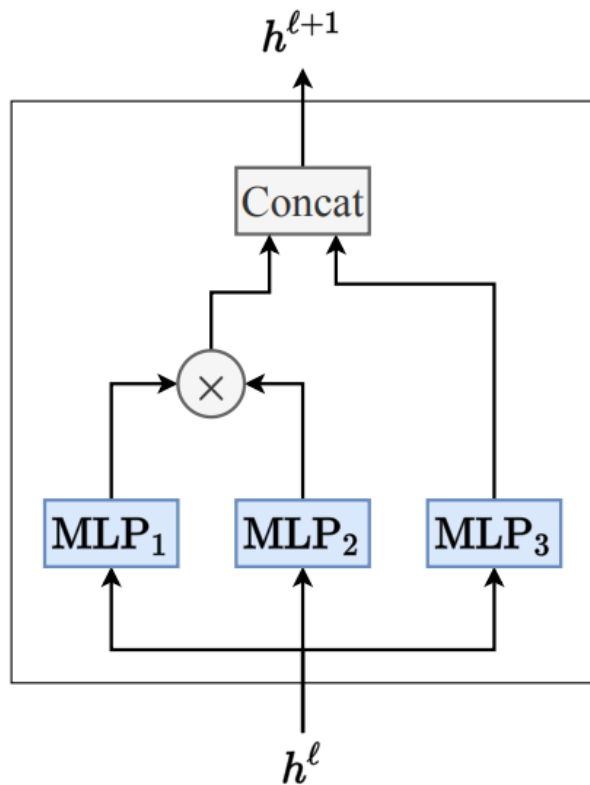


All 4 graphs are isomorphic.

Further reading: <http://irregulardeep.org/>

Theoretically Expressive WL-GNNs

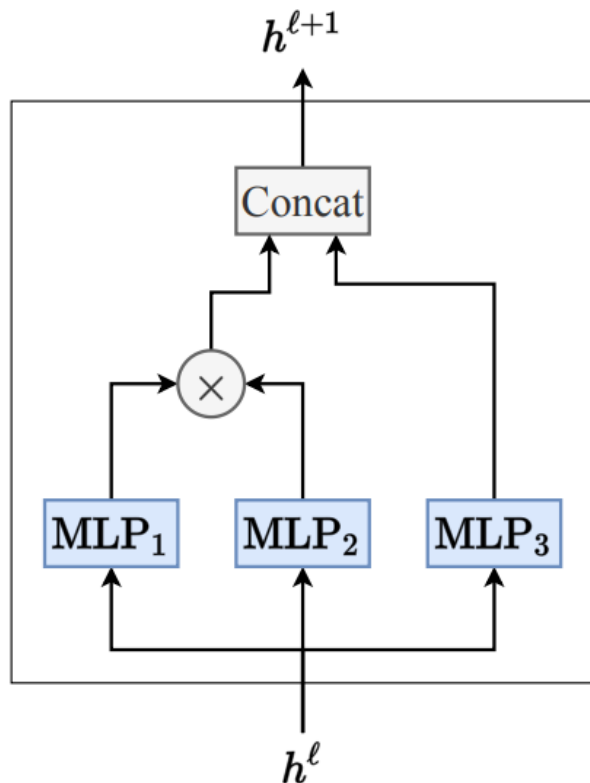
New models designed based on graph isomorphism testing (**Weisfeiler-Lehman Test**): Higher Order GNN, Moris-etal, 2019; **Equivariant GNN**, Maron-etal, 2019.



$$h^{\ell+1} = \text{Concat}\left(M_{W_1^\ell}(h^\ell) \cdot M_{W_2^\ell}(h^\ell), M_{W_3^\ell}(h^\ell)\right),$$

Theoretically Expressive WL-GNNs

New models designed based on graph isomorphism testing (**Weisfeiler-Lehman Test**): Higher Order GNN, Moris-etal, 2019; **Equivariant GNN**, Maron-etal, 2019.



$$h^{\ell+1} = \text{Concat} \left(M_{W_1^\ell}(h^\ell) \cdot M_{W_2^\ell}(h^\ell), M_{W_3^\ell}(h^\ell) \right),$$

Dense ' $n \times n \times d$ ' tensor
 \Rightarrow global graph update

Product of two 3D Tensors
 $\Rightarrow O(n^3)$ complexity!

WL-GNNs need to use **dense 3D Tensors**, which leads to:

1. Comparatively poor space/time complexity: $O(n^2)/O(n^3)$.
2. Issues with **batching** graph data.

Outline of this Talk

1. What are Graph Neural Networks?
2. **Why do we need new benchmarks?**
3. Our proposed benchmark
4. Our insights from benchmarking + future directions

Issues with current Graph ML datasets

- **Small datasets** → unable to statistically separate architectures.
- **Non-standardized** experimental protocols → reproducibility crisis.
- **Graph-agnostic architectures** (MLP on node features) can often match GNN performance → do we even need GNNs?

$$h_i^{\ell+1} = \text{ReLU}\left(U^\ell h_i^\ell + \sum_{j \in \mathcal{N}_i} V^\ell h_j^\ell\right), \quad \text{vs.} \quad h_i^{\ell+1} = \text{ReLU}\left(U^\ell h_i^\ell\right)$$

No message-passing!!

