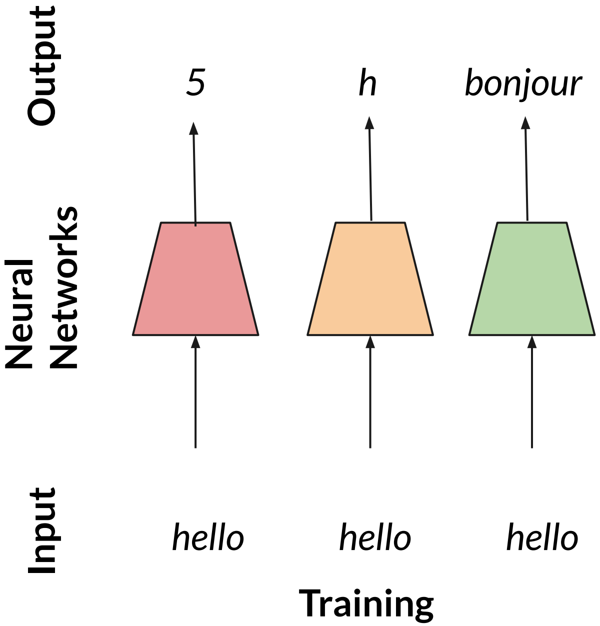
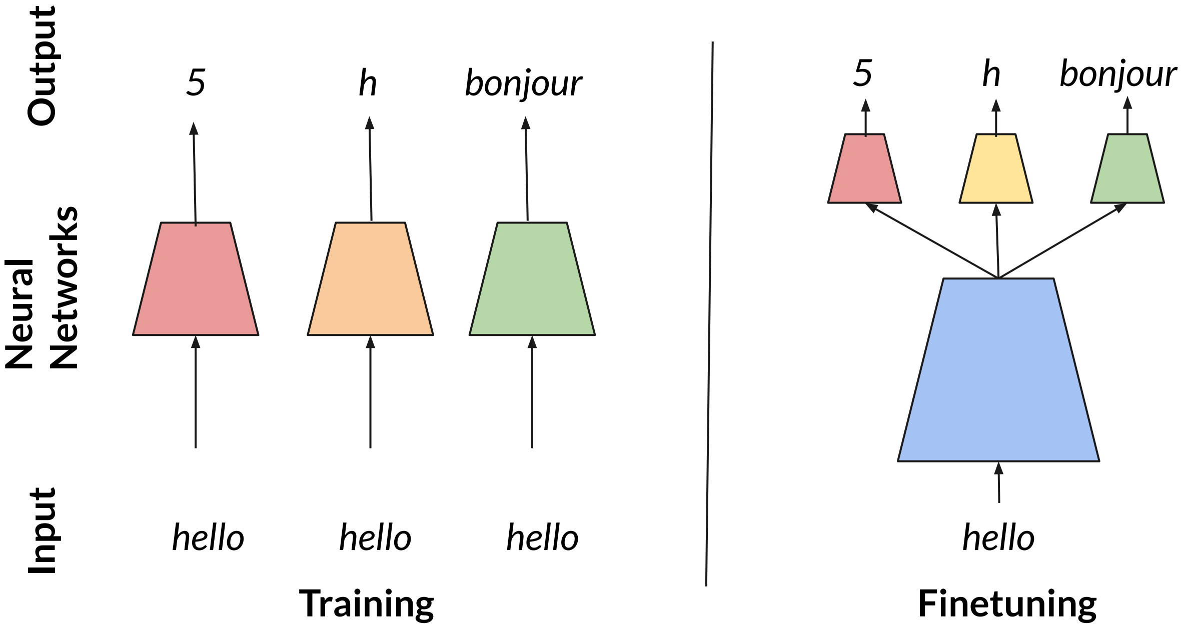
PRODIGY: Enabling In-context Learning Over Graphs

CS224W: Machine Learning with Graphs
Qian Huang, Stanford University
11/28

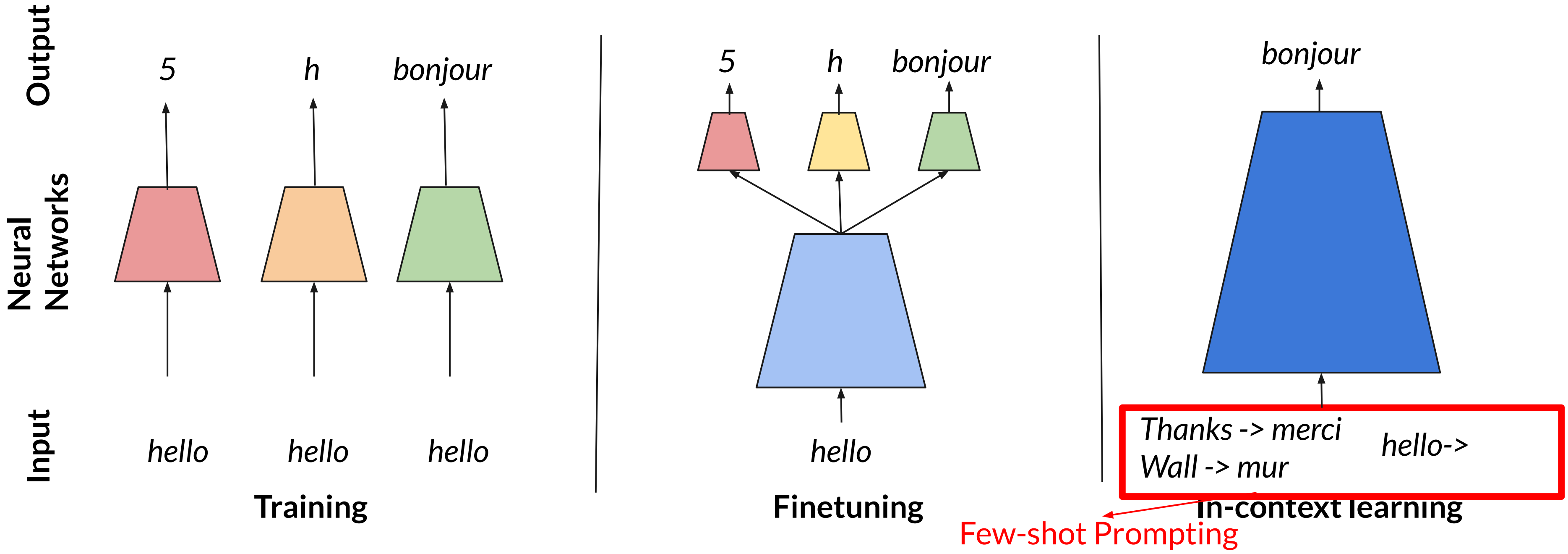
Huang, Q., Ren, H., Chen, P., Krvzmanc, G., Zeng, D.D., Liang, P., & Leskovec, J. (2023). PRODIGY: Enabling In-context Learning Over Graphs. NeurIPS 2023


Output
Neural
Networks
Input
5
h
bonjour
hello
hello
hello
Training

Different Machine Learning Paradigms

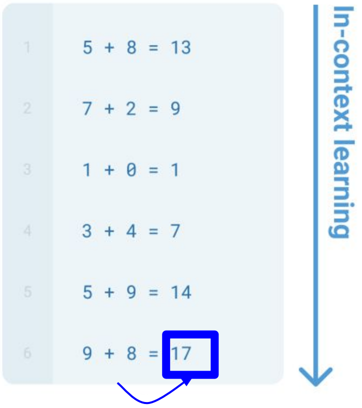


Different Machine Learning Paradigms



In-context Learning

Performing a new task by “learning” from the input context w/o gradient update. Very powerful — we only need one foundation model directly answering all tasks now!



Predicting sum

Unscrambling

translating

In-context Learning Over Graphs

Thanks -> merci
Wall -> mur

hello->?

Few-shot prompting over text

What is in-context learning over graphs?

Today's Plan

We formulate and enable **in-context learning over graphs**

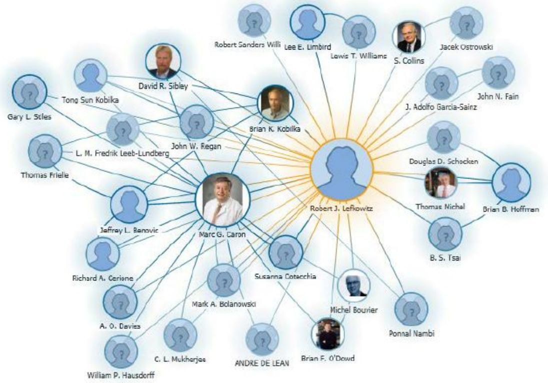
- **Formulation** : An in-context learner for graphs should be able to solve novel tasks on novel graphs.
- PRODIGY:
    - **Prompt Graph representation:** represent the few-shot prompt for different graph tasks in the same input format, so that it can be consumed by one shared model
    - **Prompt Graph Inference:** in-context prediction GNN
    - **Pretraining:** generate diverse pretraining tasks in the format of PromptGraph
        - Neighbor Matching
        - MultiTask

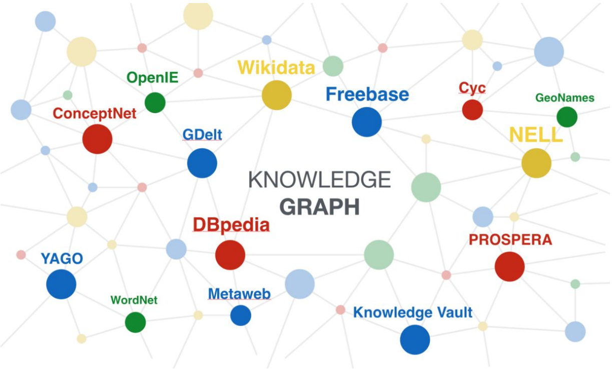
Formulation

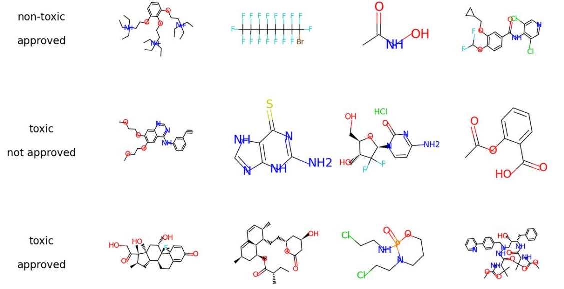
Graph Learning Tasks

What are the tasks on graphs?

- Node classification
- Link Prediction
- Graph classification







In-context Learning Over Graphs

Thanks -> merci
Wall -> mur

hello->?

Few-shot prompting over text

What is in-context learning over graphs?

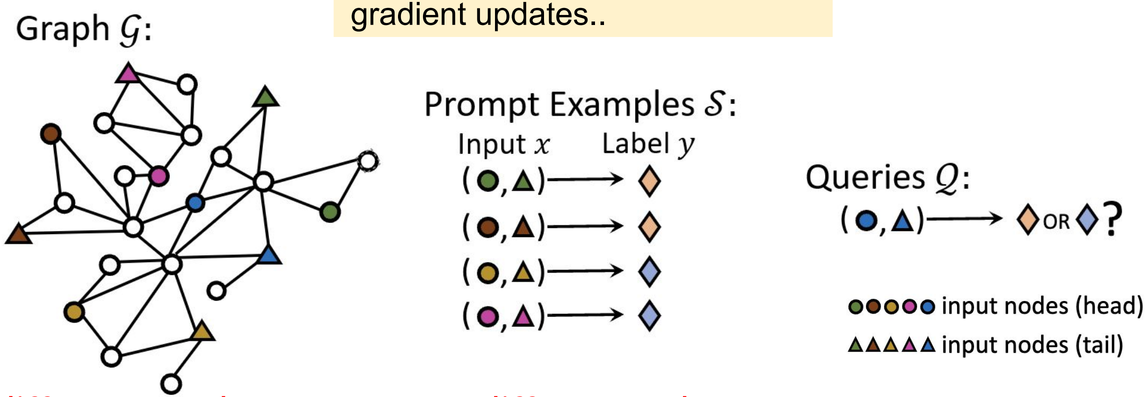
In-context Learning Over Graphs: Link Prediction Example

An in-context learner for graphs should be able to solve novel tasks on novel graphs without gradient updates..

Thanks -> merci
Wall -> mur

hello->?

different tasks



different graphs

different tasks

Few-shot prompting over text

Few-shot prompting over graph (for link classification)

But, how to achieve this? Two Challenges:

1. How to represent the few-shot prompt for **different graph tasks** in the **same input format**, so that it can be consumed by one shared model?

2. How to **pretrain** a model that can solve any task in this format?

But, how to achieve this? Two Challenges:

1. How to represent the few-shot prompt for **different graph tasks** in the **same input format**, so that it can be consumed by one shared model?

   => **Prompt Graph**: represent each few-shot prompt over graph as a meta hierarchical graph

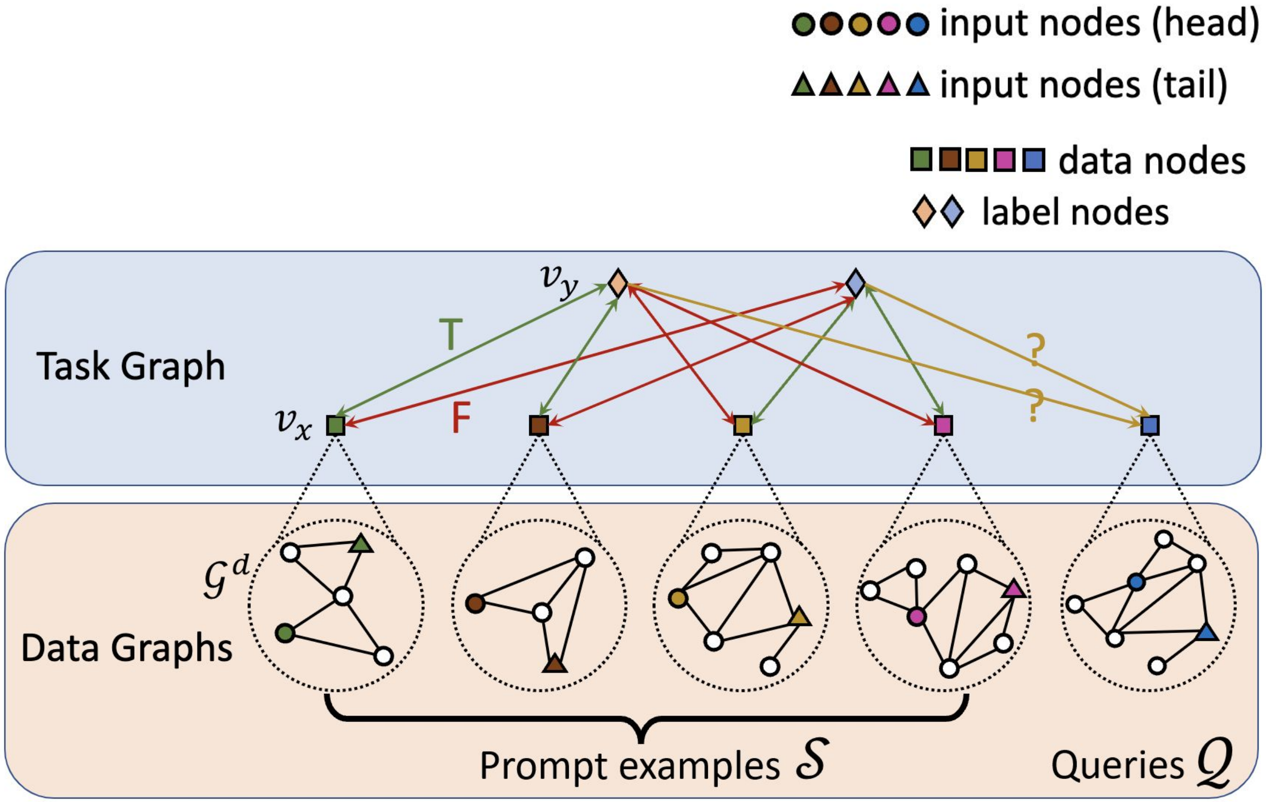
2. How to **pretrain** a model that can solve any task in this format?

   => **PGPretraining**: Pretrain a message passing model over self-supervised tasks in PromptGraph format with diverse underlying structures

Prompt Graph

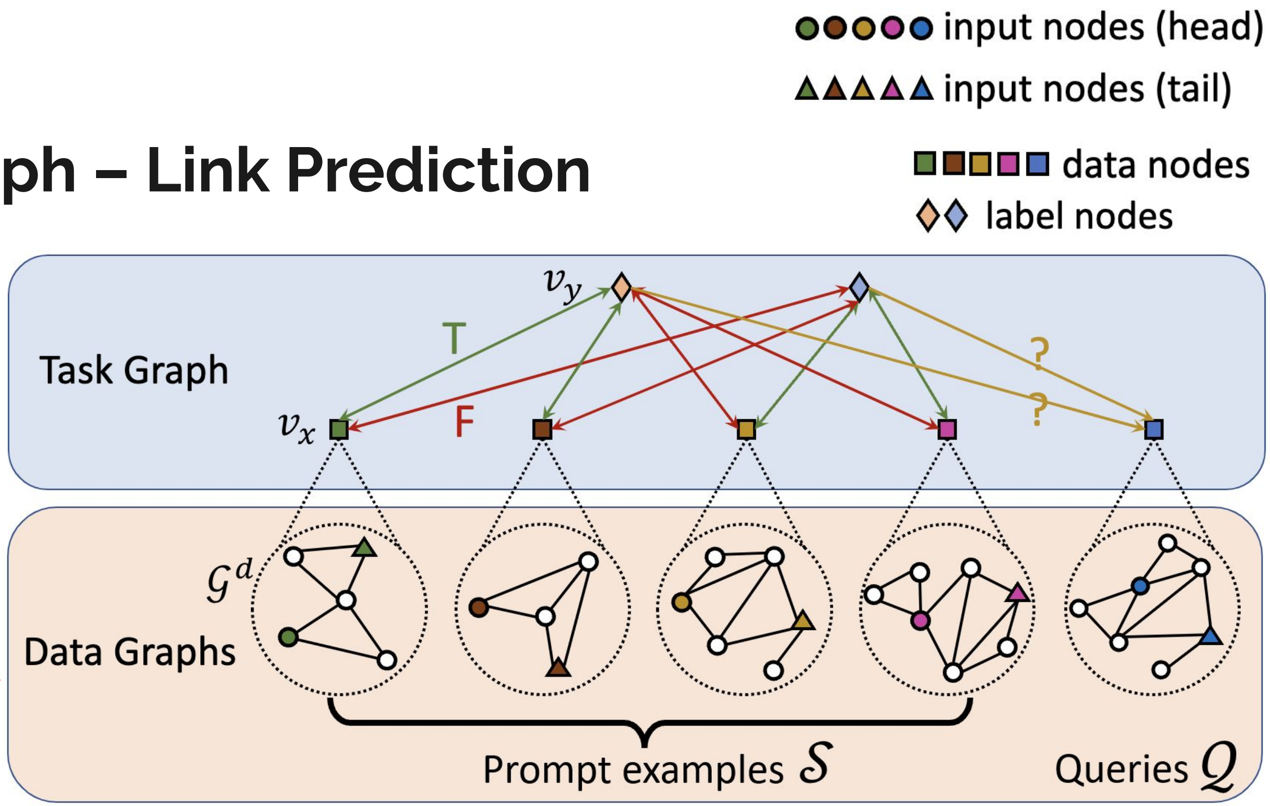
Prompt Graph

Prompt Graph is a unified representation of few-shot prompts over graph for diverse tasks



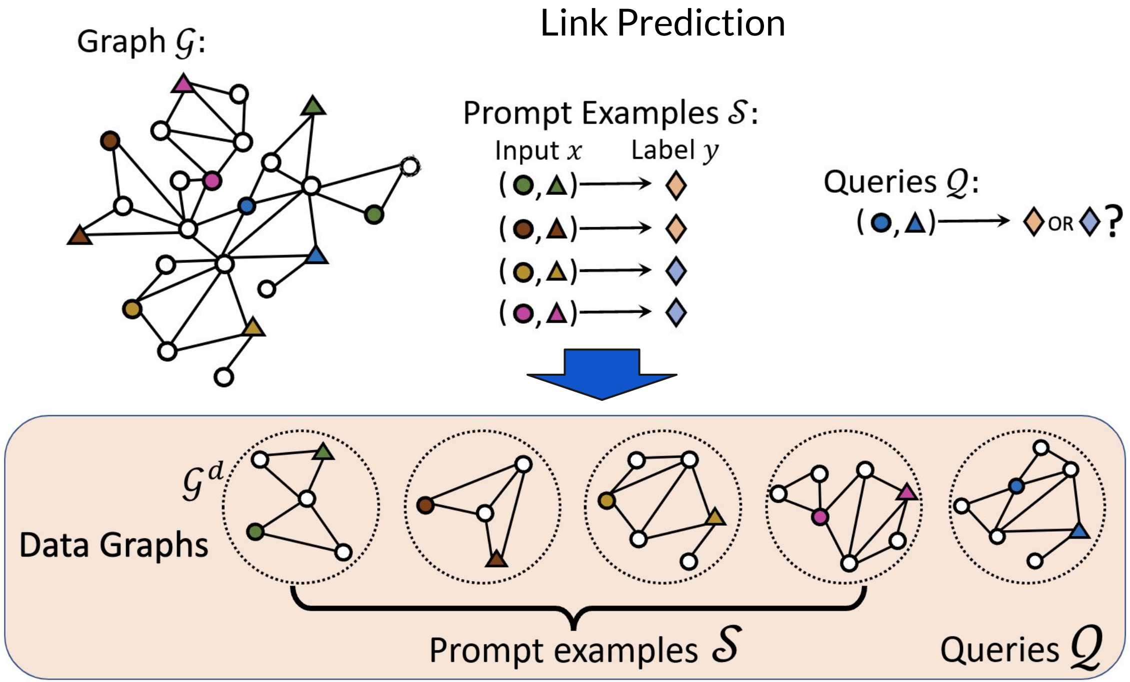
Step1: Data Graph – Link Prediction

Data Graph contextualizes each input x in the graph G (e.g. by **subgraph extraction)**