Issues with current Graph ML datasets

Pitfalls of Graph Neural Network Evaluation

Oleksandr Shchur*, Maximilian Mumme*, Aleksandar Bojchevski, Stephan Günnemann
Technical University of Munich, Germany
{shchur, mumme, a.bojchevski, guennemann}@in.tum.de

A FAIR COMPARISON OF GRAPH NEURAL NETWORKS FOR GRAPH CLASSIFICATION

Federico Errica*
Department of Computer Science
University of Pisa
federico.errica@phd.unipi.it

Marco Podda*
Department of Computer Science
University of Pisa
marco.podda@di.unipi.it

Davide Bacciu*
Department of Computer Science
University of Pisa
bacciu@di.unipi.it

Alessio Micheli*
Department of Computer Science
University of Pisa
micheli@di.unipi.it

Are Powerful Graph Neural Nets Necessary? A Dissection on Graph Classification

Ting Chen*
University of California, Los Angeles
tingchen@cs.ucla.edu

Song Bian
Zhejiang University
songbian@zju.edu.cn

Yizhou Sun
University of California, Los Angeles
yzsun@cs.ucla.edu

Revisiting Graph Neural Networks: All We Have is Low-Pass Filters

Hoang NT*
Tokyo Institute of Technology, RIKEN
hoang.nguyen.rh@riken.jp

Takanori Maehara
RIKEN
takanori.maehara@riken.jp

Benchmarking GNNs Repository

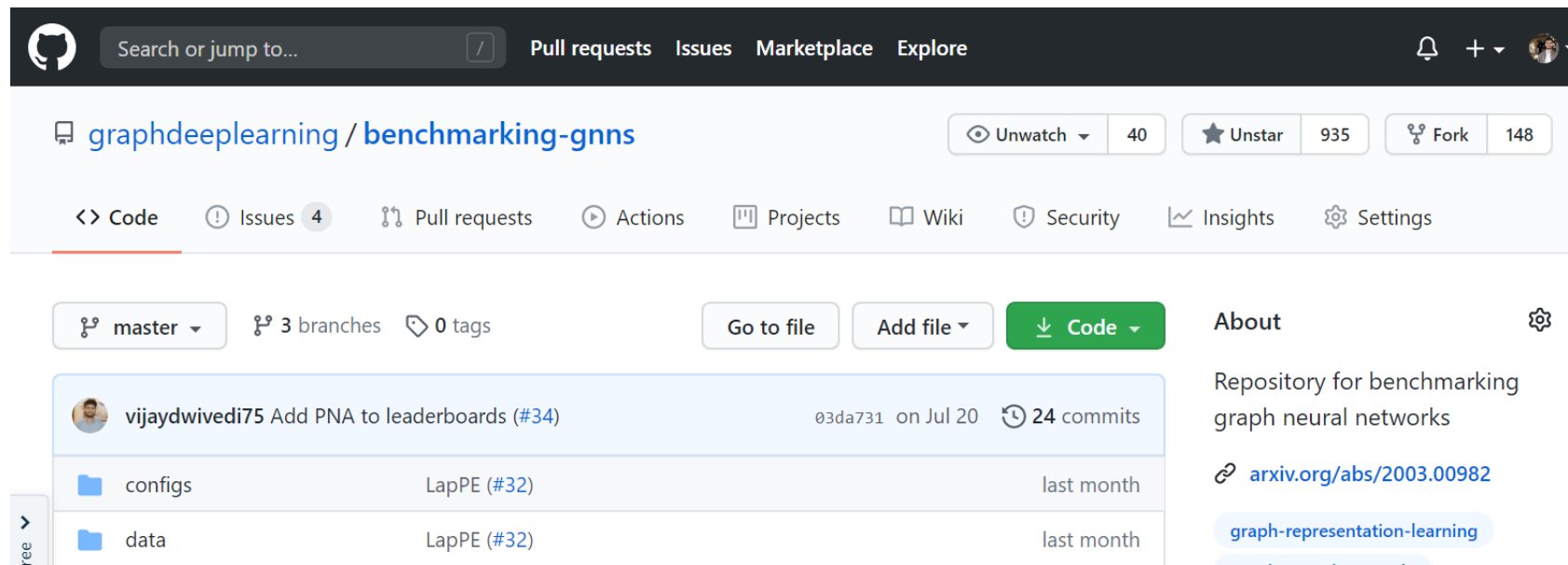
- Our goal: identify **architectures** and **mechanisms** that are universal, generalizable and scalable to large/real-world graphs:
 1. **Datasets** which can statistically separate performance.
 2. **Rigorous** experimental settings and **reproducible** results.
 3. **Future-proof** and **open-source** to enable new advances.
- Complementary + concurrent work: **Open Graph Benchmark**, [Hu-etal, 2020](#): real-world and high-quality graph ML datasets and evaluators.

Outline of this Talk

1. What are Graph Neural Networks?
2. Why do we need new benchmarks?
- 3. Our proposed benchmark**
4. Our insights from benchmarking + future directions

Benchmarking GNNs Repository

- Data pipeline to integrate **any graph ML datasets**.
- GNN layers and models, including **MP-GNNs** and **WL-GNNs**.
- Standardized **Training** and **Evaluation functions**.
- Configurable **hyperparameters** and **scripts** for reproducibility.