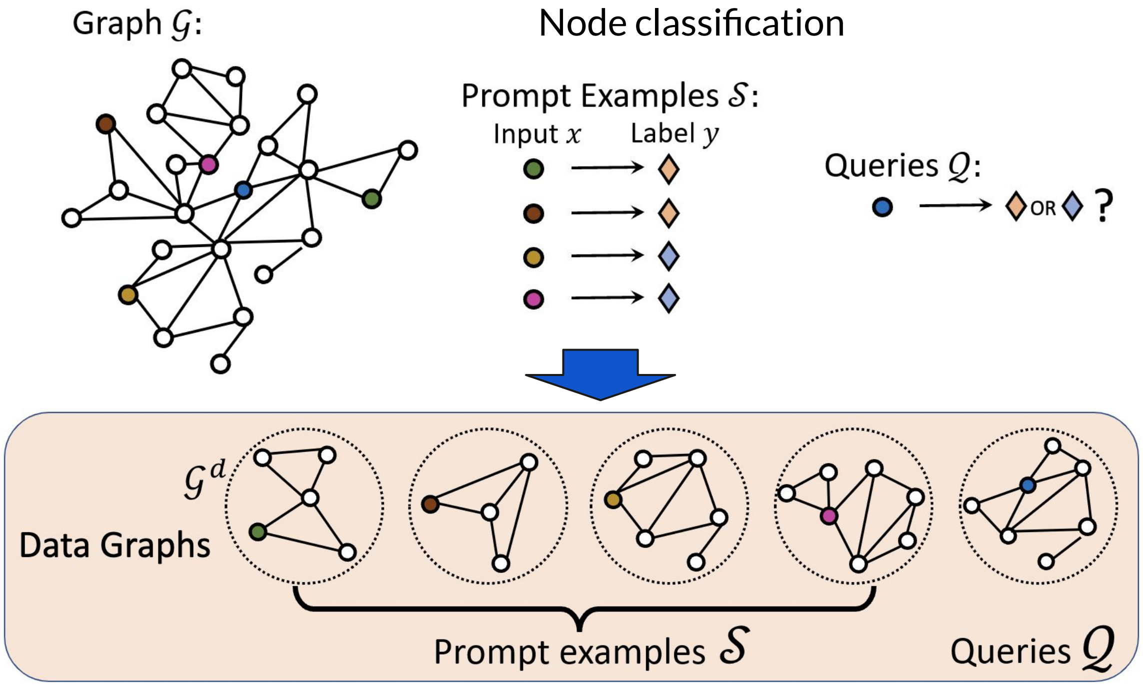
Step1: Data Graph – Link Prediction

Data Graph contextualizes each input x in the graph G (e.g. by **subgraph extraction)**



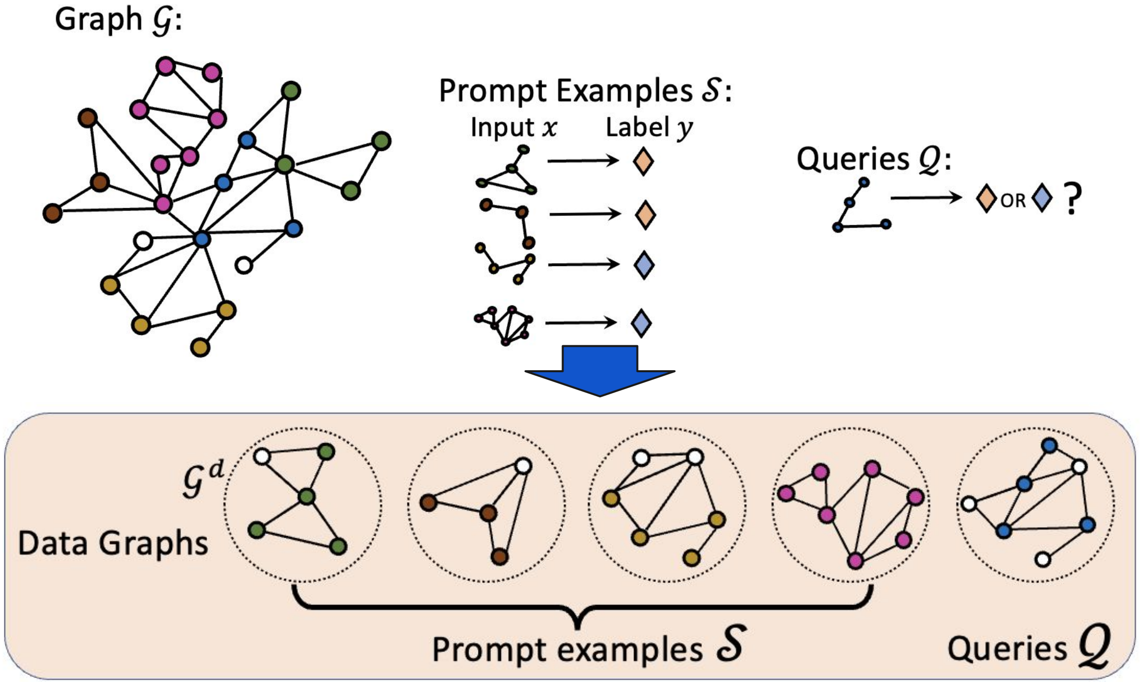
Step1: Data Graph – Node Classification

Data Graph contextualizes each input x in the graph G (e.g. by **subgraph extraction)**



Step1: Data Graph – Graph Classification

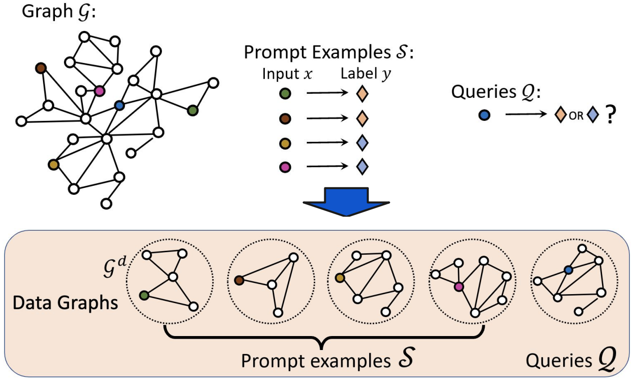
Data Graph contextualizes each input x in the graph G (e.g. by **subgraph extraction)**



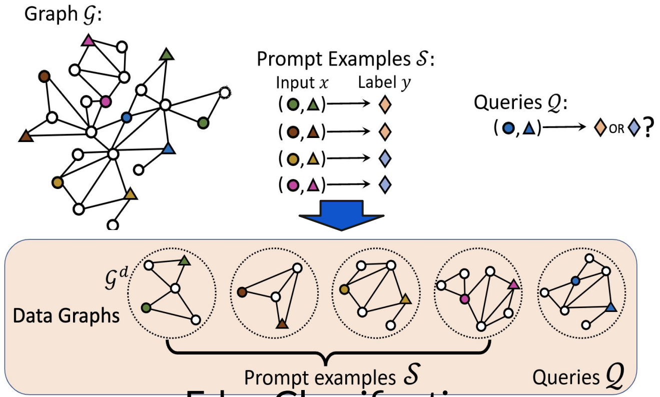
Step1: DataGraph Construction

DataGraph unifies input format:

- Use text feature to unify features over different datasets
- Use different **input node set** for different classification over different levels (nodes vs edge vs graph)



Node Classification

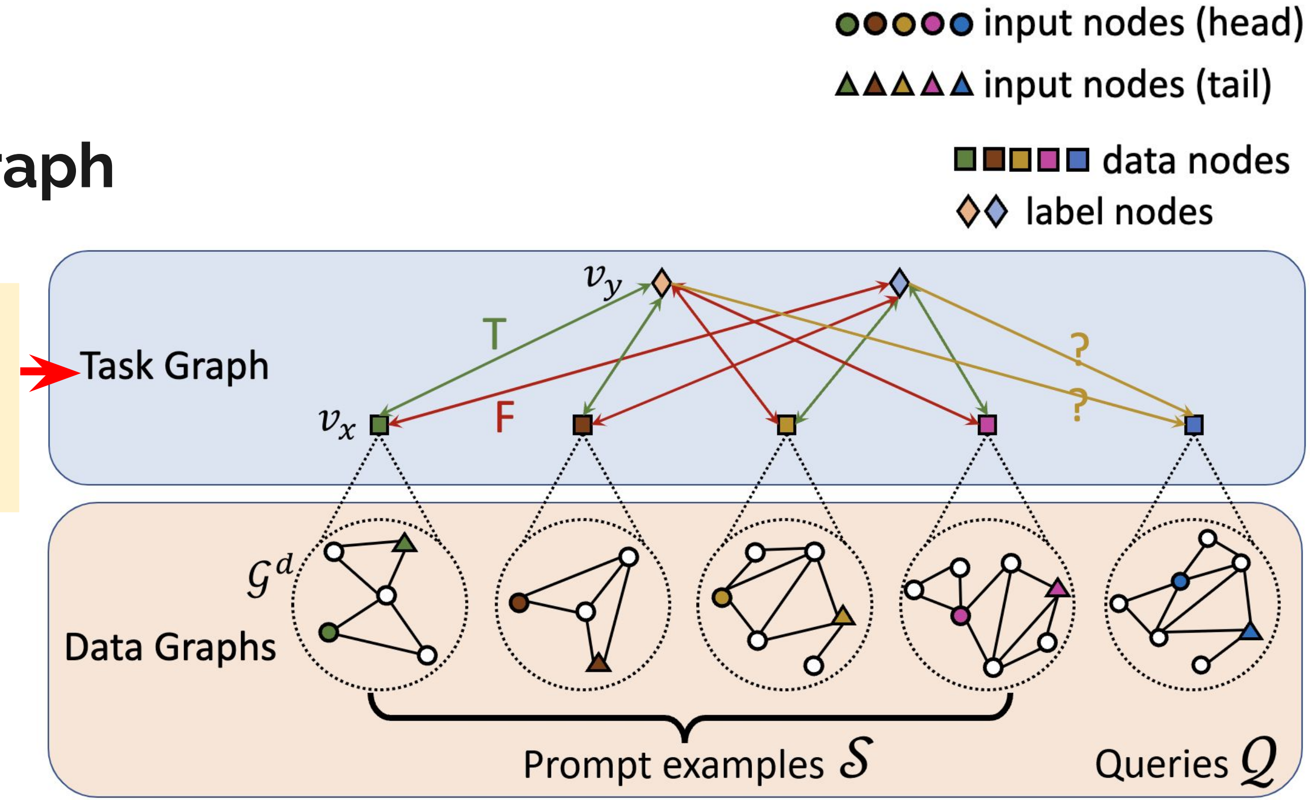


Edge Classification

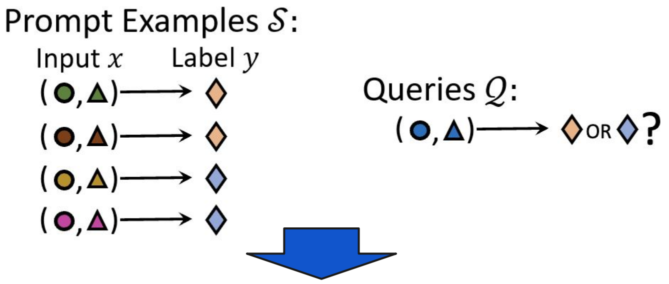
...

Step2: Task Graph

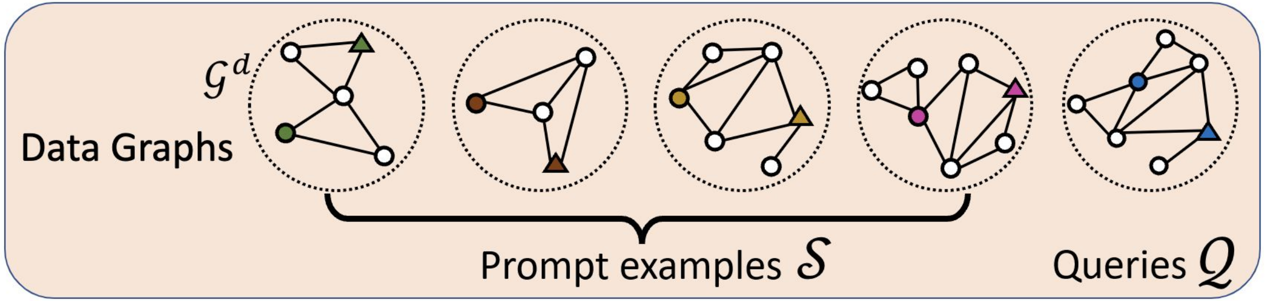
Task Graph interconnects inputs and labels across examples to form context for queries



Step2: Task Graph – Link Prediction

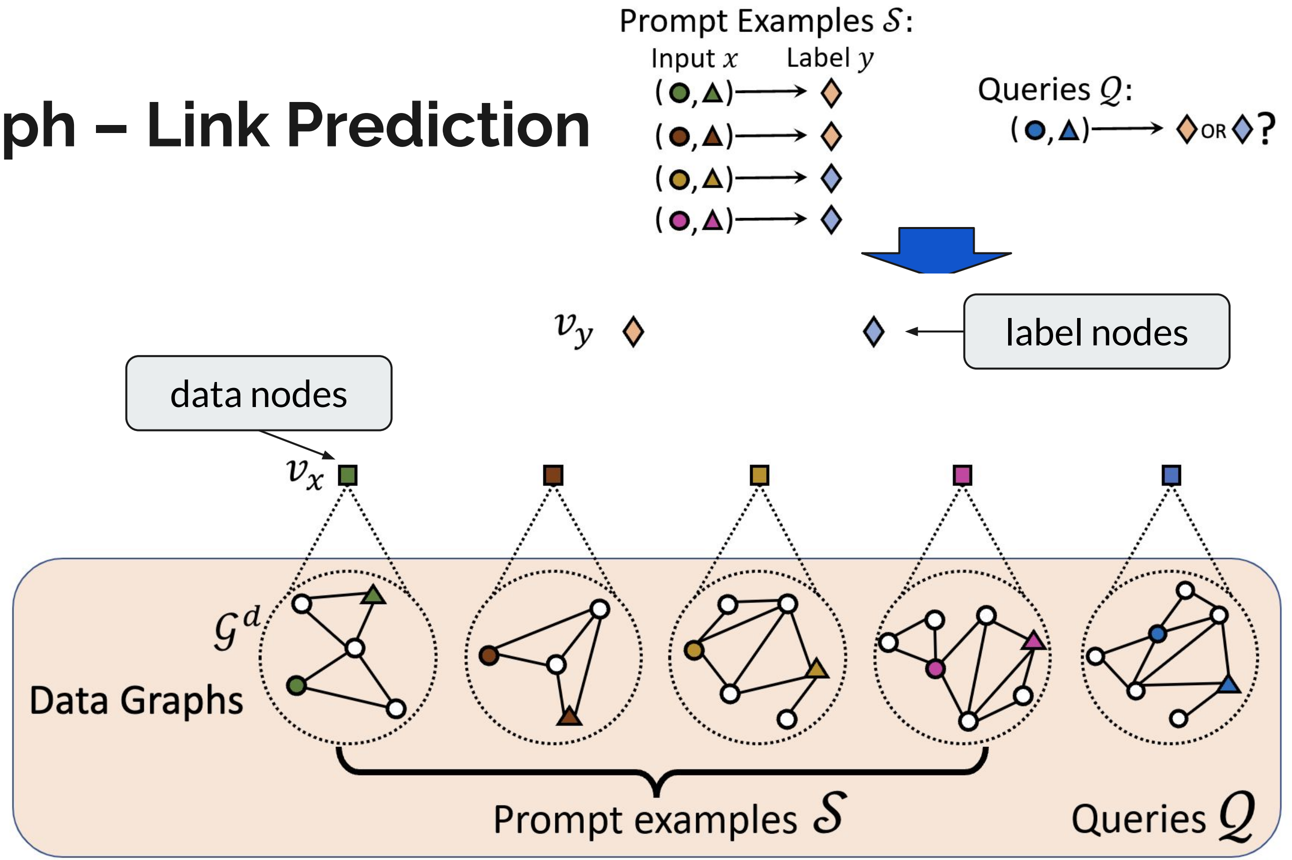


Task Graph interconnects inputs and labels across examples to form context for queries



Step2: Task Graph – Link Prediction

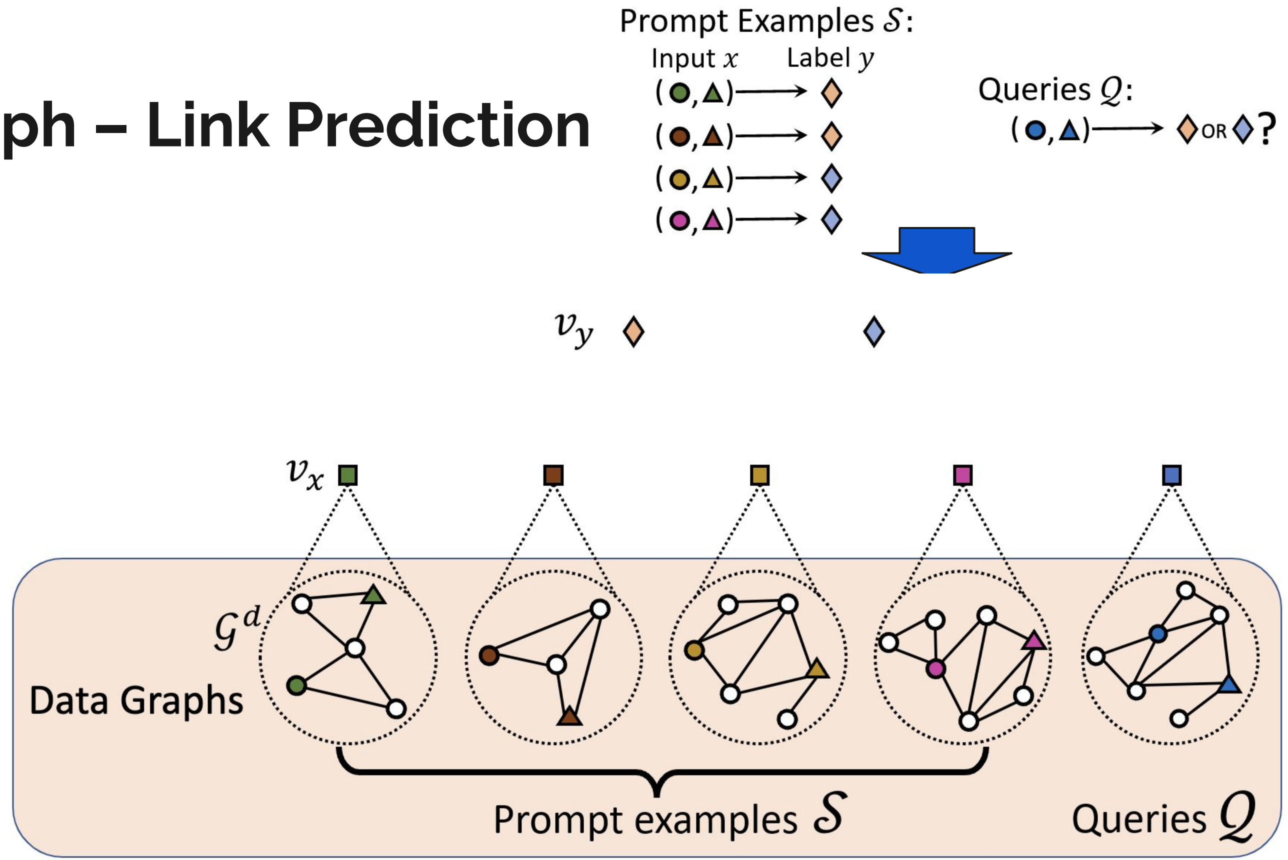
Task Graph interconnects inputs and labels across examples to form context for queries



Step2: Task Graph – Link Prediction

Task Graph interconnects inputs and labels across examples to form context for queries