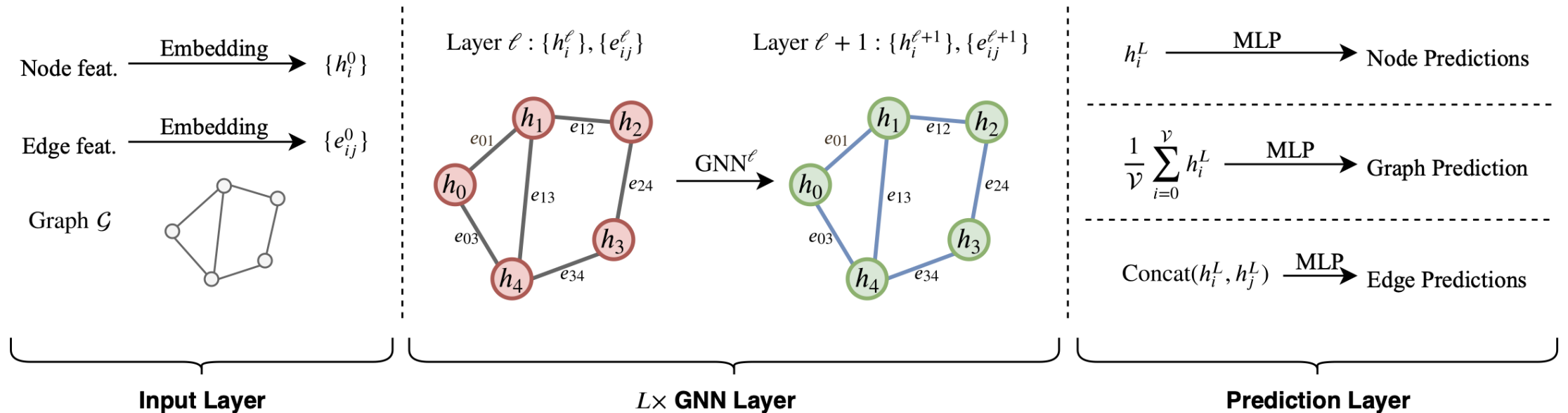


The screenshot shows the GitHub interface for the repository `graphdeeplearning / benchmarking-gnns`. The repository has 40 watchers, 935 stars, and 148 forks. The main navigation bar includes links for Code, Issues (4), Pull requests, Actions, Projects, Wiki, Security, Insights, and Settings. The repository is currently on the `master` branch, with 3 other branches and 0 tags. A recent commit by `vijaydwivedi75` titled "Add PNA to leaderboards (#34)" is shown, dated Jul 20, with 24 commits. The file list includes `configs` and `data`, both linked to issue `LapPE (#32)` and updated last month. The "About" section describes the repository as a "Repository for benchmarking graph neural networks" and provides a link to the arXiv paper `arxiv.org/abs/2003.00982`. A tag `graph-representation-learning` is also visible.



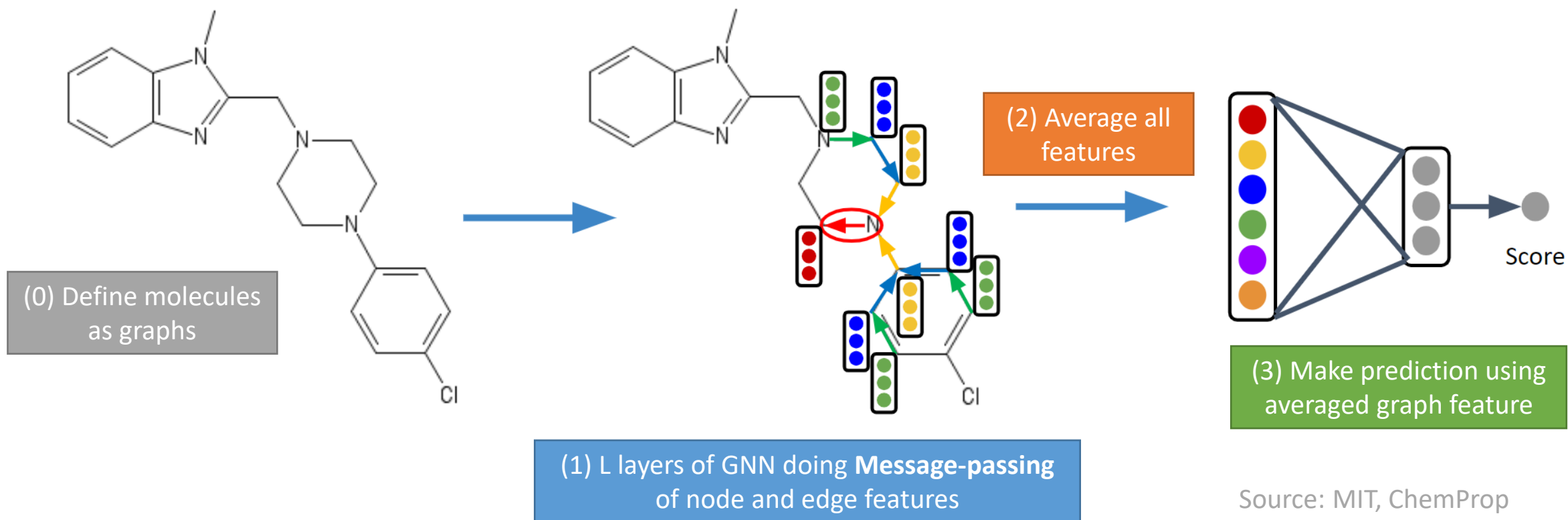
Graph ML Pipeline

Three fundamental tasks: **Node-level, Graph-level, Edge-level**



Example: Molecular Property Prediction

Drug Discovery – screening billions of known molecules to rapidly approximate their properties, e.g. anti-viral activity to COVID-19 or other diseases.



Datasets included with the benchmark

Domain & Construction	Dataset	#Graphs	#Nodes	Total #Nodes	Task
Chemistry: Real-world molecular graphs	ZINC	12K	9-37	277,864	Graph Regression
Mathematical Modelling: Artificial graphs generated from Stochastic Block Models	PATTERN	14K	44-188	1,664,491	Node Classification
	CLUSTER	12K	41-190	1,406,436	
Computer Vision: Graphs constructed with SLIC super-pixels of images	MNIST	70K	40-75	4,939,668	Graph Classification
	CIFAR10	60K	85-150	7,058,005	
Combinatorial Optimization: Uniformly generated artificial Euclidean graphs	TSP	12K	50-500	3,309,140	Edge Classification
Social Networks: Real-world citation graph	COLLAB	1	235,868	235,868	Edge Classification
Circular Skip Links: Isomorphic graphs with same degree	CSL	150	41	6,150	Graph Classification

- We focus on **medium-scale datasets** which are accessible to **academic hardware** capabilities.
- We are also compatible with OGB for **real-world** and **large-scale** graph ML datasets.