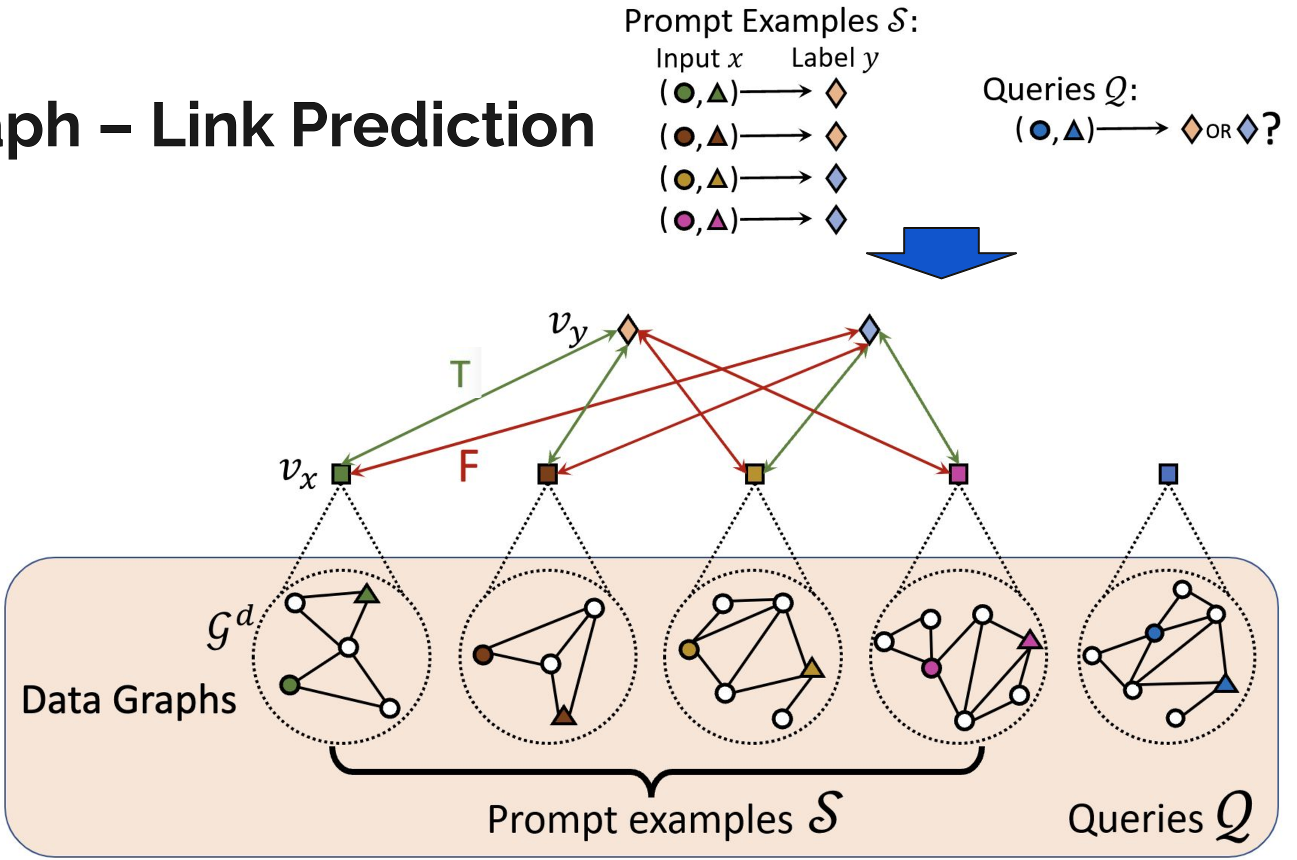
- **Prompt examples**: bidirectional edges between data nodes and all label nodes



Step2: Task Graph – Link Prediction

Task Graph interconnects inputs and labels across examples to form context for queries

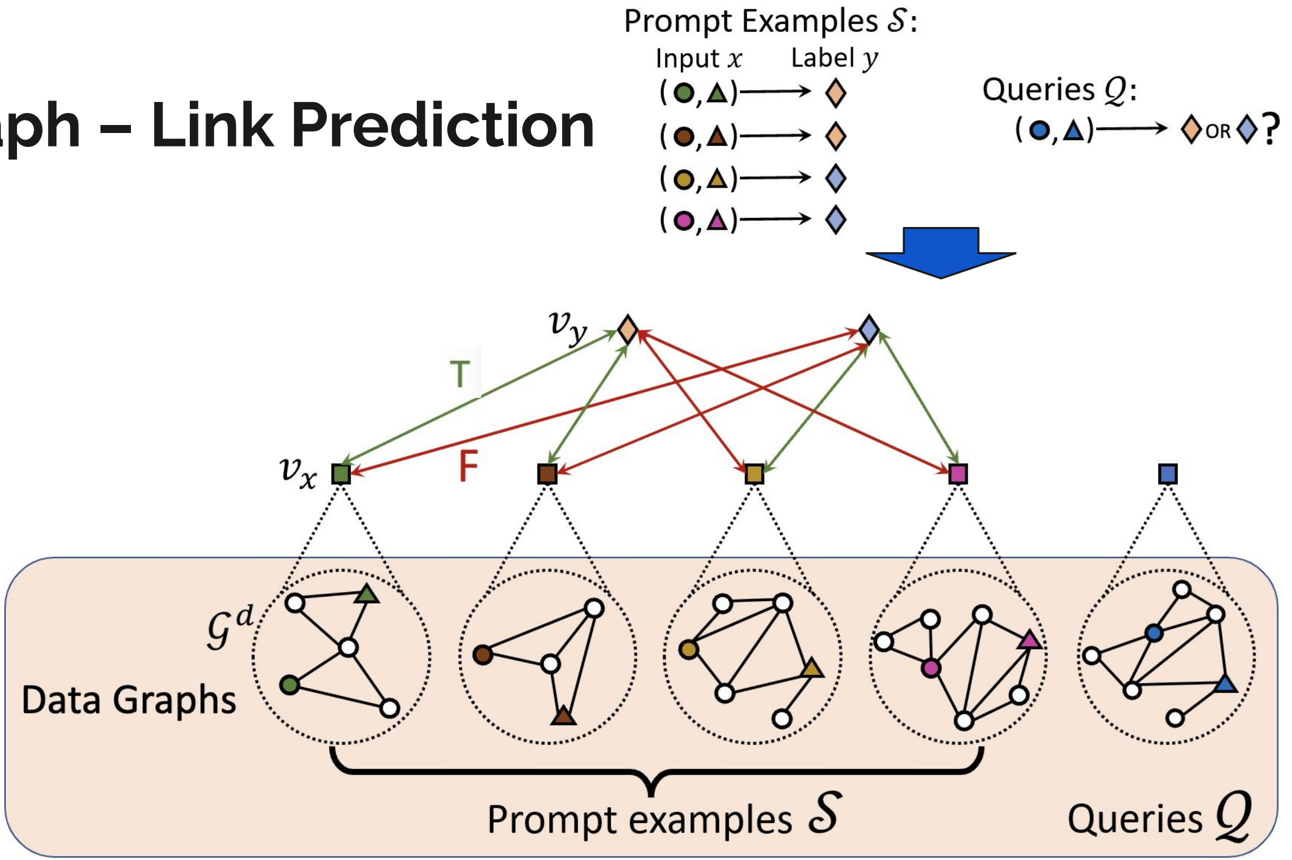
- **Prompt examples**: bidirectional edges between data nodes and all label nodes



Step2: Task Graph – Link Prediction

Task Graph interconnects inputs and labels across examples to form context for queries

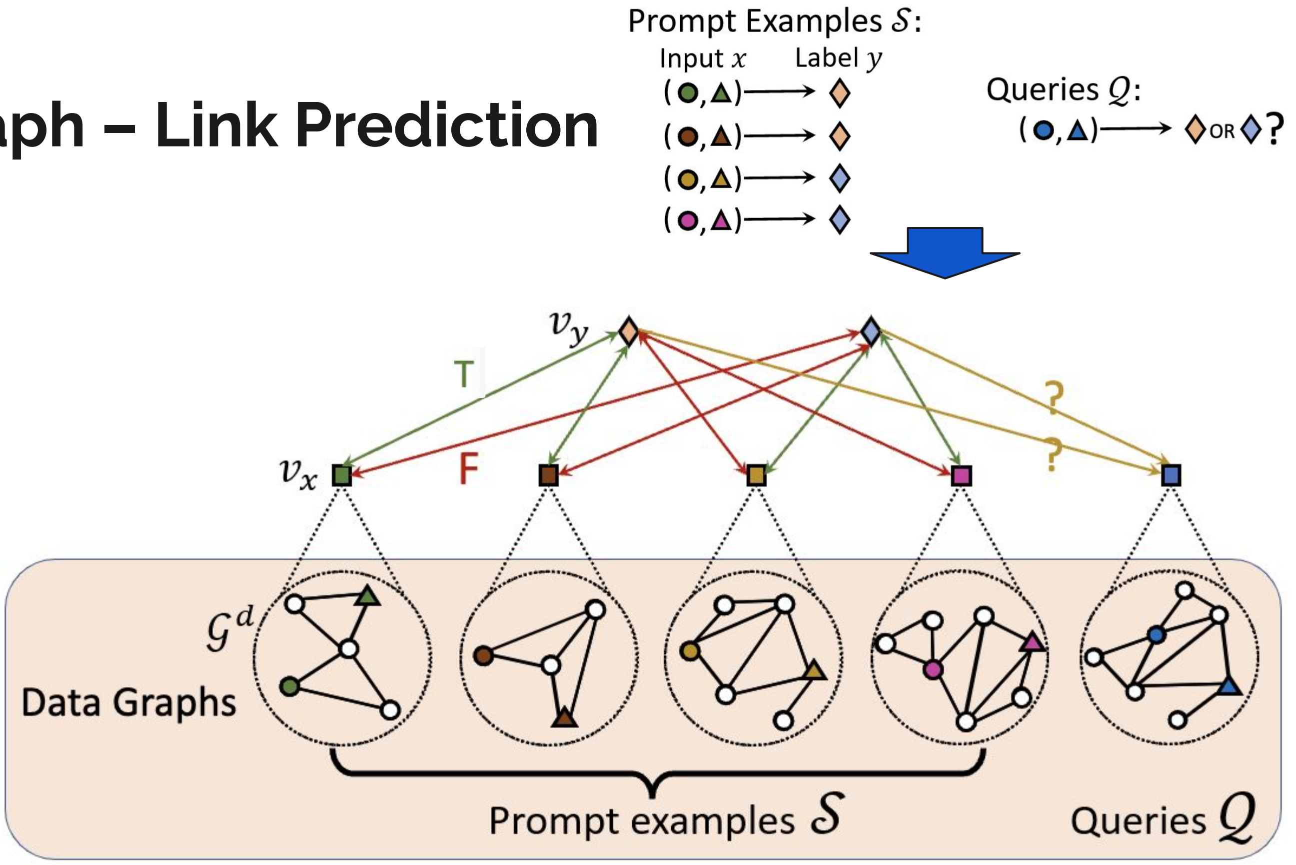
- **Prompt examples**: bidirectional edges between data nodes and all label nodes
- **Queries**: single directed edges from each label to each data node



Step2: Task Graph – Link Prediction

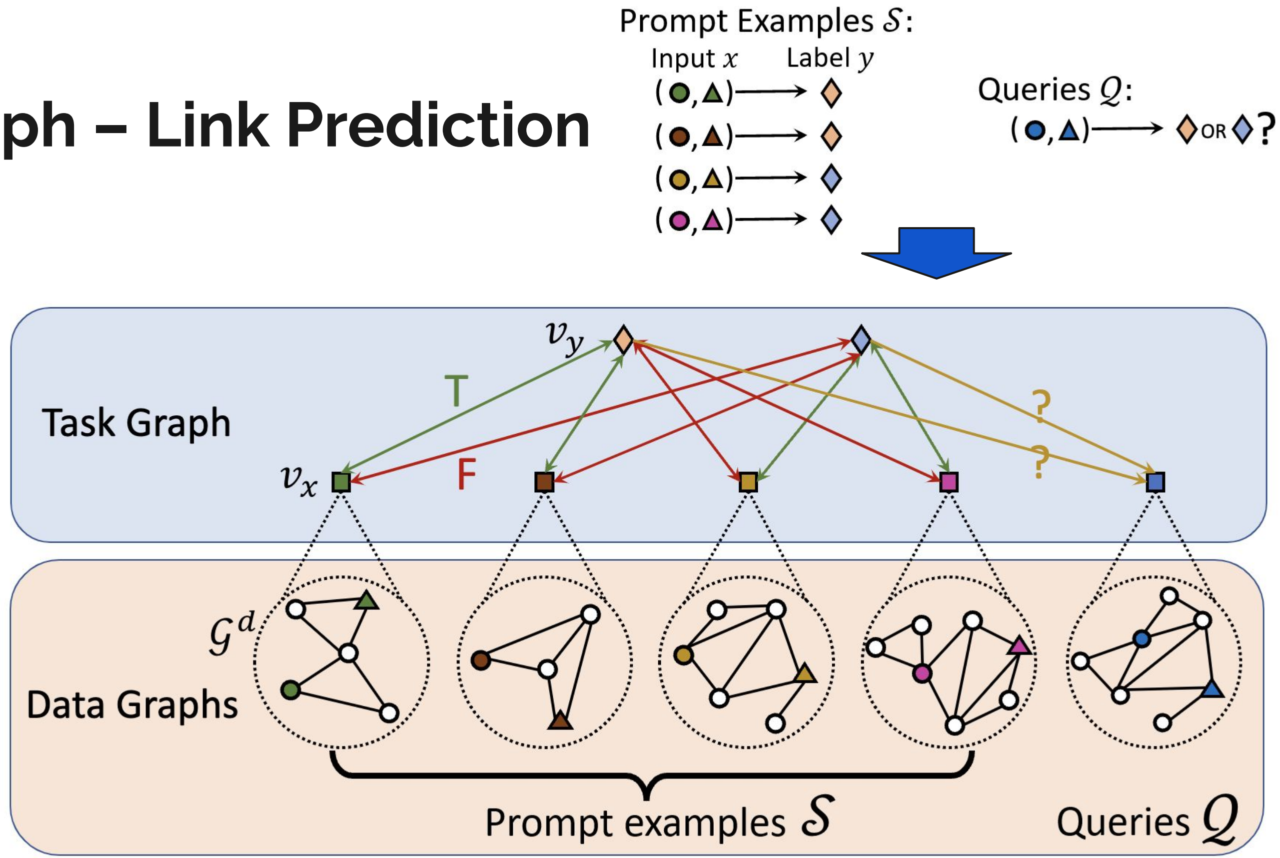
Task Graph interconnects inputs and labels across examples to form context for queries

- **Prompt examples**: bidirectional edges between data nodes and all label nodes
- **Queries**: single directed edges from each label to each data node

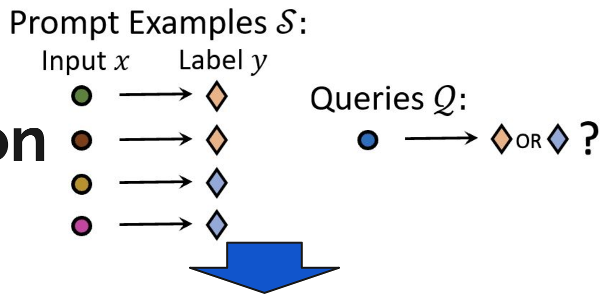


Step2: Task Graph – Link Prediction

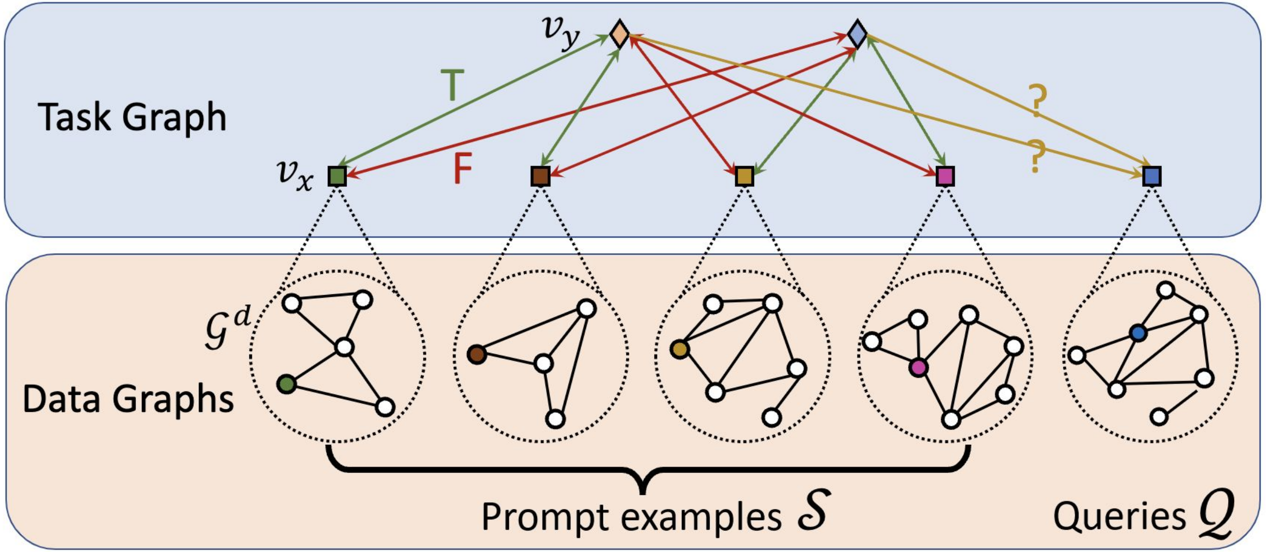
Task Graph interconnects inputs and labels across examples to form context for queries



Step2: Task Graph – Node Classification

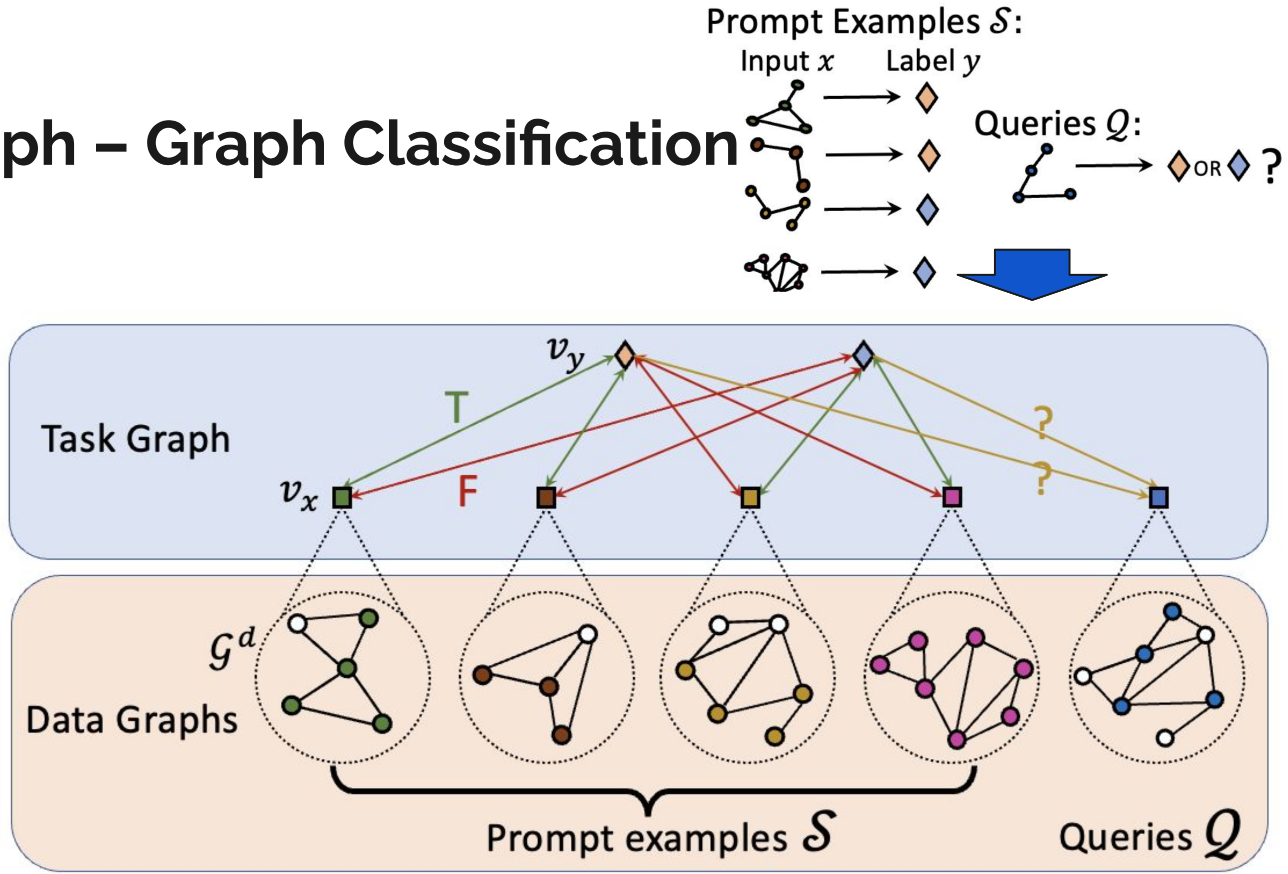


Task Graph interconnects inputs and labels across examples to form context for queries