Experimental Settings and Best Practices

- **Train/Val/Test splits:** Given with datasets, or random splits.
- LR and Optimizer: **Adam** with **LR decay** strategy.
 - **Initial LR 1e-3/1e-4, halved** every time Val loss doesn't decrease after 10 epochs of patience.
 - Stop training when LR reaches **1e-6** or time \geq 12 hours.
- Statistically significant results through reporting mean and std across **4 different random seeds.**
- Fixed **parameter budgets** for fair comparison:
 1. 100K parameters, 3 or 4 GNN layers
 2. 500K parameters, 8 or 16 GNN layers

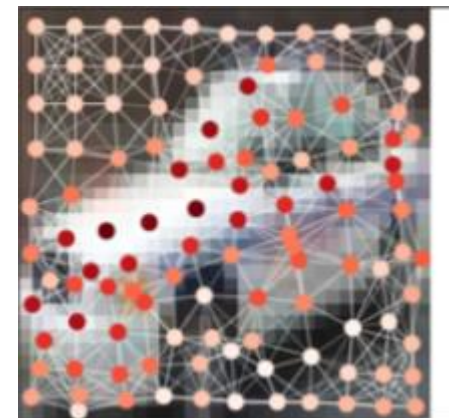
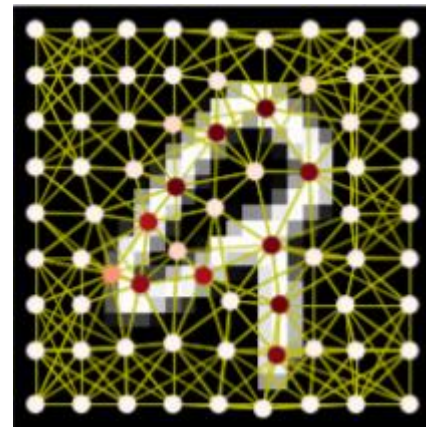
Outline of this Talk

1. What are Graph Neural Networks?
2. Why do we need new benchmarks?
3. Our proposed benchmark
4. **Our insights from benchmarking + future directions**

Graph Classification – MNIST & CIFAR10

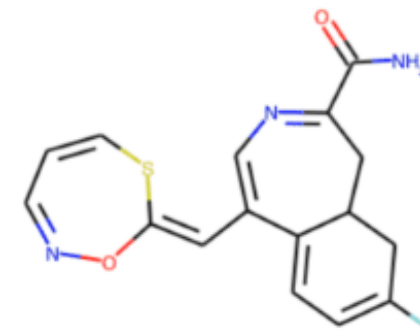
Model	Configurations	MNIST Accuracy	CIFAR10 Accuracy
Graph-agnostic MLP	100K, 4L	95.34 +- 0.13	56.34 +- 0.18
Simple MP-GNN	100K, 4L, GraphSage	97.31 +- 0.09	65.76 +- 0.30
Anisotropic MP-GNN	100K, 4L, Gated GCN	97.34 +- 0.14	67.31 +- 0.31
3WL-GNN	100K, 3L	95.07 +- 0.96	59.17 +- 1.59

- MNIST and CIFAR10 are **sanity checks**.
- GNNs don't match CNNs for image classification, yet:
 - But this is to be expected as CNNs are specialized to images.
 - For CV, GNNs are more useful for **3D shapes** or **scene graphs**.



Graph Regression – ZINC

Model	Configurations	Regression MAE
Graph-agnostic MLP	100K, 4L	0.70 +- 0.00
Simple MP-GNN	100K, 4L, GCN	0.45 +- 0.00
Anisotropic MP-GNN	100K, 4L, Gated GCN	0.37 +- 0.00
3WL-GNN	100K, 3L	0.25 +- 0.05
Simple MP-GNN	500K, 16L, GCN	0.36 +- 0.01
Anisotropic MP-GNN	500K, 16L, Gated GCN	0.28 +- 0.01
3WL-GNN	500K, 8L	0.30 +- 0.05

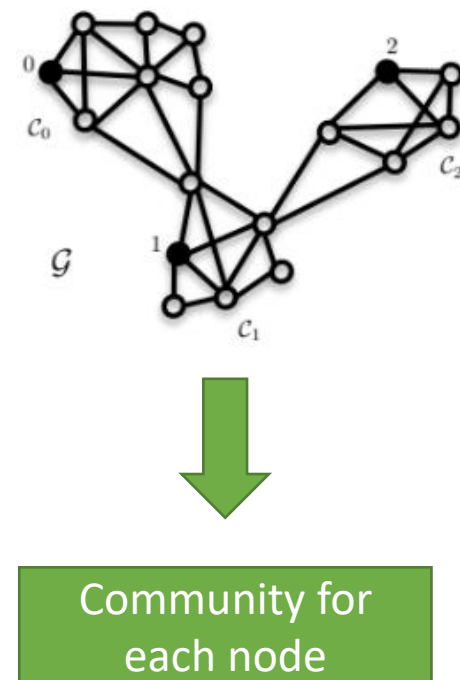


Chemical property

- **Graph structure** is needed: GNNs outperform MLPs.
- **Anisotropy** improves MP-GNN performance.
- WL-GNNs are powerful, but **difficult to scale** (in #params as well as speed).

Node Classification – Semi-sup. Clustering

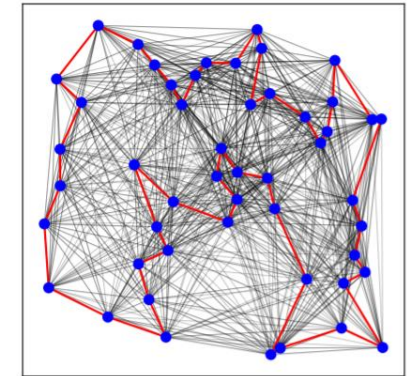
Model	Configurations	Accuracy (weighted)
Graph-agnostic MLP	100K, 4L	20.97 +- 0.00
Simple MP-GNN	100K, 4L, GCN	53.44 +- 2.02
Anisotropic MP-GNN	100K, 4L, Gated GCN	60.40 +- 0.41
3WL-GNN	100K, 3L	57.13 +- 6.53
Simple MP-GNN	500K, 16L, GCN	68.49 +- 0.97
Anisotropic MP-GNN	500K, 16L, Gated GCN	73.84 +- 0.32
3WL-GNN	500K, 3L (8L diverges)	55.48 +- 7.86



- **Anisotropy** is crucial: **Softmax-attention** is flexible over max/mean/sum.
- WL-GNNs are difficult to scale, especially on **larger graphs than chemistry**: $O(n^2)/O(n^3)$ space/time complexity; use of dense **3D tensors**.

Link Prediction – Travelling Salesman Problem

Model	Configurations	F1 Score	Epoch Time
Graph-agnostic MLP	100K, 4L	0.54 +- 0.00	50.15 s
Simple MP-GNN	100K, 4L, GCN	0.63 +- 0.00	105.89 s
Anisotropic MP-GNN	100K, 4L, Gated GCN	0.80 +- 0.00	218.20 s
3WL-GNN	100K, 3L	0.69 +- 0.07	17,468.81 s



Yes/No for each edge

- Anisotropy is **especially powerful for edge tasks** → **More analysis in paper.**
 - But comes at computational cost over simple GCNs.
- Clearly, there's a need for practical **tradeoffs** between **expressivity** and **performance** in GNN architectures.
 - **In the paper** → novel **positional encodings** based on spectral graph theory for MP-GNNs.

Open Problems

1. **Next generation of GNNs** which lead to meaningful real-world improvements over the message-passing paradigm.
2. Understanding the tradeoffs between **theoretical expressivity** and **computational tractability** as well as **generalization**.
3. **Structure discovery**: GNNs operate on *a priori* graph structure, but this may be incomplete/noisy/containing latent interactions, etc.
4. **Scientific applications**: how to better augment GNNs with domain knowledge to accelerate scientific discovery?

Bonus Content!

- Transformers from NLP are GNNs in disguise?
- GNNs for data-driven Combinatorial Optimization

Transformers are GNNs?



Full post:

thegradient.pub/transformers-are-graph-neural-networks/



Transformers are Graph Neural Networks

12.SEP.2020



Chaitanya K. Joshi

| RECENT STORIES

1. Shortcuts: How Neural Networks Love to Cheat



My engineering friends often ask me: deep learning on graphs sounds great, but are there any real applications?

While Graph Neural Networks are used in recommendation systems at [Pinterest](#), [Alibaba](#) and [Twitter](#), a more subtle success story is the **Transformer architecture**, which has [taken the NLP world by storm](#). Through this post, I want to establish a link between [Graph Neural Networks \(GNNs\)](#) and Transformers. I'll talk about the intuitions behind model architectures in the NLP and GNN communities, make connections using equations and figures, and discuss how we can work together to drive future progress. Let's start by talking about the purpose of model architectures—*representation learning*.

Oriol Vinyals @OriolVinyalsML

Transformers are a special case of Graph Neural Networks. This may be obvious to some, but the following blog post does a good job at explaining these important concepts.

Chaitanya Joshi @chaitjo

Excited to share a blog post on the connection between #Transformers for NLP and #GraphNeuralNetworks (GNNs or GCNs).

graphdeeplearning.github.io/post/transform...

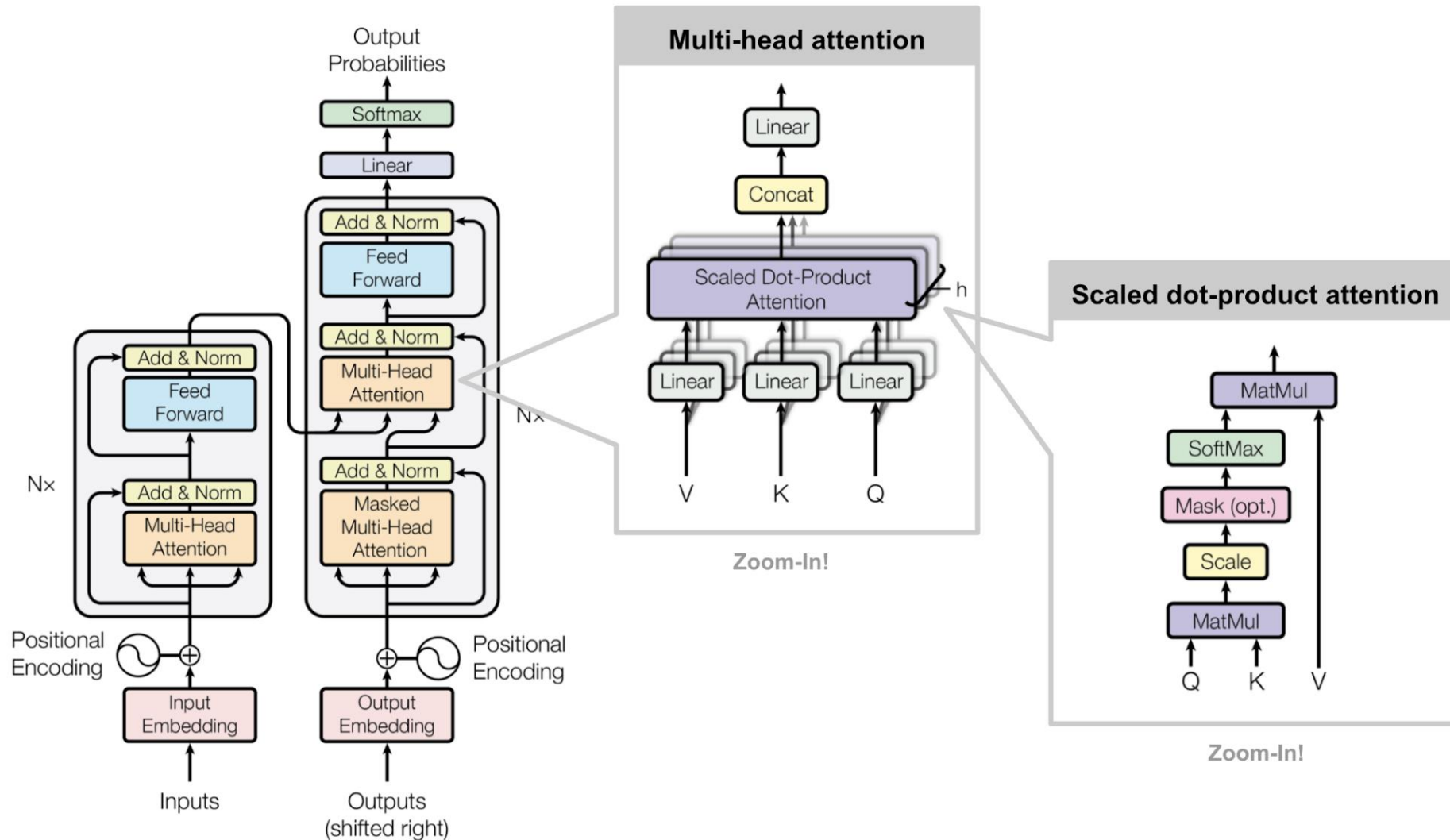
The diagram shows two neural network architectures for the sentence "This is a sentence".