Step2: Task Graph – Graph Classification

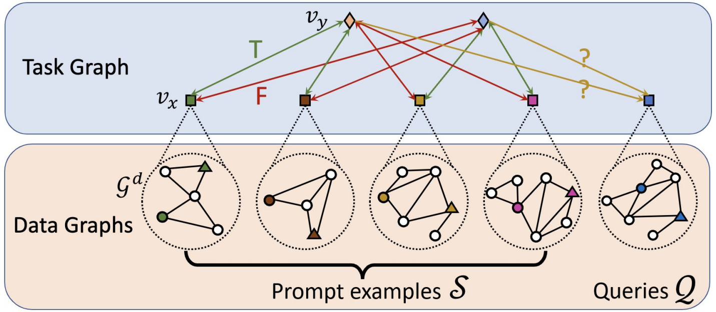
Task Graph interconnects inputs and labels across examples to form context for queries



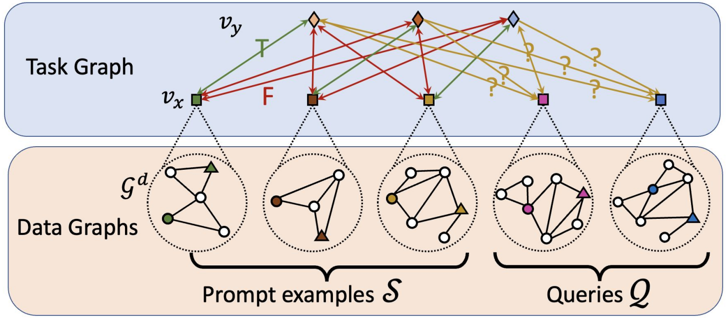
Flexibility of Task Graph

Task Graph unifies classification task format:

- Different number of **classes** are represented as different number of **label nodes**
- Different number of **prompt examples (i.e. shots)** and queries are represented as different number of **data nodes** as well as how they connect with label nodes

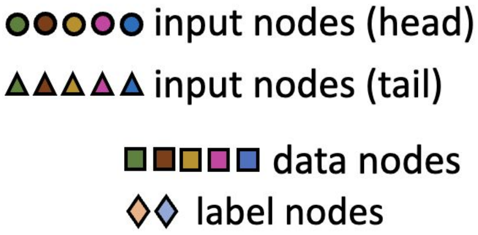


2-shots prompt for 2-class classification



1-shot prompt for 3-class classification

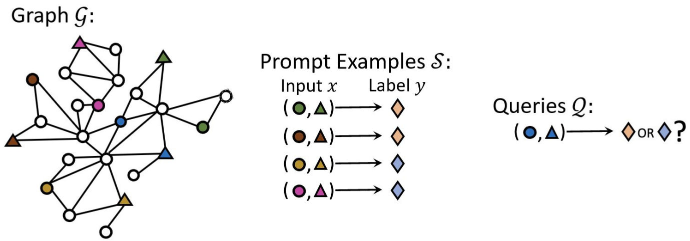
...

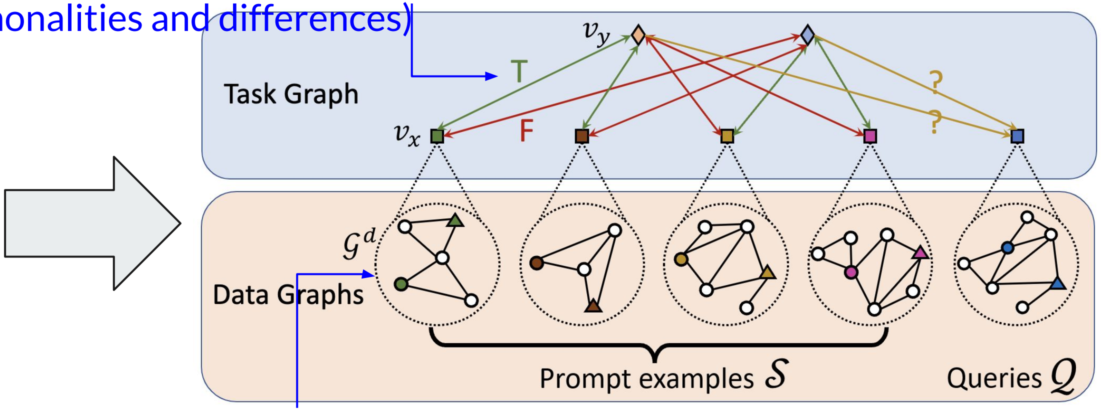


Prompt Graph for in-context learning

How to use **PromptGraph** for in-context learning?

Reflects what is the task using examples (commonalities and differences)





Captures all relevant information about the input

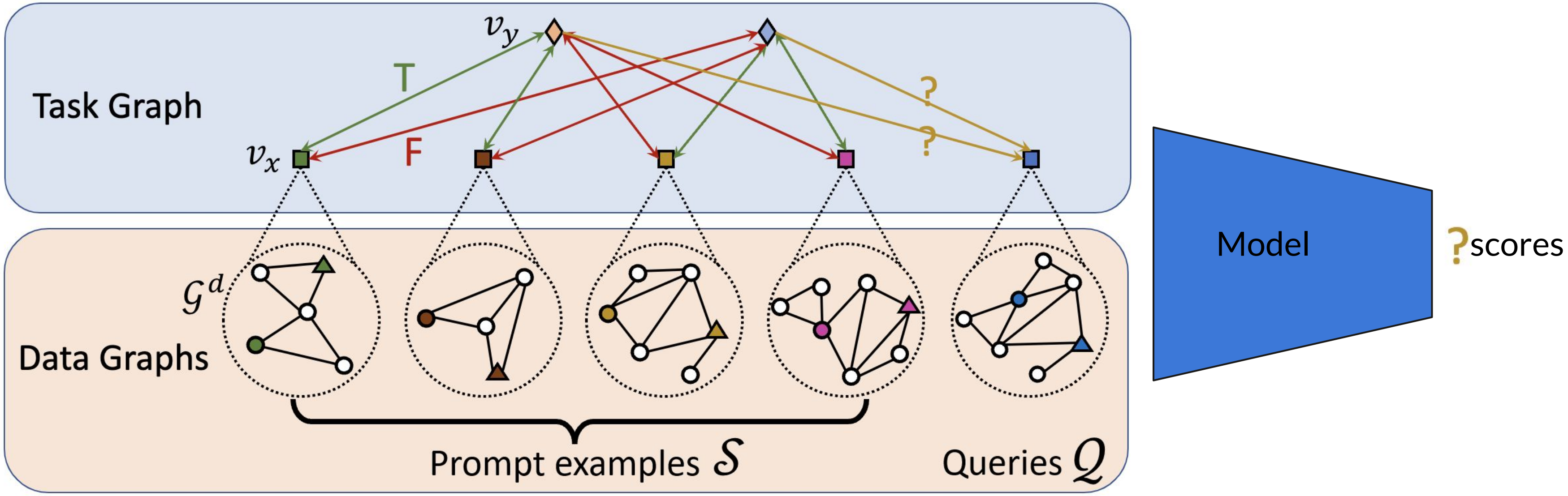
In-context learning over graph

inductive **link prediction** over hierarchical graph

PrompGraph Inference

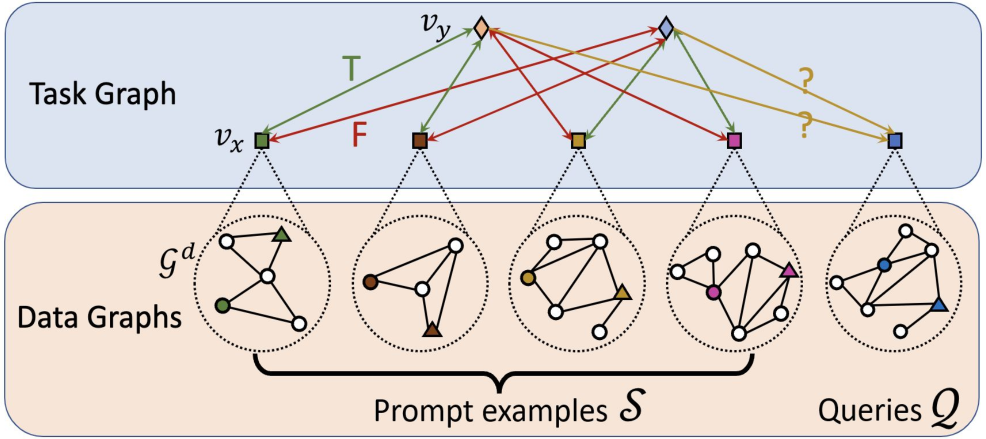
In context prediction GNN

In context prediction GNN



In context prediction GNN

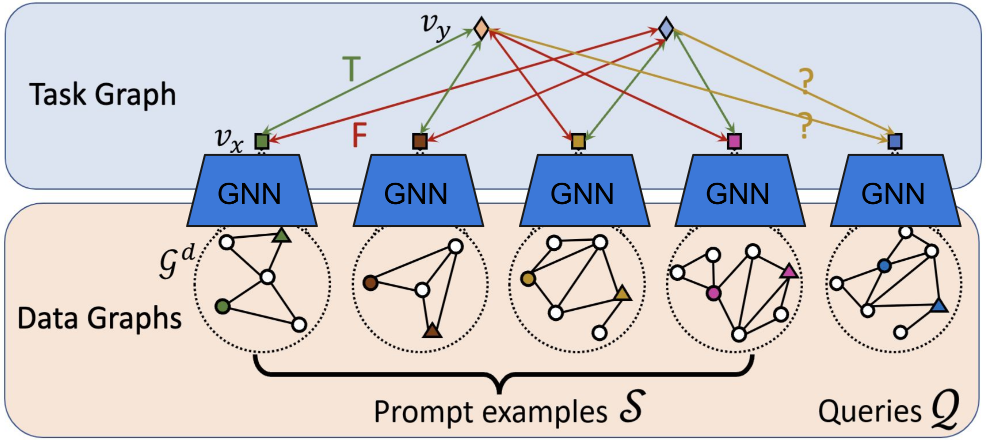
Hierarchical Message passing over PromptGraph



In context prediction GNN

Hierarchical Message passing over PromptGraph

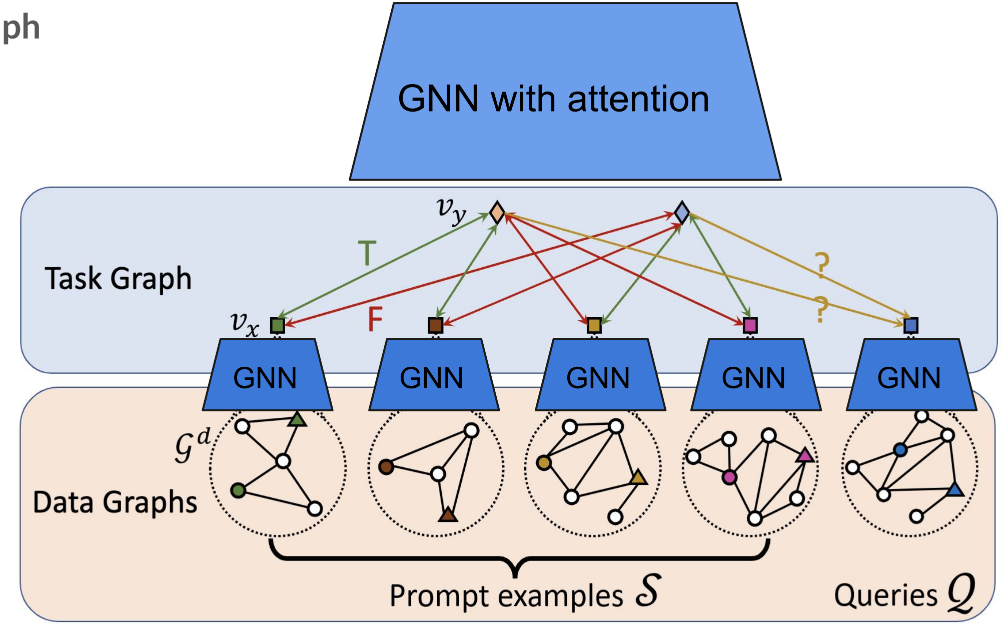
- Data Graph Encoder



In context prediction GNN

Hierarchical Message passing over PromptGraph

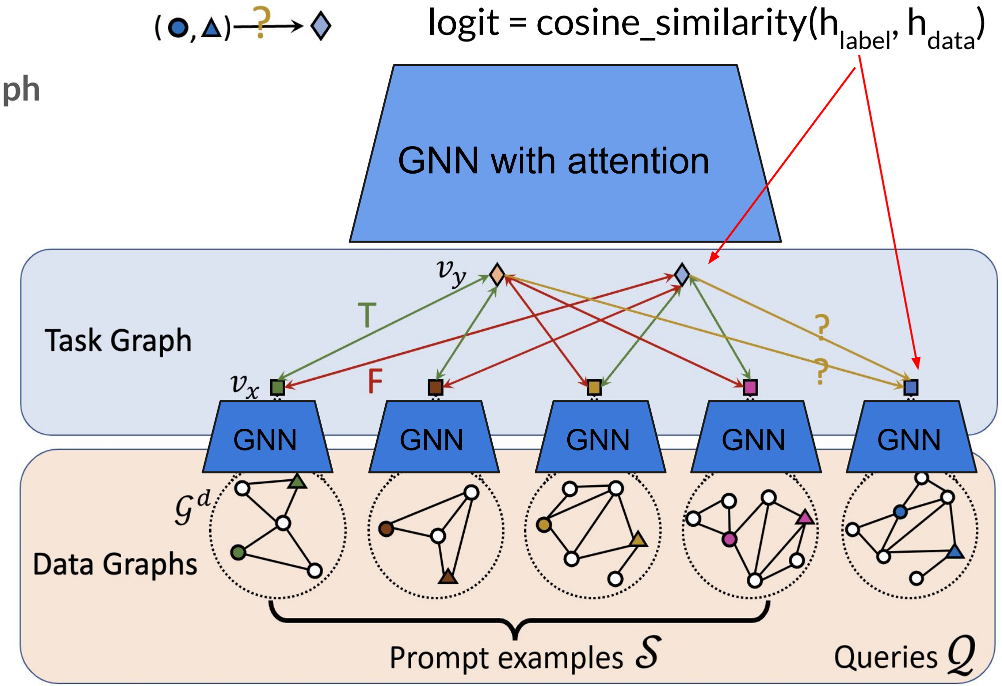
- Data Graph Encoder
- Message Passing over Task Graph



In context prediction GNN

Hierarchical Message passing over PromptGraph

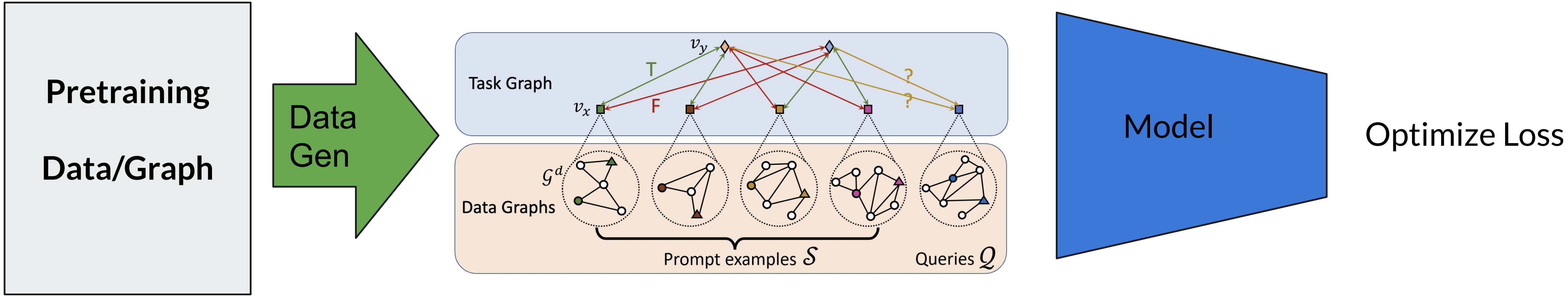
- Data Graph Encoder
- Message Passing over Task Graph
- Compute Logits



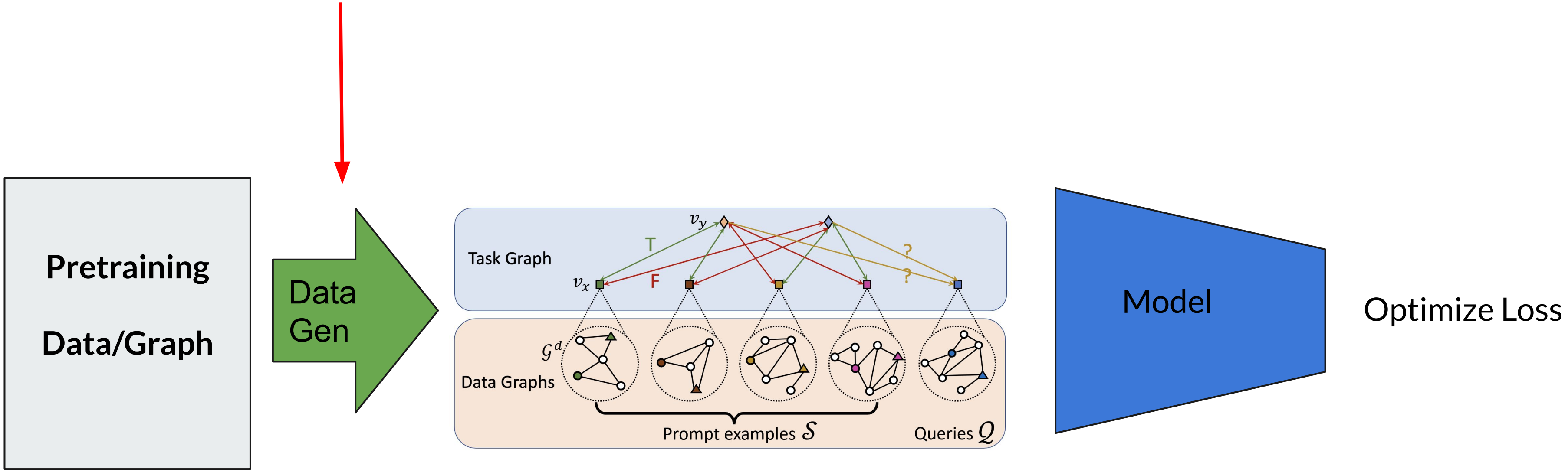
PRODIGY Pretraining

In-context Pretraining

We want to generate pretraining tasks in the format of **PromptGraph** and pretrain our model over them!



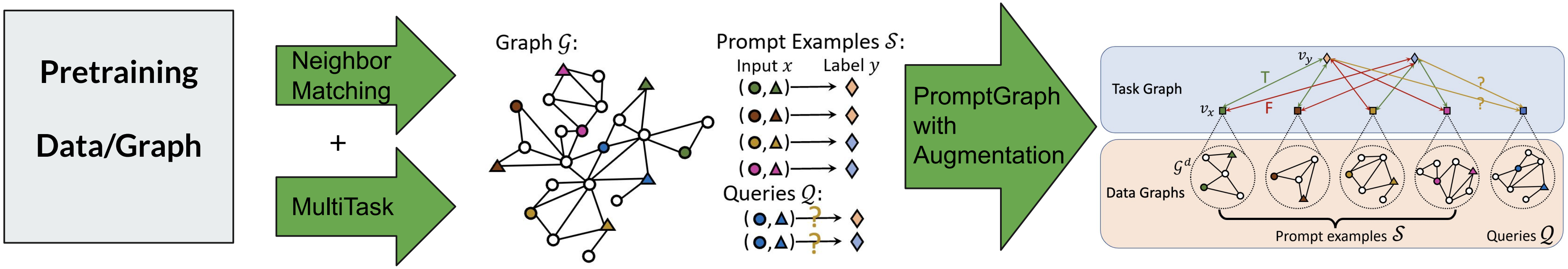
Pretraining Data Generation



Pretraining Data Generation

Two stages:

few-shot prompt generation **convert to PromptGraph**

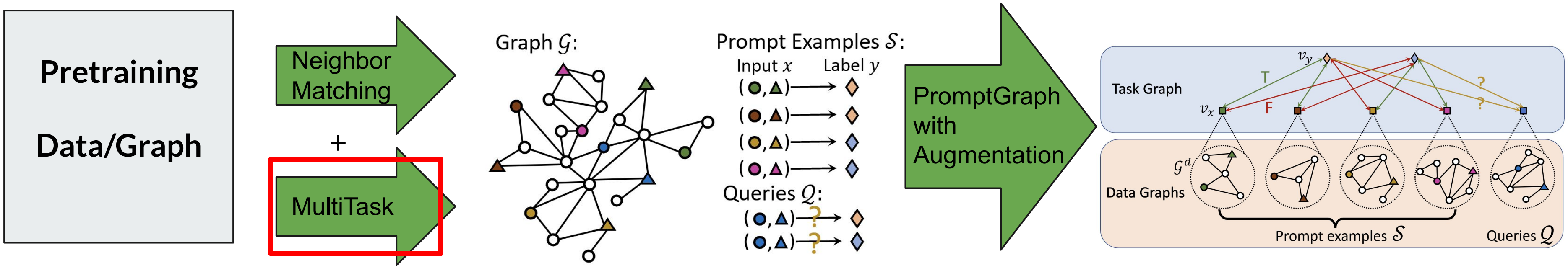


Pretraining Data Generation

Two stages:

few-shot prompt generation

convert to PromptGraph



Simply select multiple tasks for pretraining
-> Supervised pretraining

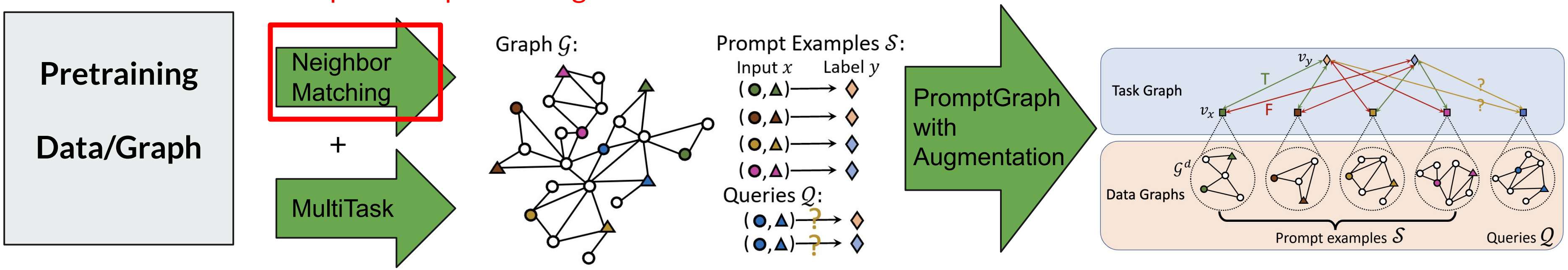
Pretraining Data Generation

Two stages:

few-shot prompt generation

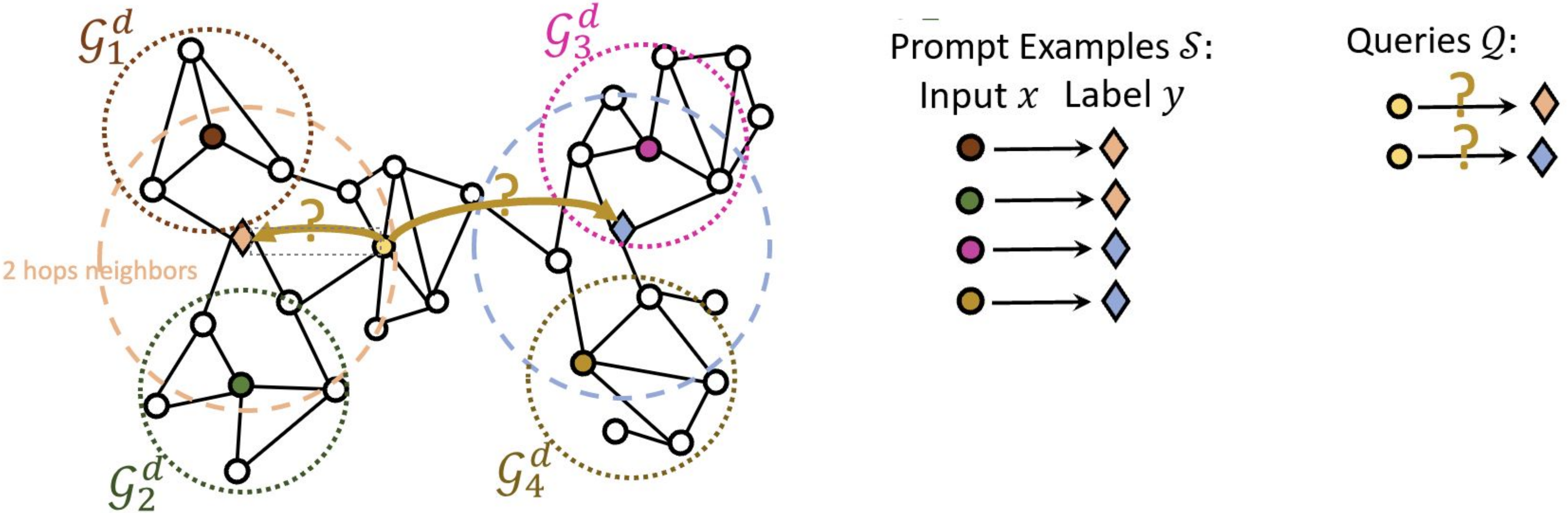
self- supervised pretraining

convert to PromptGraph



Self-supervised Task Example: Neighbor Matching

Idea: the task is to classify which neighborhood a node is in, where each neighborhood is defined by other nodes in it.

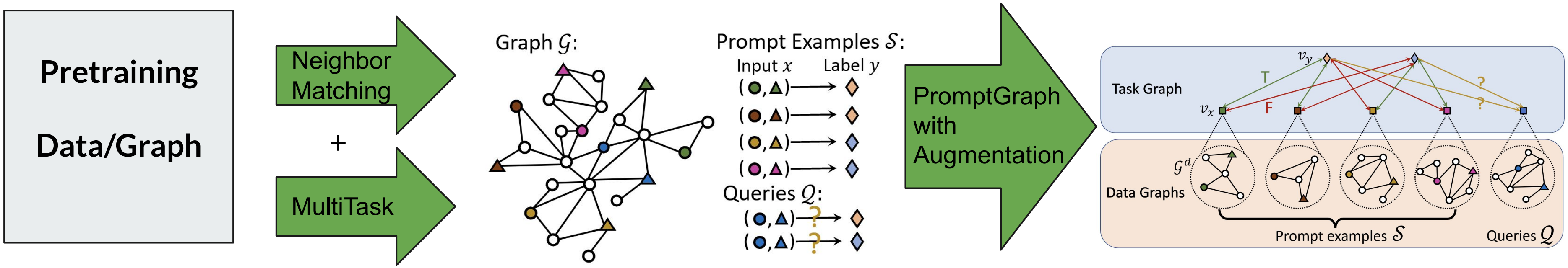


Pretraining Data Generation

Two stages:

few-shot prompt generation

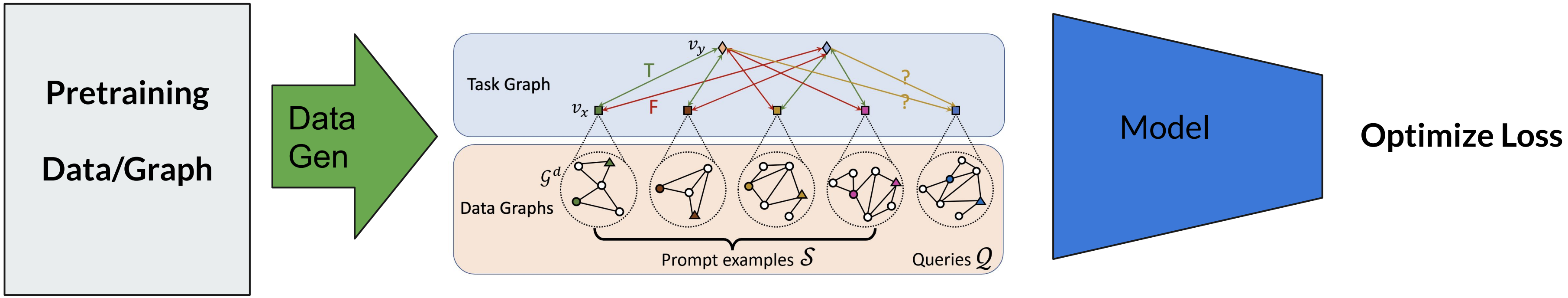
convert to PromptGraph



Mixing different data generation pipeline gives more diverse pretraining data and better results!

In-context Pretraining Objectives

- **Classification loss** over generated tasks
- Attribute prediction loss: reconstruct corrupted features during DataGraph augmentation



Conclusion

PRODIGY enables **in-context learning over graphs** with a novel in-context task representation, Prompt Graph, and corresponding pretraining architecture and objectives.

- **Prompt Graph** provides a unified representation of few-shot prompts over graph for diverse tasks
- **Pretrain** a hierarchical message passing model over Prompt Graph enables in-context learning over unseen tasks on unseen graphs
- **Hard and diverse pretraining tasks** allow the model to keep scaling with more training data

Reliable Graph Learning with Guaranteed Uncertainty Estimates

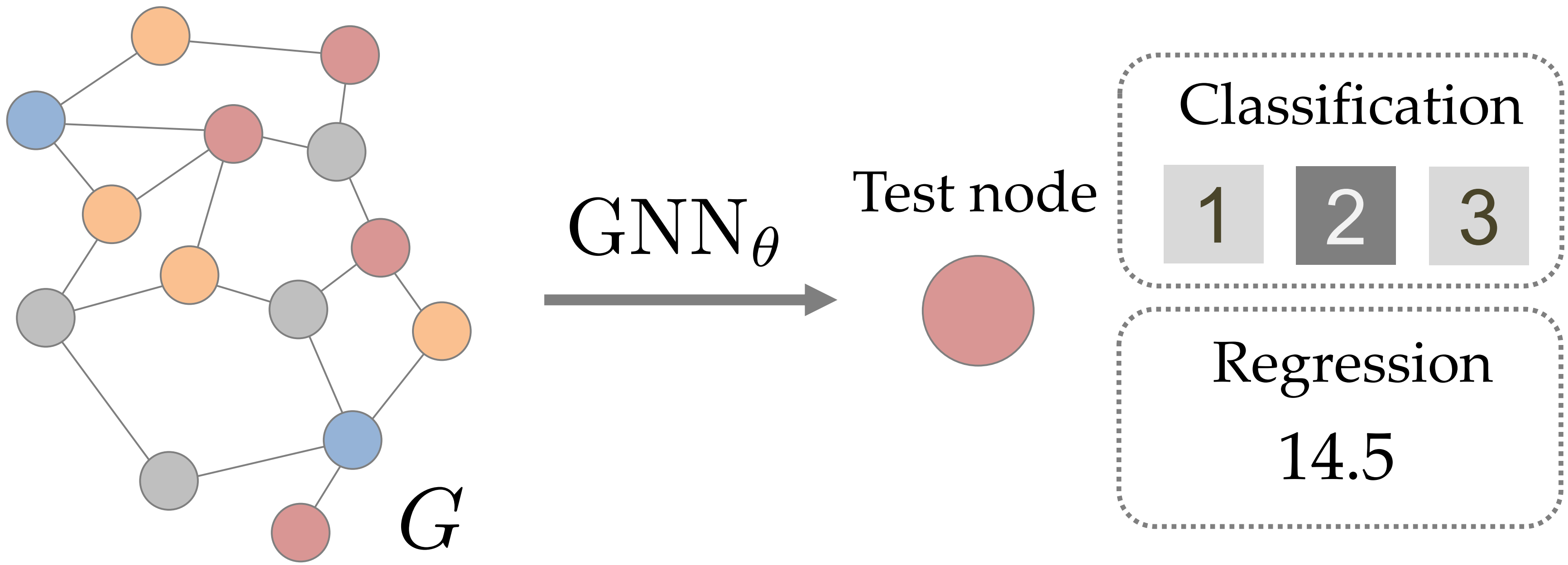
Guest Lecture at Stanford CS 224W: Machine Learning with Graphs

Kexin Huang
PhD student at Stanford CS
@KexinHuang5
kexinhuang.com
kexinh@cs.stanford.edu

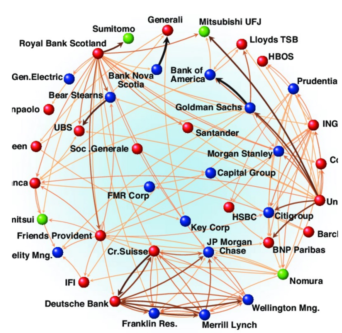
Stanford ARTIFICIAL INTELLIGENCE

SNAP

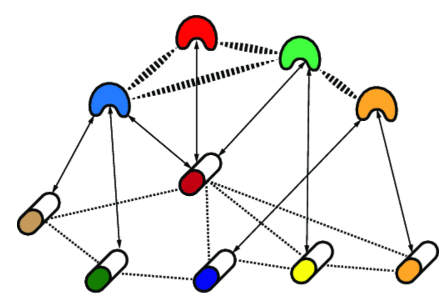
GNNs are powerful



Patient Network



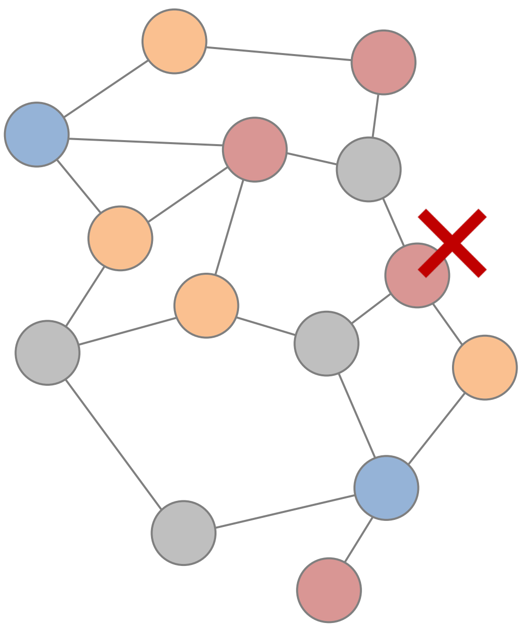
Financial Network



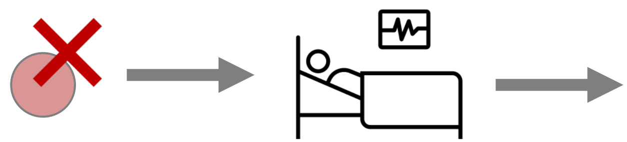
Drug-discovery Network

Errors in critical applications

In high stake settings, errors have huge costs.



GNN on patient network:



Wrong diagnosis

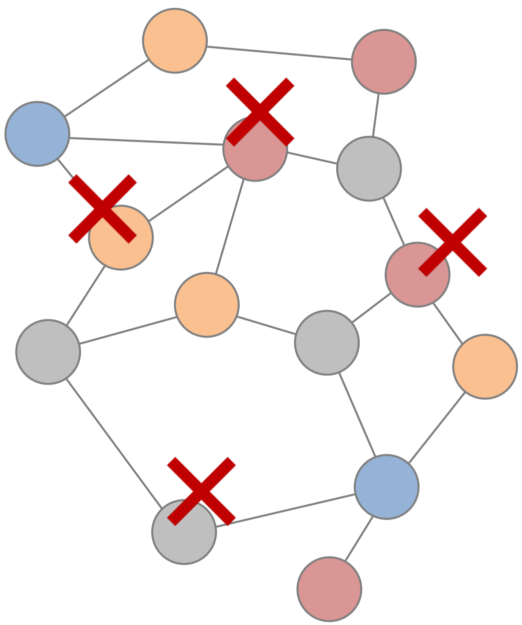
GNN on drug discovery network:

Wasted experiment

GNN on financial network :

Undetected Fraud

As the errors accumulate…



People stop trusting GNNs!

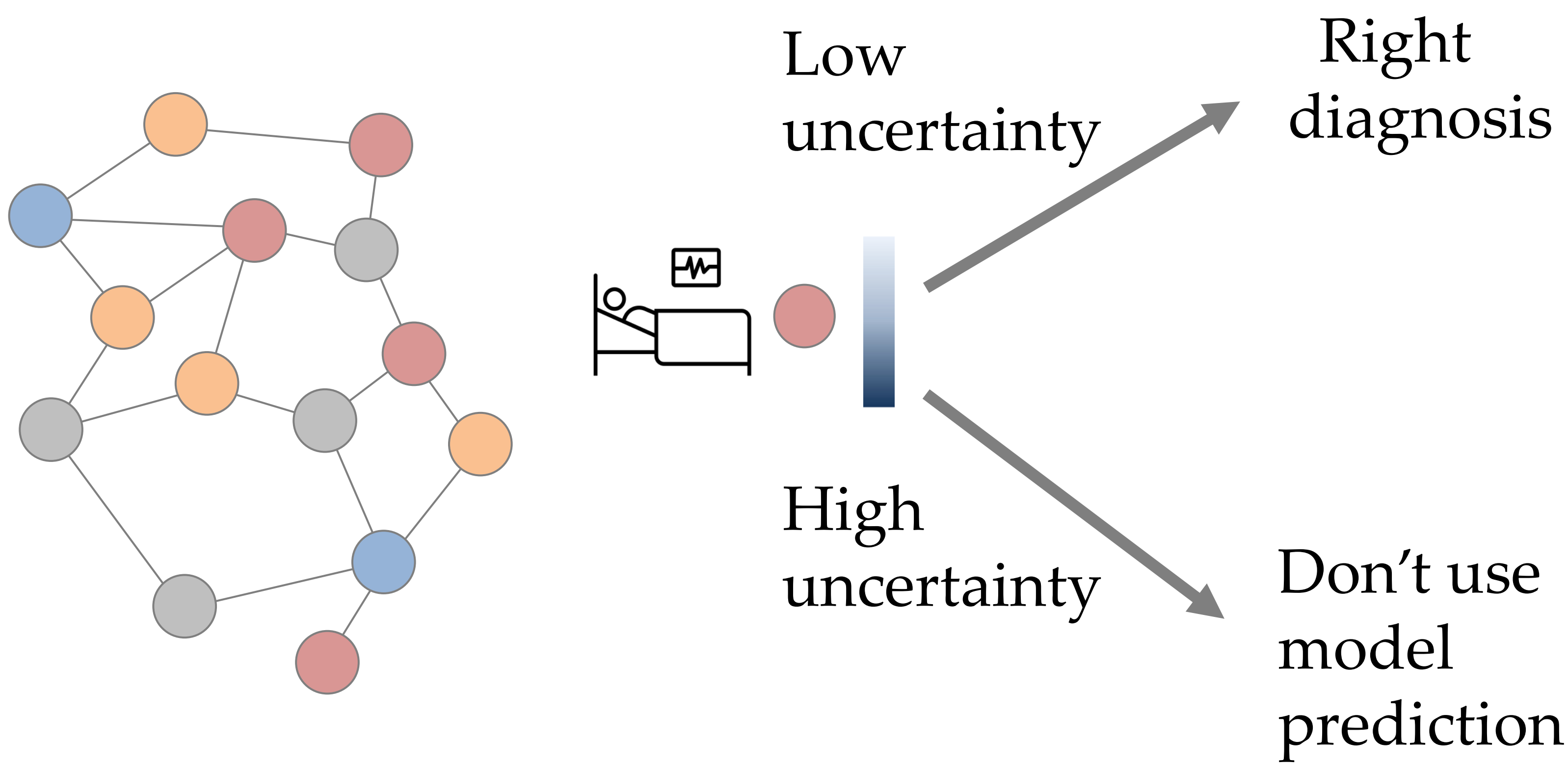
What should we do?

Can we know when will the model break down?

In another word, can we produce measure of uncertainty for model prediction?

If we know when the model will break down, …

GNN on patient graph:



How to produce a good uncertainty estimation?

After this lecture, you will learn…

- How to evaluate if an uncertainty estimation method is good?
  - What is reliability mathematically?
- How to produce uncertainty estimates with reliability guarantees?
  - Introduction to conformal prediction
- How to produce reliable uncertainty estimates for graphs?
  - State-of-the-art: conformalized GNNs

Plan for Today

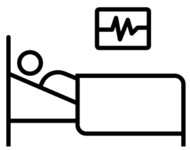
- **How to evaluate if an uncertainty estimation method is good?**
- How to produce uncertainty estimates with reliability guarantees?
- How to produce reliable uncertainty estimates for graphs?