- RNN:** A linear sequence of nodes: This → is → a → sentence. An arrow labeled "RNN" points to the right, with associated tasks: Translation?, Sentiment?, Next word?.
- Transf.:** A graph structure where nodes are interconnected. Nodes: This, is, another, sentence. Arrows show connections between "This" and "is", "is" and "another", "another" and "sentence", and "This" and "sentence". A red arrow labeled "Transf." points to the right, with associated tasks: Part-of-speech tags?.

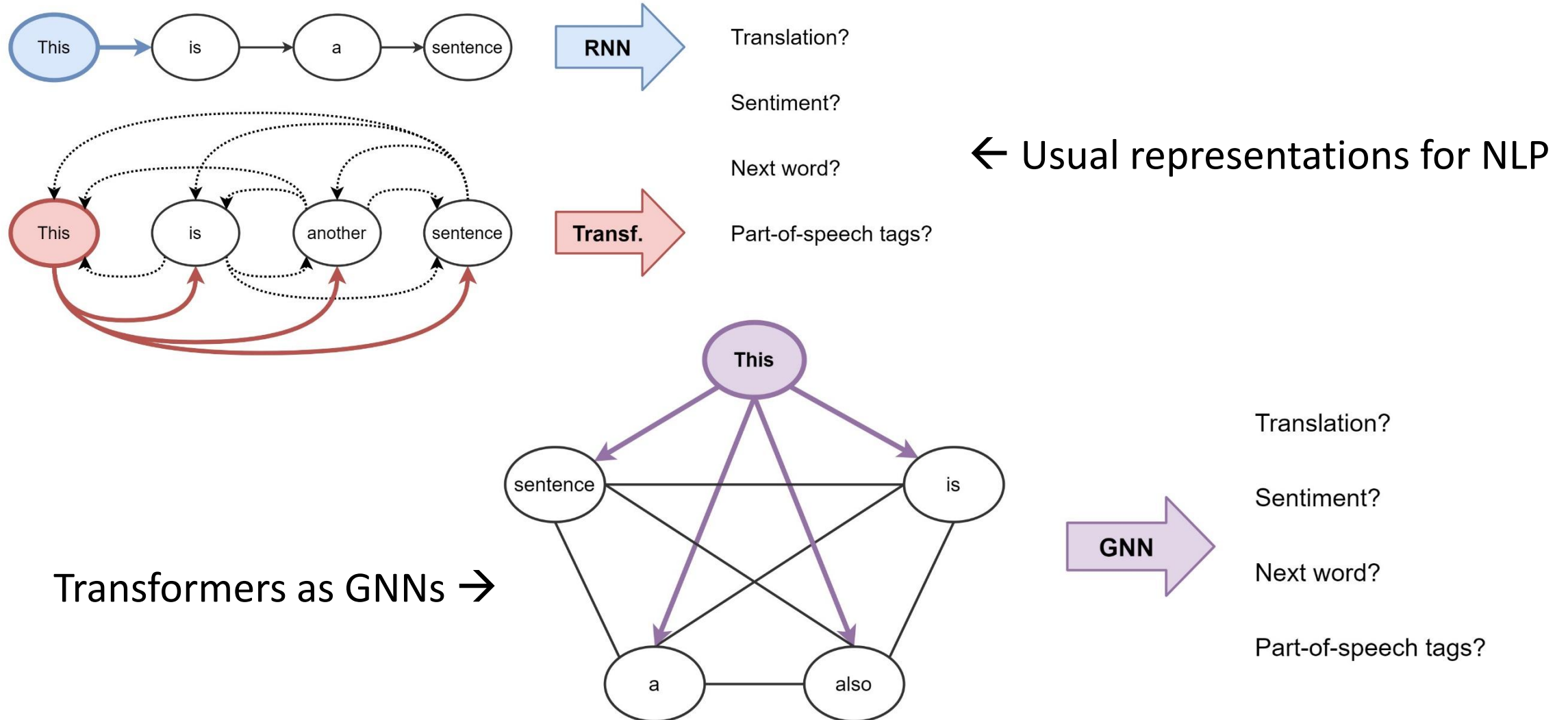
11:58 PM · Feb 29, 2020

971 likes, 244 people are Tweeting about this

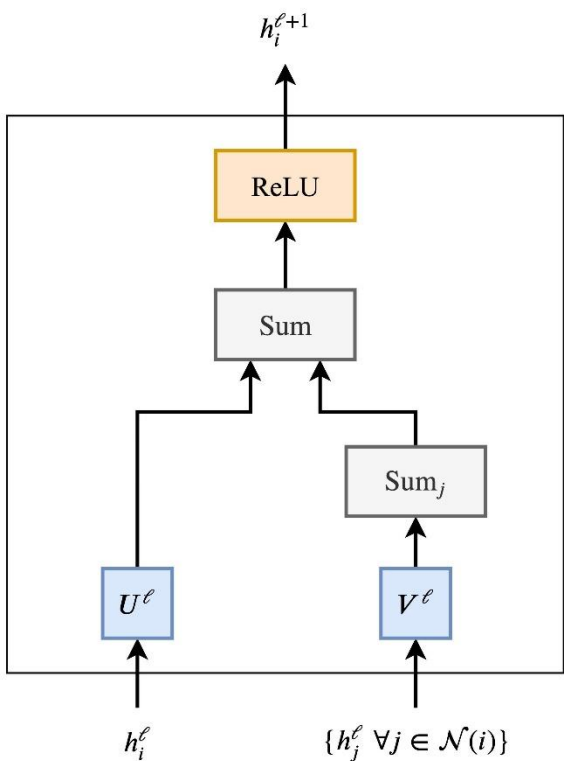
Transformer Architecture and (Multi-head) Attention