When will the model break down? quantifying uncertainty

	Point prediction $\hat{Y}_{n+1}$	→	A range of plausible predictions $C(X_{n+1})$
Test node $X_{n+1} \in \mathcal{D}_{\text{test}}$	Classification: 1 **2** 3	→	Prediction set $\{2, 1\}$
	Regression: 14.5	→	Prediction interval $[14.1, 14.8]$

Key benefits of prediction set/interval

A range of plausible predictions



$$C(X_{n+1})$$

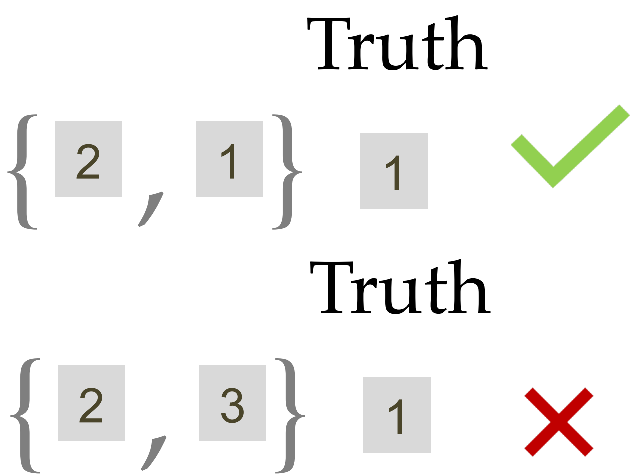
{Disease A, Disease B, Disease C}

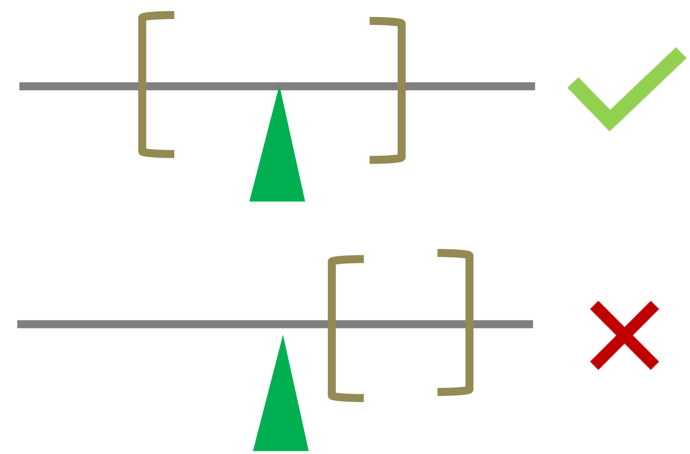
- Offers a meaningful range of values for more informed decision-making.
- The size of set/interval measures level of uncertainty.
- Size is large - the model will break down
- Enable a rigorous notion of reliability

How to define if a prediction set/interval is good? - **Coverage**

$$\text{Coverage} := \frac{1}{|\mathcal{D}_{\text{test}}|} \sum_{i \in \mathcal{D}_{\text{test}}} \mathbb{I}(Y_i \in C(X_i))$$

i.e. % of testing points X_i where ground truth Y_i falls into the prediction set/interval C



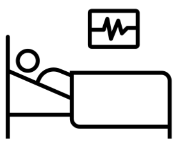


Can we control coverage?

$$\text{Coverage} := \frac{1}{|\mathcal{D}_{\text{test}}|} \sum_{i \in \mathcal{D}_{\text{test}}} \mathbb{I}(Y_i \in C(X_i))$$

- A truly trustworthy ML model have 100% coverage.
- But prediction set/interval is predicted from the model, so 100% is not possible.
- Is it possible to control it with guarantees?

What if we can control the coverage with guarantees?



99% of predicted diagnosis set **provably** includes ground truth

Given a pre-defined target coverage level $1 - \alpha$, we reach this coverage level with guarantees.

Reliability & trust = coverage guarantees

Achieving coverage guarantee is hard

UQ Model	Cora	DBLP	CiteSeer	PubMed	Computers	Covered?
Temp. Scale.	0.946±.003 ✗	0.920±.009 ✗	0.952±.004 ✓	0.899±.002 ✗	0.929±.002 ✗	✗
Vector Scale.	0.944±.004 ✗	0.921±.009 ✗	0.951±.004 ✓	0.899±.003 ✗	0.932±.002 ✗	✗
Ensemble TS	0.947±.003 ✗	0.920±.008 ✗	0.953±.003 ✓	0.899±.002 ✗	0.930±.002 ✗	✗
CaGCN	0.939±.005 ✗	0.922±.004 ✗	0.949±.005 ✗	0.898±.003 ✗	0.926±.003 ✗	✗
GATS	0.939±.005 ✗	0.921±.004 ✗	0.951±.005 ✓	0.898±.002 ✗	0.925±.002 ✗	✗

UQ Model	Anaheim	Chicago	Education	Election	Twitch	Covered?
QR	0.943±.031 ✗	0.950±.007 ✗	0.959±.001 ✓	0.956±.004 ✓	0.900±.015 ✗	✗
MC dropout	0.553±.022 ✗	0.427±.015 ✗	0.423±.013 ✗	0.417±.008 ✗	0.448±.017 ✗	✗
BayesianNN	0.967±.001 ✓	0.955±.003 ✓	0.957±.002 ✓	0.958±.009 ✓	0.923±.006 ✗	✗

Previous methods produce heuristic uncertainties but have no guarantees!

Plan for Today

- How to measure if an uncertainty estimation method is good? ✓
- **How to produce uncertainty estimates with reliability guarantees?**
- How to produce reliable uncertainty estimates for graphs?

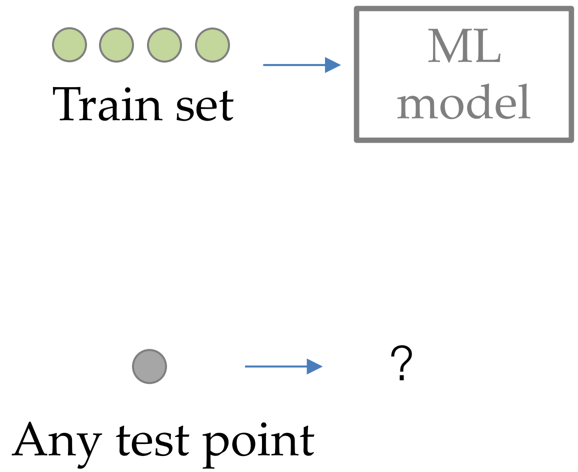
The promises of conformal prediction

- Theoretical guarantees on finite sample coverage validity
- Distribution-free
- Model-agnostic
- Post-hoc wrapper: no training modifications

Input of the data

- Training (+val)
- Calibration
- Testing

Pre-defined
coverage level
$$1 - \alpha$$

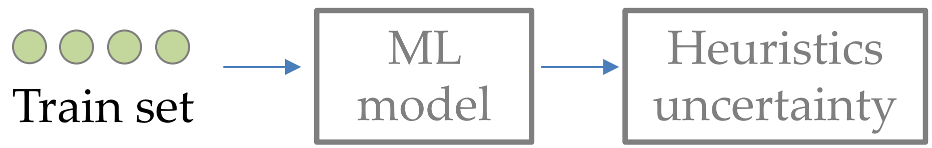


Step 1/4: define heuristics uncertainty

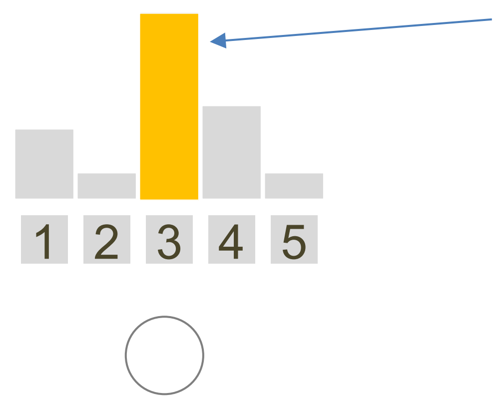
- Training (+val)
- Calibration
- Testing

Pre-defined coverage level

$$1-\alpha$$



Classification: softmax scores



$$\hat{\mu}(x)_{(3)}$$

Model "confidence" about this instance having label 3

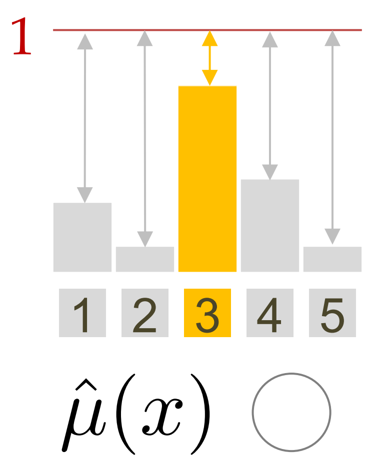
$$\hat{\mu}(x)$$

Step 2/4: non-conformity scores

$V : (\mathcal{X} \times \mathcal{Y}) \to \mathbb{R}$ Non-conformity score function

$V(X, Y)$ How much label Y "conforms" to the prediction at X
Larger *V*: higher non-conformity, lower confidence
Smaller *V*: lower non-conformity, more confidence

Let's construct one for softmax score:



$$V(X = x, Y = 3) = 1 - \hat{\mu}(x)_{(3)}$$

Low non-conformity score when softmax is high i.e. model is confident

Step 3/4: quantile computation

- Training (+val)
- Calibration
- Testing

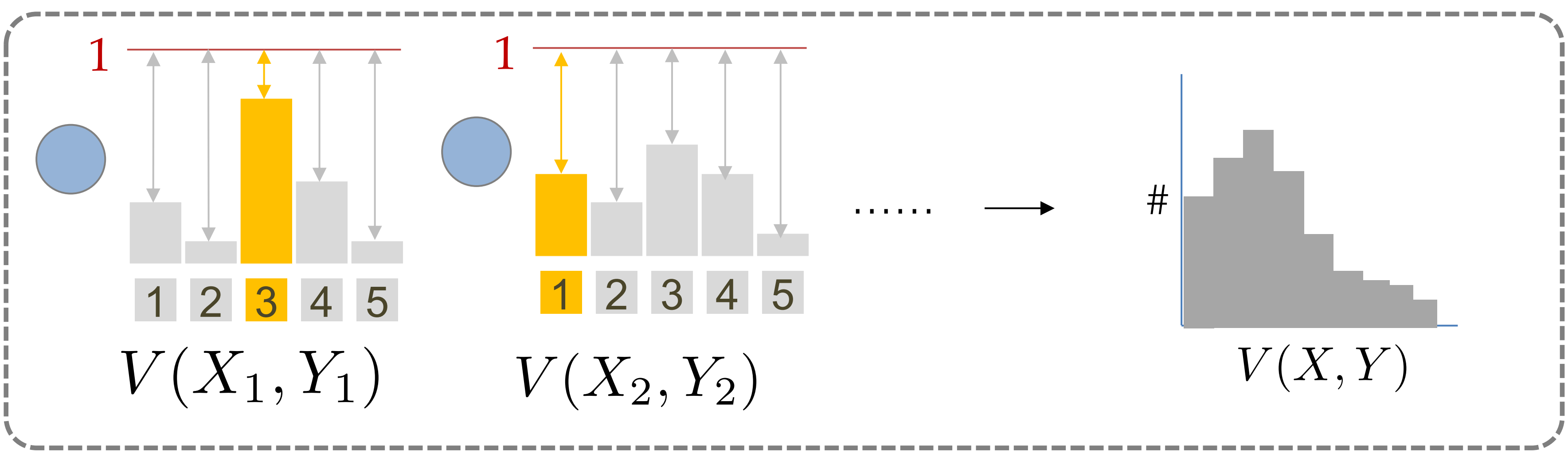
Pre-defined coverage level

$$1-\alpha$$

$V(X_1, Y_1)$ $V(X_2, Y_2)$ ····· $V(X_n, Y_n)$

$$\{V(X_i, Y_i)\}_{i=1}^{n}$$

First calculate non-conformity score for each calibration data point



Step 3/4: quantile computation

- Training (+val)
- Calibration
- Testing

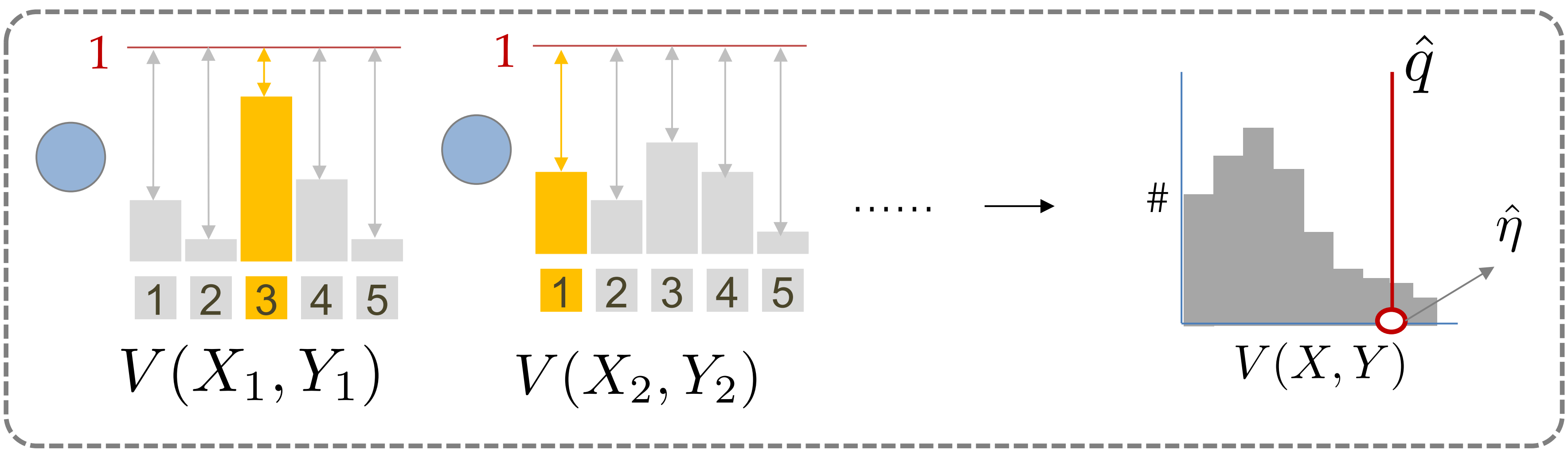
Pre-defined coverage level

$1-\alpha$

$V(X_1,Y_1)$ $V(X_2,Y_2)$ ..... $V(X_n,Y_n)$

$$\hat{\eta} = \text{quantile}(\{V(X_i,Y_i)\}_{i=1}^{n}, \underbrace{(1-\alpha)(1+\tfrac{1}{n})}_{\hat{q}})$$

Takes the quantile of all non-conformity scores (with small correction)



Step 3/4: quantile computation

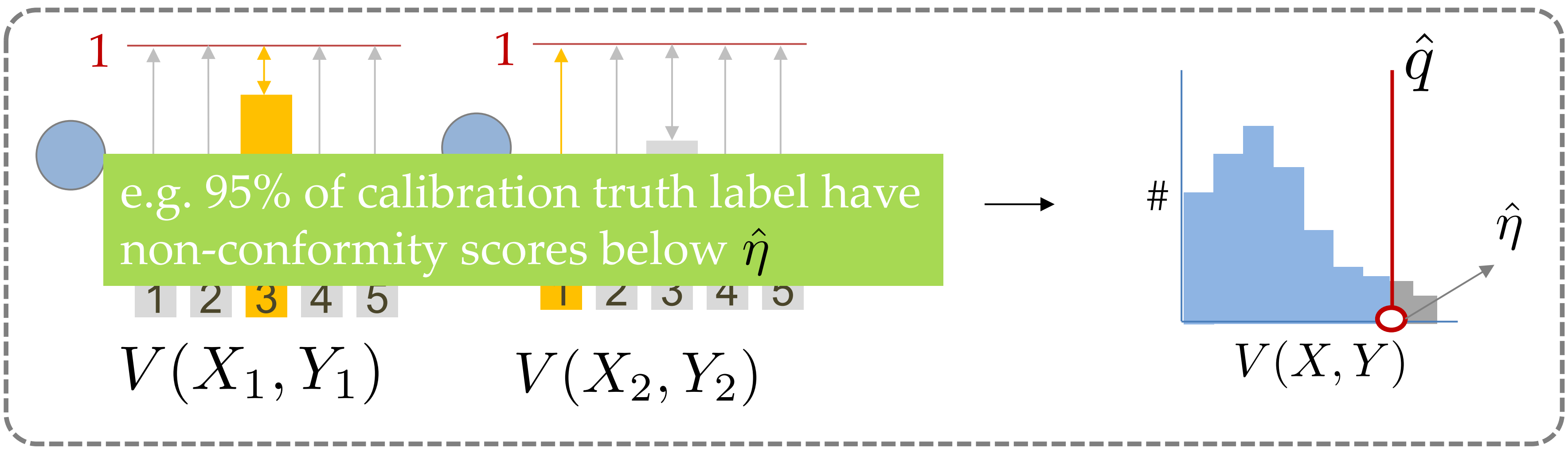
- Training (+val)
- Calibration
- Testing

Pre-defined coverage level
$1-\alpha$

$V(X_1, Y_1)$ $V(X_2, Y_2)$ ····· $V(X_n, Y_n)$

$$\hat{\eta} = \text{quantile}(\{V(X_i, Y_i)\}_{i=1}^{n}, \underbrace{(1-\alpha)(1+\tfrac{1}{n})}_{\hat{q}})$$

Takes the quantile of all non-conformity scores (with small correction)

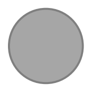


Step 4/4: prediction set/interval construction

- Training (+val)
- Calibration
- Testing

Pre-defined coverage level

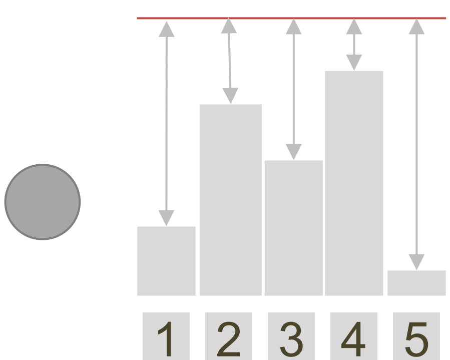
$1-\alpha$



X_{n+1}

$$C(X_{n+1}) = \{y \in \mathcal{Y} : V(X_{n+1}, y) \leq \hat{\eta}\}$$

The set of labels where non-conformity scores are smaller than $\hat{\eta}$



$$V(X_{n+1}, Y = 1)$$
$$V(X_{n+1}, Y = 2)$$
$$V(X_{n+1}, Y = 3)$$
$$V(X_{n+1}, Y = 4)$$
$$V(X_{n+1}, Y = 5)$$

Step 4/4: prediction set/interval construction

- Training (+val)
- Calibration
- Testing

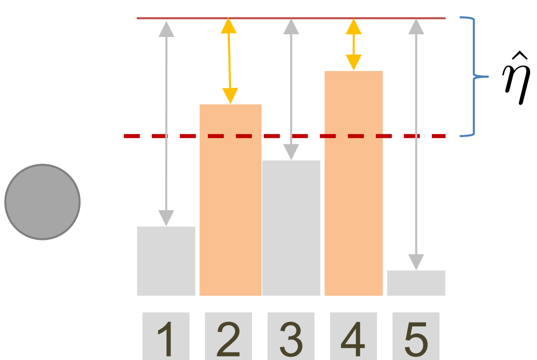
Pre-defined coverage level

$1 - \alpha$

X_{n+1}

$$C(X_{n+1}) = \{y \in \mathcal{Y} : V(X_{n+1}, y) \leq \hat{\eta}\}$$

The set of labels where non-conformity scores are smaller than $\hat{\eta}$



$$V(X_{n+1}, Y = 1) > \hat{\eta} \quad ✗$$
$$V(X_{n+1}, Y = 2) < \hat{\eta} \quad ✓$$
$$V(X_{n+1}, Y = 3) > \hat{\eta} \quad ✗ \longrightarrow \{2, 4\}$$
$$V(X_{n+1}, Y = 4) < \hat{\eta} \quad ✓$$
$$V(X_{n+1}, Y = 5) > \hat{\eta} \quad ✗$$

In Summary

- Step 1/4: define heuristics uncertainty
  - e.g. softmax class probabilities $\hat{\mu}(x)$
- Step 2/4: compute non-conformity scores for calibration sets
  - e.g. 1-softmax scores
- Step 3/4: compute quantile $\hat{\eta}$ of non-conformity scores
- Step 4/4: Construct prediction set
  - The set of labels below $\hat{\eta}$

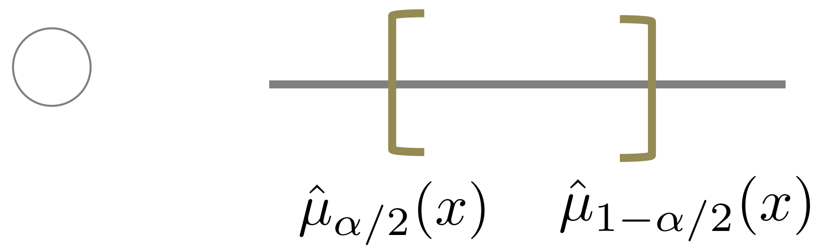
Similarly, for regression

- Training (+val)
- Calibration
- Testing

Pre-defined coverage level

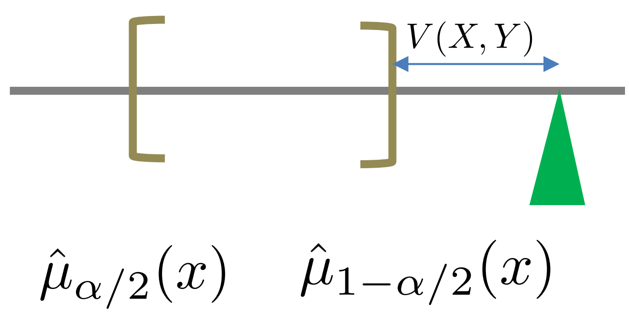
$1 - \alpha$

Step 1: heuristic uncertainty



Step 2: non-conformity score

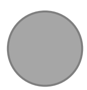
$$V(X=x, Y=y) = \max(\hat{\mu}_{\alpha/2}(x) - y, y - \hat{\mu}_{1-\alpha/2}(x))$$



Conformalized Quantile Regression

Step 3: computes the quantile over calibration data (standard and not shown)

Step 4: prediction interval construction



$$C(X_{n+1}) = \{y \in \mathcal{Y} : V(X_{n+1}, y) \leq \hat{\eta}\}$$

X_{n+1}

$$C(X_{n+1}) = [\hat{\mu}_{\alpha/2}(X_{n+1}) - \hat{\eta}, \hat{\mu}_{1-\alpha/2}(X_{n+1}) + \hat{\eta}]$$

Coverage guarantees

Theorem 1 [Vovk, Gammerman, and Saunders, 1999]

Given exchangeability between calibration set $\{(X_i, Y_i)\}_{i=1}^{n}$ and (X_{n+1}, Y_{n+1})

$$P\{Y_{n+1} \in C(X_{n+1})\} \geq 1 - \alpha$$

The probability of the ground truth falls into the prediction set is $1 - \alpha$

What is exchangeability?