Sentence as Fully-connected Word Graphs



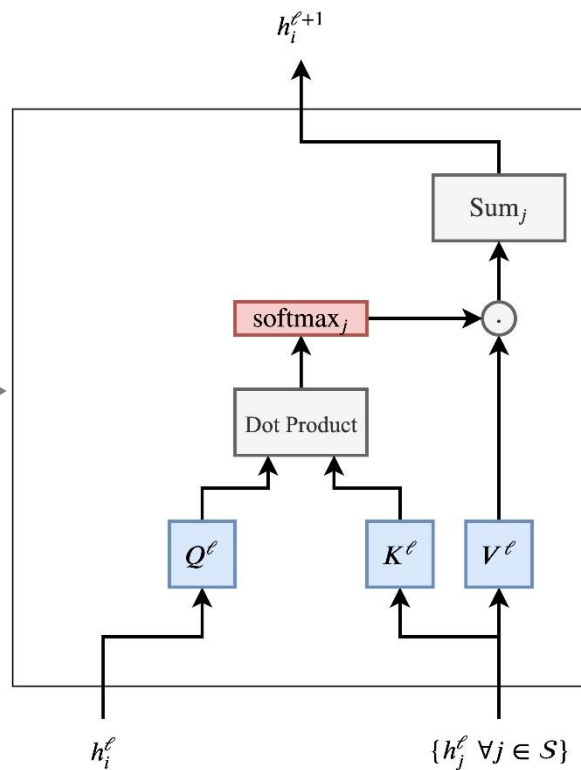
Standard GNN



+ Weighted sum aggregation

Full post: thegradient.pub/transformers-are-graph-neural-networks/

Graph Attention Network

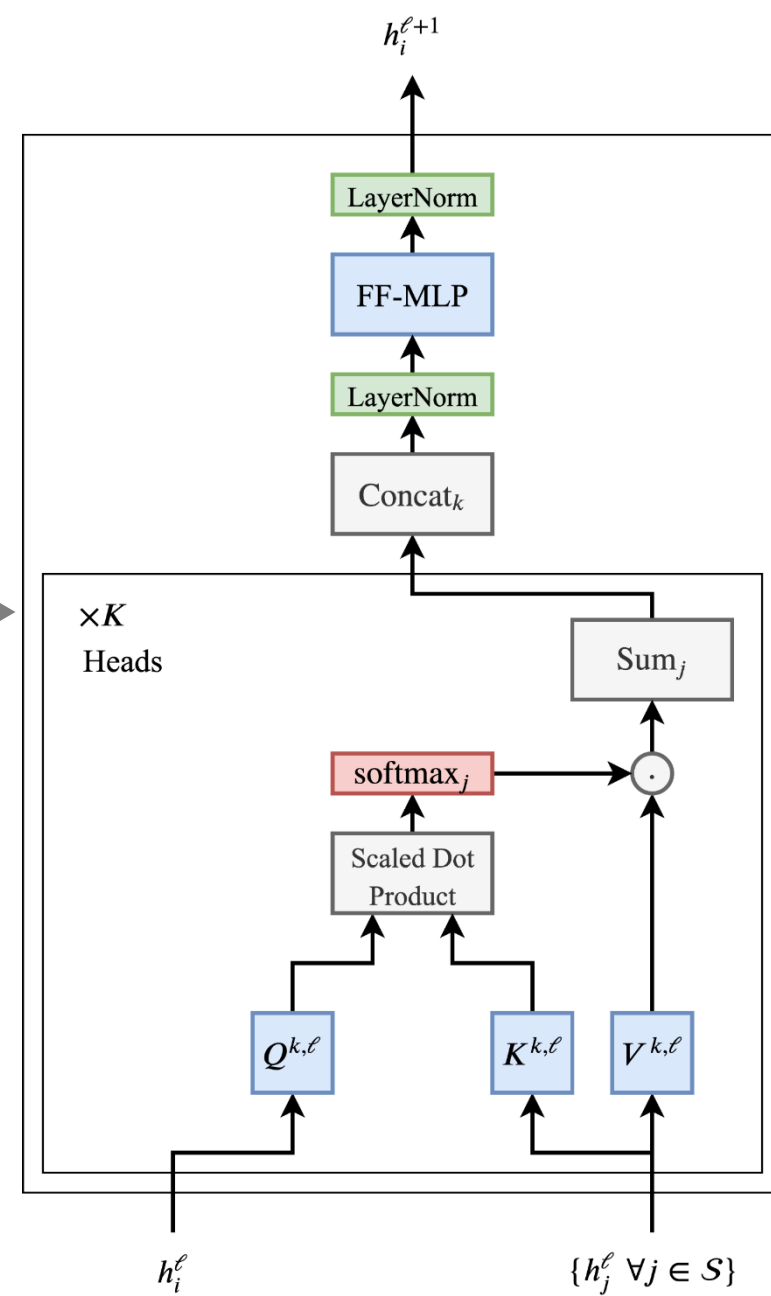


+ Multi-head mechanism

+ Normalization layers

+ Residual links

Transformer



GNNs for Combinatorial Optimization

GNNs in combination with **Reinforcement Learning** are an interesting approach towards solving **NP-hard** optimization problems defined via graphs:

1. Classical problems: **Travelling Salesman**, Minimum Vertex Cover, Satisfiability