Assumption: Exchangeability

Exchangeability between calibration set and test point

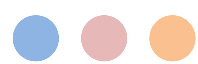
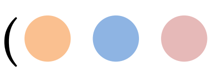
$$\underbrace{(X_1, Y_1), \cdots, (X_n, Y_n)}_{\text{calibration set}}, \underbrace{(X_{n+1}, Y_{n+1})}_{\text{Any test point}}$$

Denote $Z_i = (X_i, Y_i)$

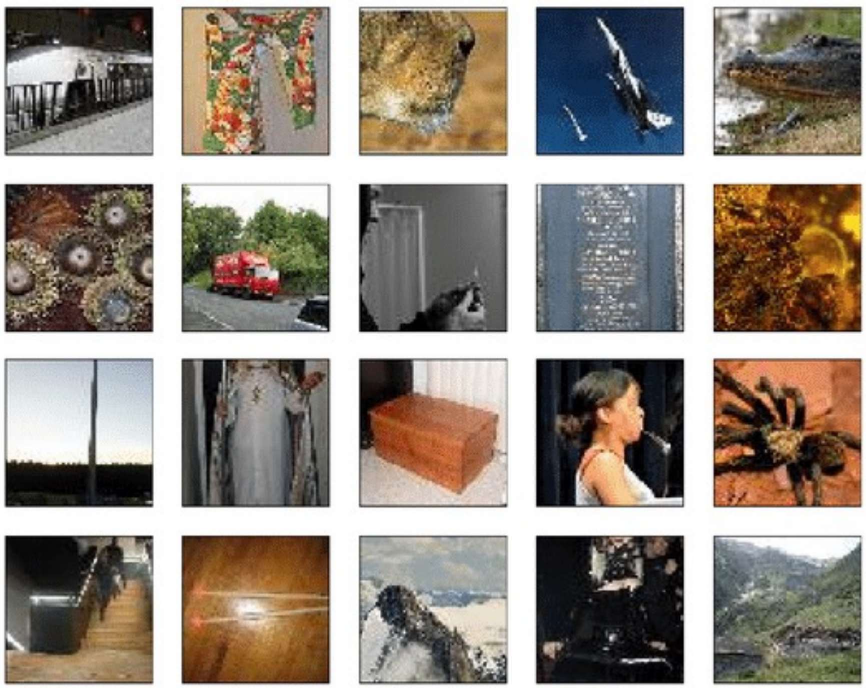
Given any permutation π of $\{1, \cdots, n+1\}$, it holds that

$$P((Z_{\pi(1)}, \ldots, Z_{\pi(n+1)}) = (z_1, \ldots, z_{n+1})) = P((Z_1, \ldots, Z_{n+1}) = (z_1, \ldots, z_{n+1}))$$

Draw 3 colored balls from a bag,

P() = P ()

Conformal Prediction have been applied to vision, NLP, etc.



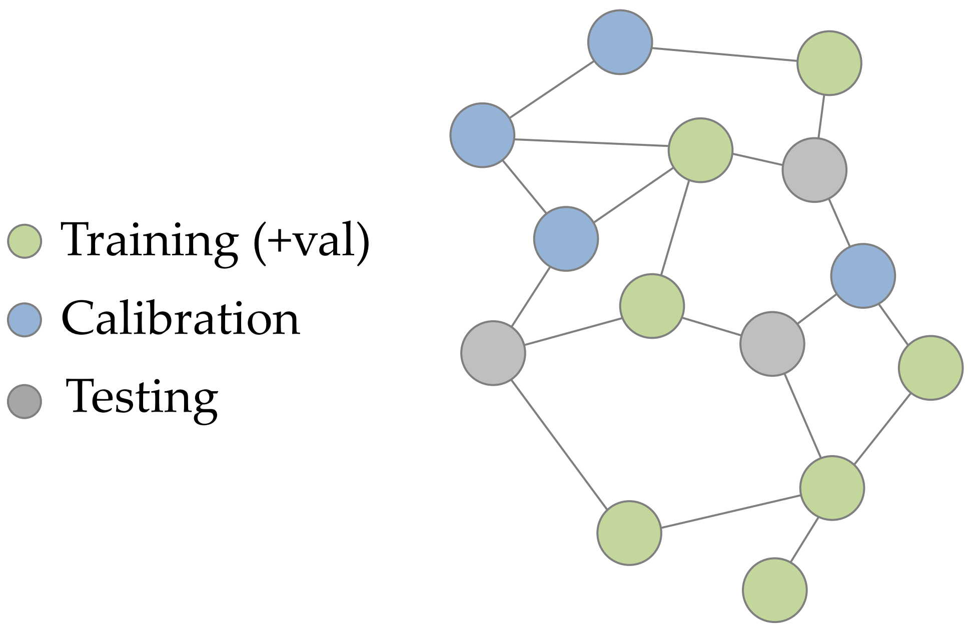
In computer vision, NLP problems, the **calibration** and **test points** are **i.i.d** (independent, and identically distributed).

Note: i.i.d implies exchangeability whereas the reverse is not true.

Plan for Today

- How to measure if an uncertainty estimation method is good? ✓
- How to produce uncertainty estimates with reliability guarantees? ✓
- **How to produce reliable uncertainty estimates for graphs?**

However, does exchangeability hold for graph structured data?



- Dependencies between testing and calibration nodes. (i.e., not i.i.d).
- Message-passing during training includes calibration and test nodes.

Previously, $V(X, Y; \{Z_v\}_{v \in \mathcal{D}_{\text{train}} \cup \mathcal{D}_{\text{valid}}})$

Now,

$$V(X, Y; \{Z_v\}_{v \in \mathcal{D}_{\text{train}} \cup \mathcal{D}_{\text{valid}}}, \underbrace{\{X_v\}_{v \in \mathcal{D}_{\text{calib}} \cup \mathcal{D}_{\text{test}}}}_{\text{Calib \& test seen in training}}, \overbrace{\mathcal{V}, \mathcal{E}}^{\text{Graph dependencies}})$$

Yes!

Theorem 1: in transductive node-level prediction problem under random data split, graph exchangeability holds given permutation invariance.

Most popular GNNs are permutation invariant! → Validity of coverage guarantees for GNNs.

Kexin Huang, Ying Jin, Emmanuel Candès, Jure Leskovec. Uncertainty Quantification over Graph with Conformalized Graph Neural Network. **NeurIPS 2023, spotlight.**

Okay, we have coverage, is that enough?

- No! An arbitrarily large prediction set ensures coverage validity but is practically useless

If there are 5 classes:

{1, 2, 3, 4, 5}

100% coverage!

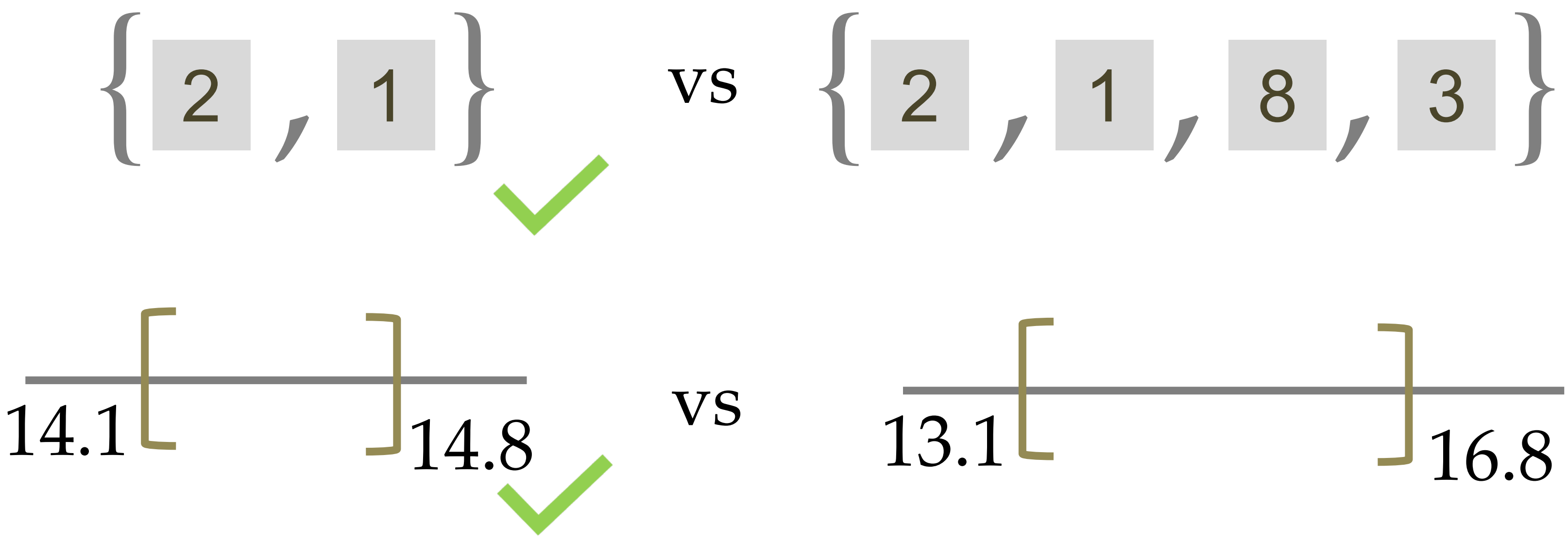
100% coverage!

Both are not usable!

How to define if a prediction set/interval is good? - **Efficiency**

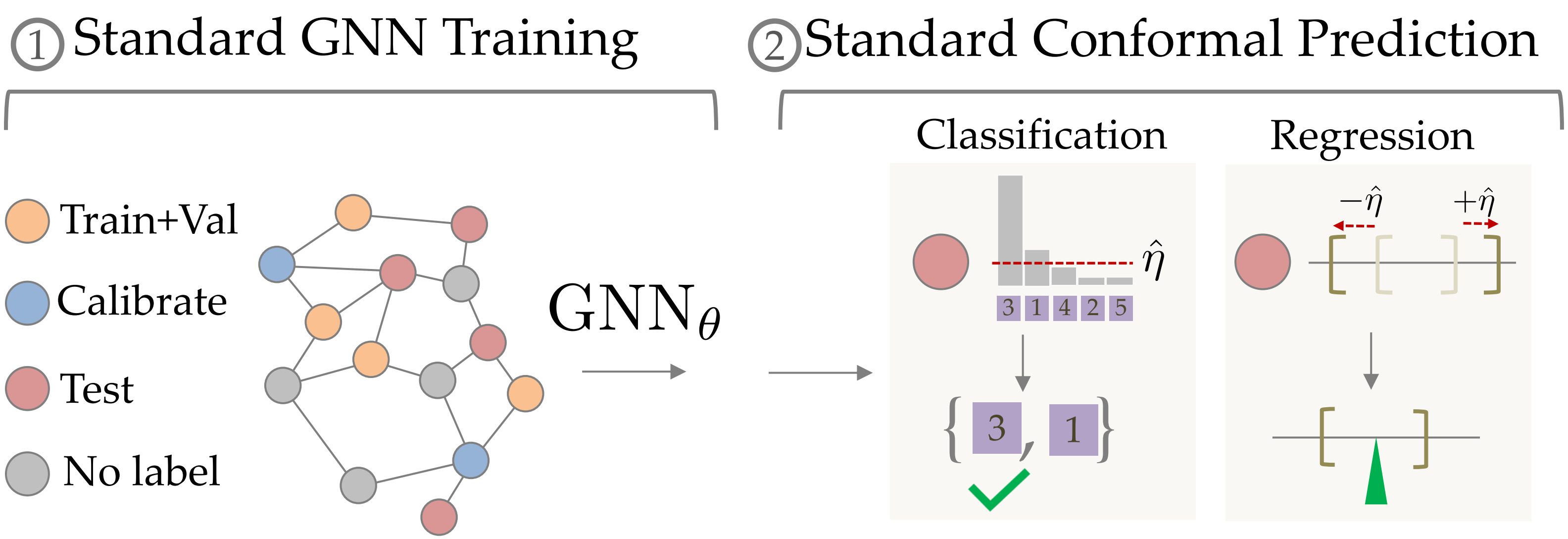
- Inefficiency calculates the length of the prediction set size

$$\text{Inefficiency} := \frac{1}{|\mathcal{D}_{\text{test}}|} \sum_{i \in \mathcal{D}_{\text{test}}} |C(X_i)|$$



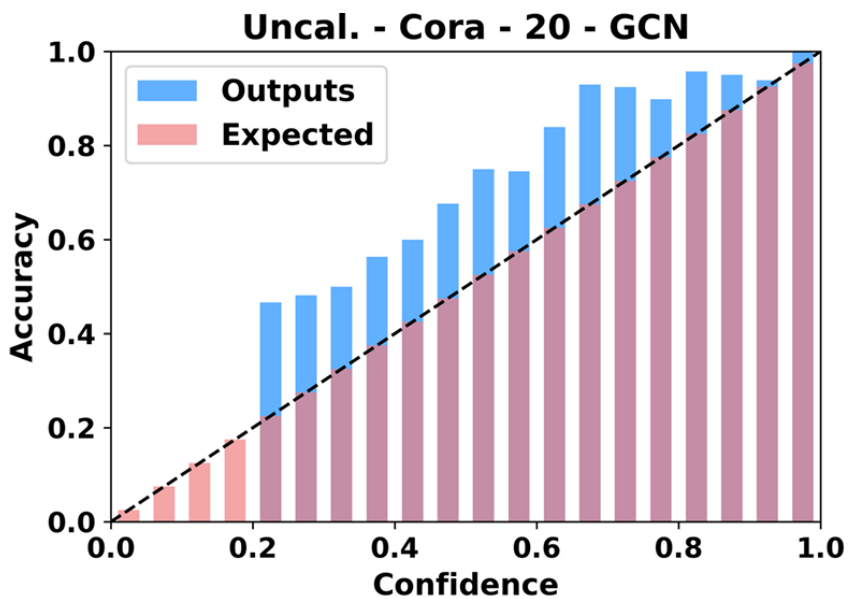
- Of course, while ensuring coverage guarantees!

How to improve efficiency?



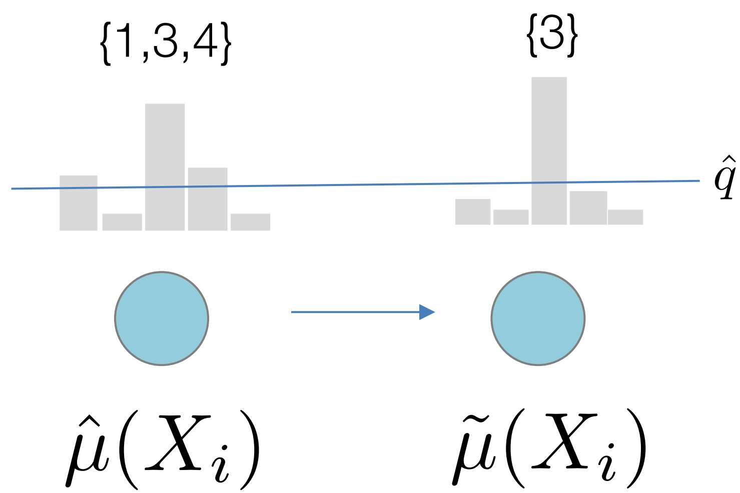
Key observation: GNNs prediction scores are not optimized for conformal efficiency.

GNN prediction scores are shown to be biased for uncertainty quantification



GNN prediction scores are under-confident.

Implications for conformal prediction:

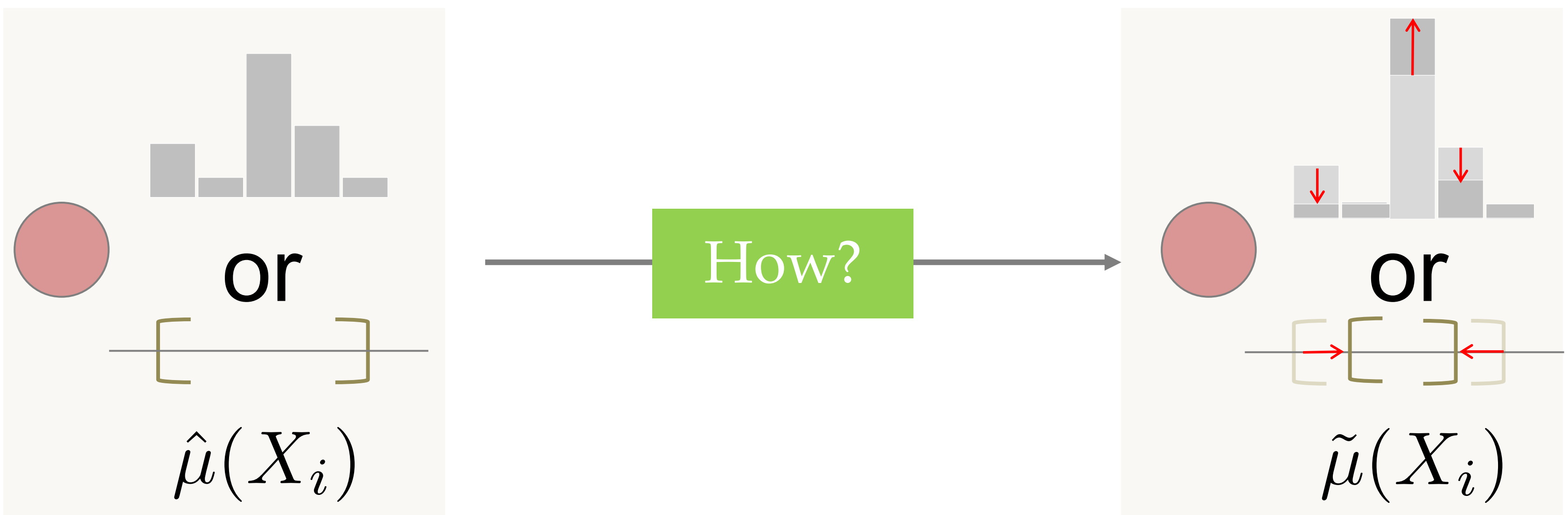


Can we update scores automatically to maximize efficiency?

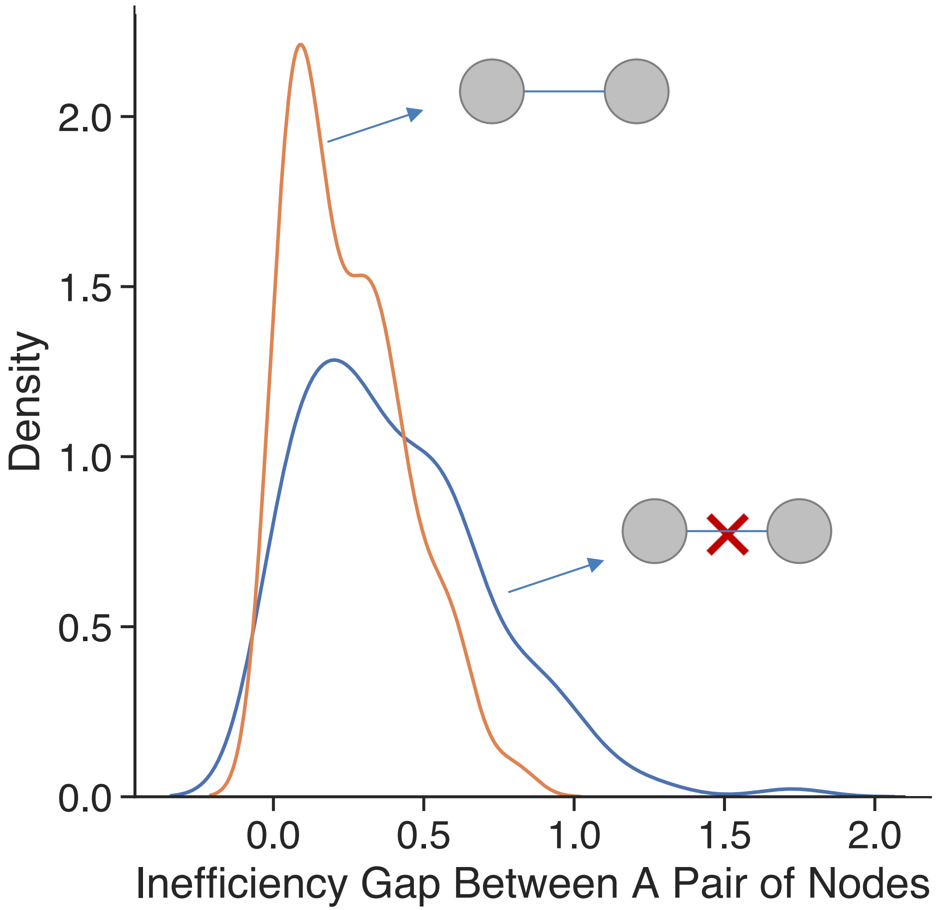
Wang, Xiao, et al. "Be confident! towards trustworthy graph neural networks via confidence calibration." *NeurIPS 2021.*

Can we learn to update the scores to make it conformal aware?