2. Physical sciences: Generating Graphs, **Generating Molecules**, Drug Discovery
3. Computer architecture: **Device Placement Optimization**, Chip Design



Further reading: <https://graphdeeplearning.github.io/project/combinatorial-optimization/>

End-to-end Comb. Opt. Pipeline

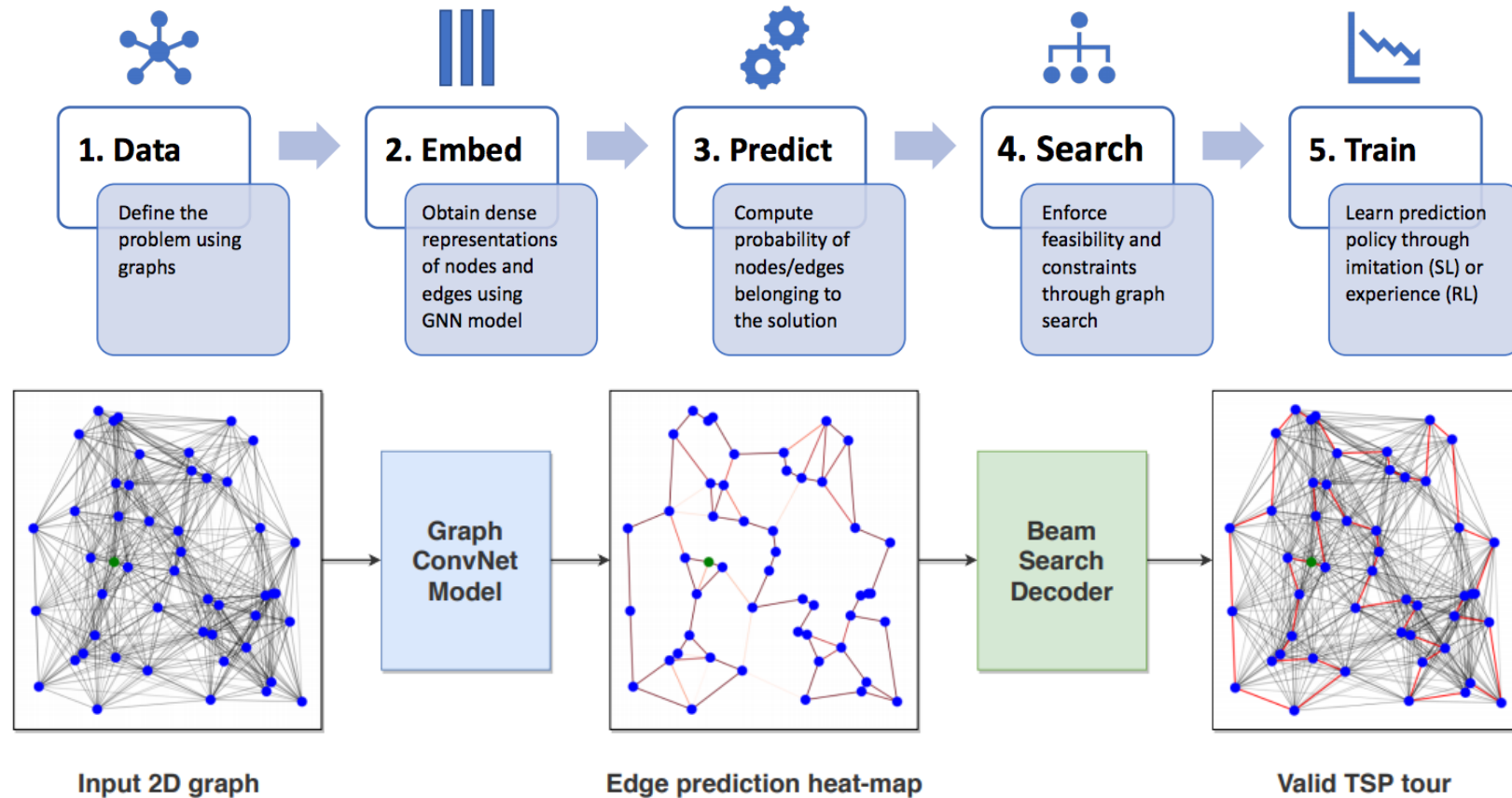


Figure 1: Overview of our approach. Taking a 2D graph as input, the graph ConvNet model outputs an edge adjacency matrix denoting the probabilities of edges occurring on the TSP tour. This is converted to a valid tour using beam search. All components are highly parallelized and solutions are produced in a one-shot, non-autoregressive manner.

The End. Questions? 😊