- A post-hoc step (not affect training!)

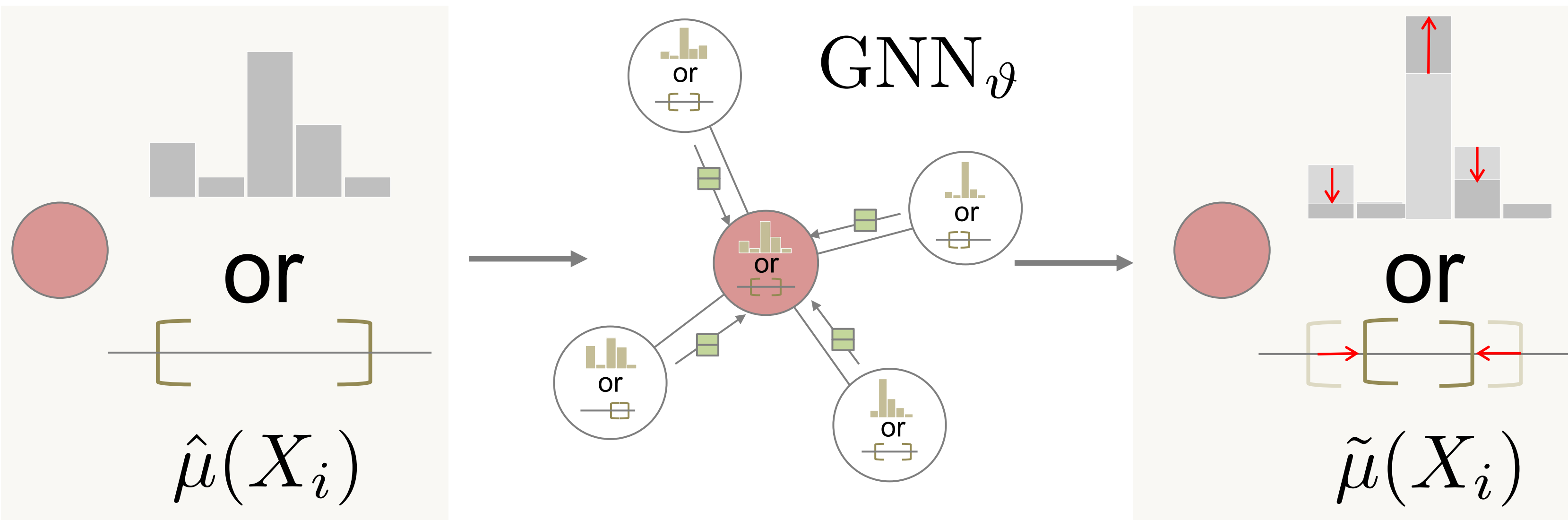


How to adjust the prediction scores for GNNs?



- Inefficiency has a topological root?
- Could we use network structure to improve efficiency?

Key idea: use another GNN over prediction scores

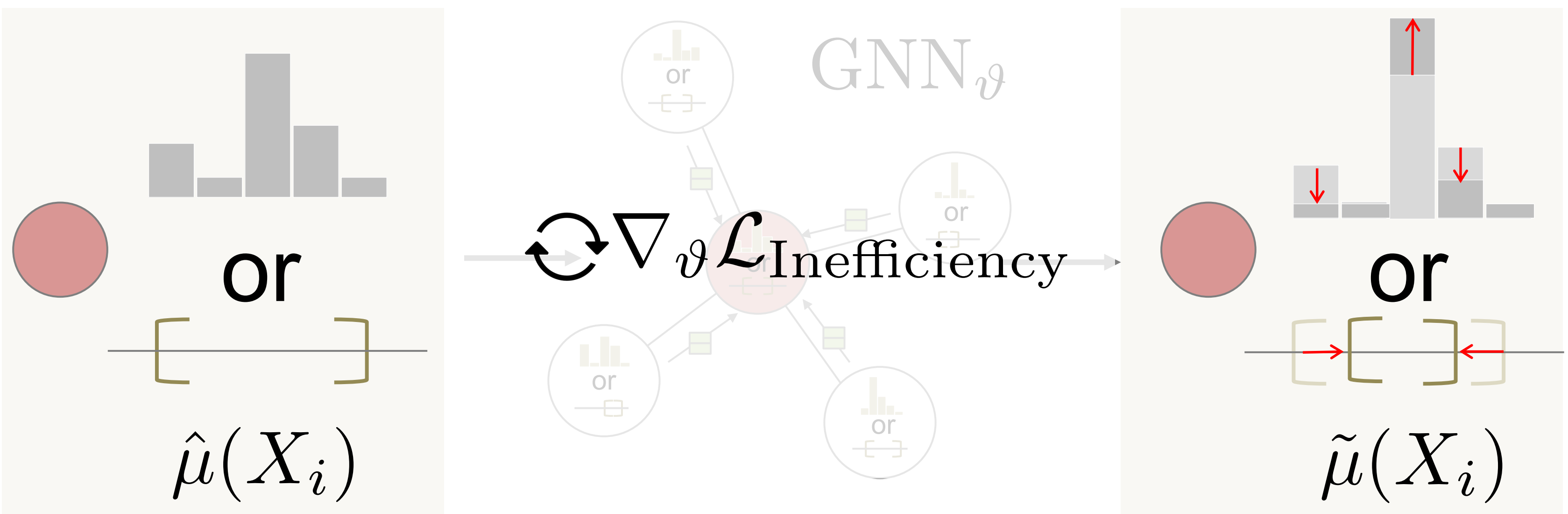


- Note: the input node embedding is prediction scores from the base GNN.
- Asking around the neighbors on how to update the prediction scores to maximize efficiency.

How do we update the correction GNN?

Key insight: design a loss to approximate efficiency and directly learn to optimize over it

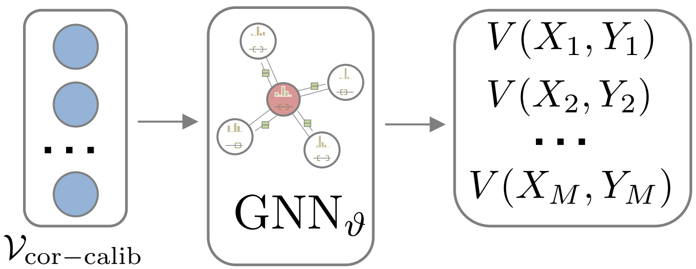
This requires us to make the downstream conformal step differentiable.

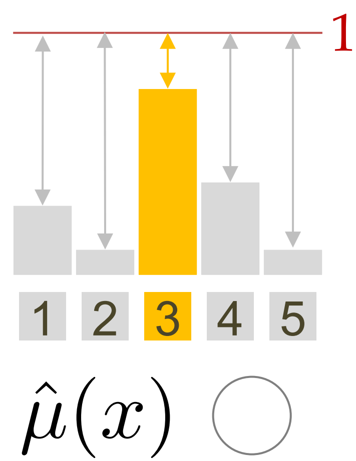


Differentiable conformal proxy

Step 1: define heuristic uncertainty
Step 2: non-conformity scores

Step 3: quantile computation
Step 4: prediction set/interval construction

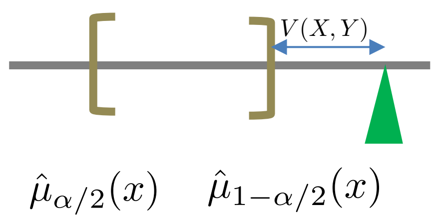




$$V(X,Y) = 1 - \tilde{\mu}(X)_{(Y)}$$

Differentiable!

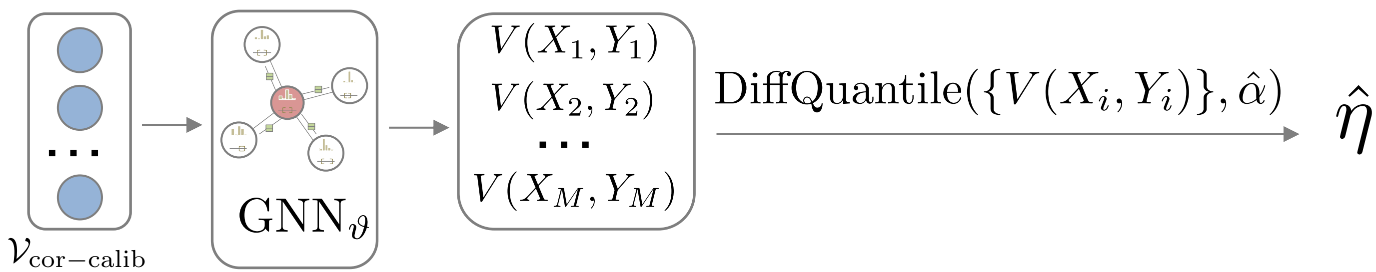
$$V(X,Y) = \max(\tilde{\mu}_{\alpha/2}(X) - Y, \\ Y - \tilde{\mu}_{1-\alpha/2}(X))$$

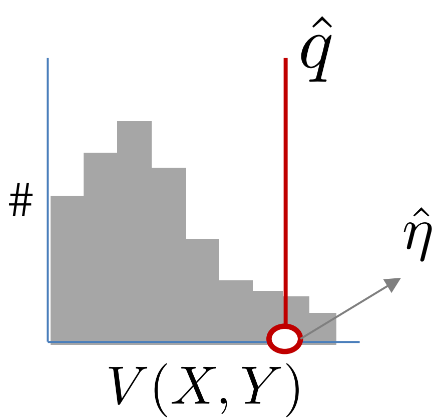


Differentiable conformal proxy

Step 1: define heuristic uncertainty
Step 2: non-conformity scores

Step 3: quantile computation
Step 4: prediction set/interval construction





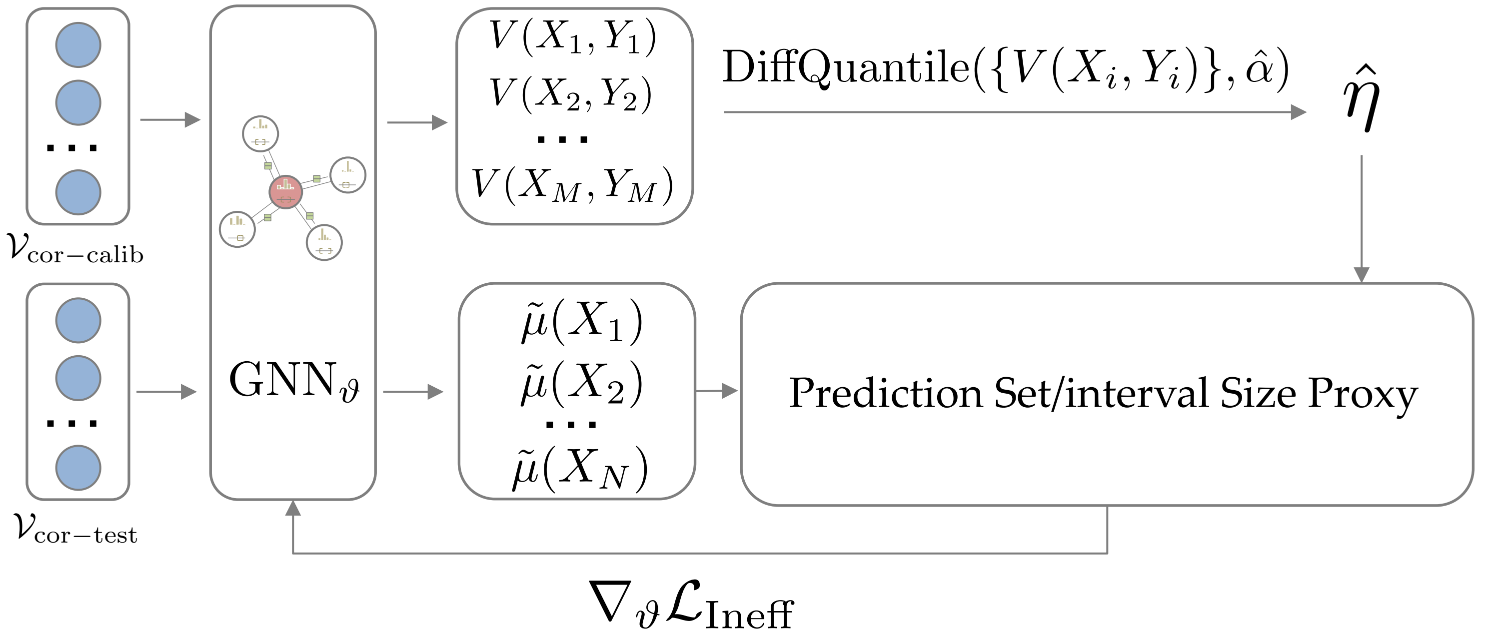
- Smooth quantile operator

(e.g. torch.quantile)

Differentiable conformal proxy

Step 1: define heuristic uncertainty
Step 2: non-conformity scores

Step 3: quantile computation
Step 4: prediction set/interval construction



$$\text{Inefficiency} := \frac{1}{|\mathcal{D}_{\text{test}}|} \sum_{i \in \mathcal{D}_{\text{test}}} |C(X_i)|$$

$$|C(X_{n+1})| = |\{y \in \mathcal{Y} : V(X_{n+1}, y) \leq \hat{\eta}\}|$$

Not differentiable

Differentiable conformal proxy

Step 1: define heuristic uncertainty
Step 2: non-conformity scores

Step 3: quantile computation
Step 4: prediction set/interval construction

Prediction Set Size Proxy

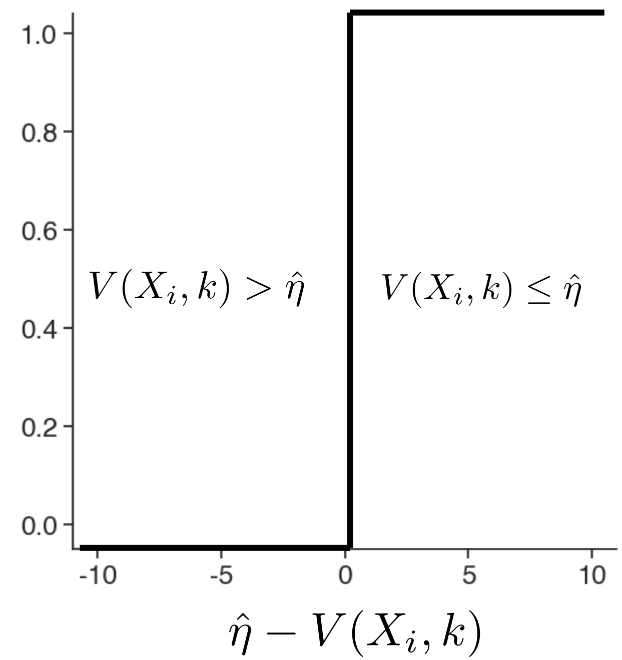
$$|C(X_{n+1})| = |\{y \in \mathcal{Y} : V(X_{n+1}, y) \leq \hat{\eta}\}|$$

$V(X_i, k) \leq \hat{\eta} \longrightarrow 1$

$V(X_i, k) > \hat{\eta} \longrightarrow 0$

Use smooth indicator function:

$$\sigma((\hat{\eta} - V(X_i, k))/\tau)$$



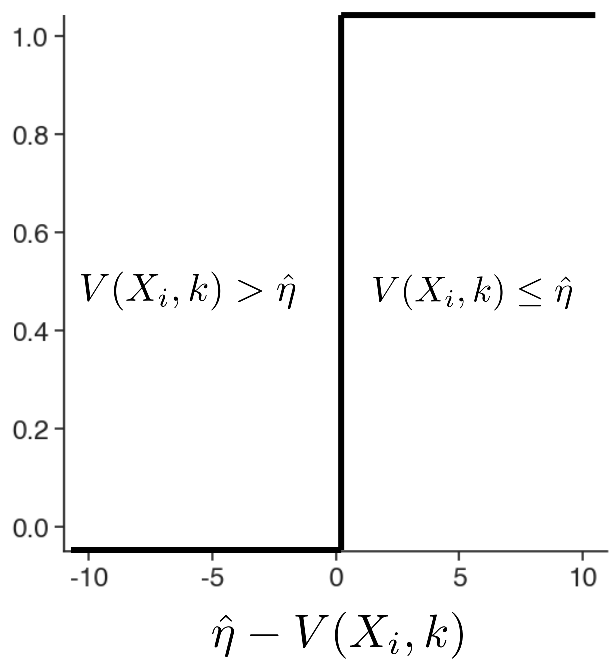
Differentiable conformal proxy

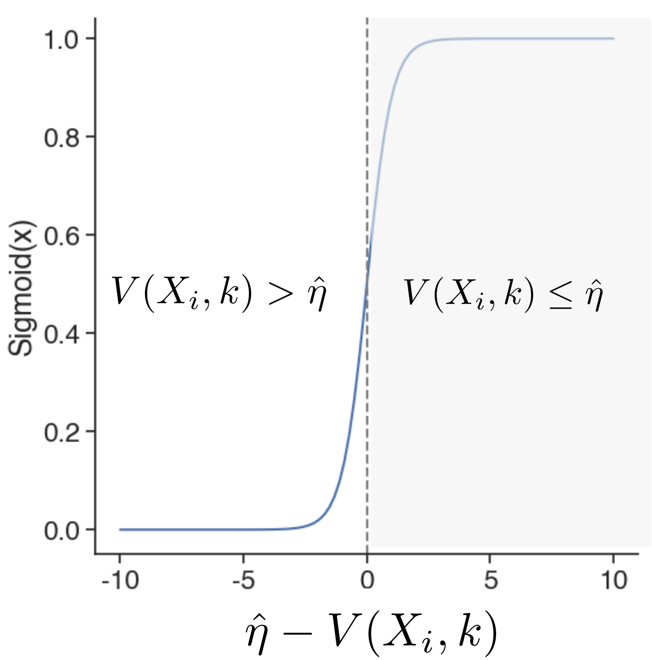
Step 1: define heuristic uncertainty
Step 2: non-conformity scores

Step 3: quantile computation
Step 4: prediction set/interval construction

Prediction Set Size Proxy

$$\mathcal{L}_{\text{Ineff}} = \frac{1}{N} \sum_{i \in \mathcal{V}_{ct}} \sum_{k \in \mathcal{Y}} \sigma((\hat{\eta} - V(X_i, k))/\tau)$$





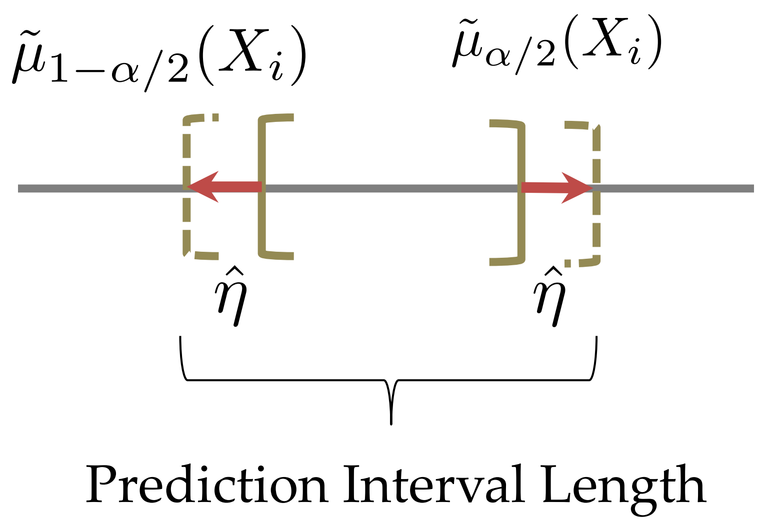
Differentiable conformal proxy

Step 1: define heuristic uncertainty
Step 2: non-conformity scores

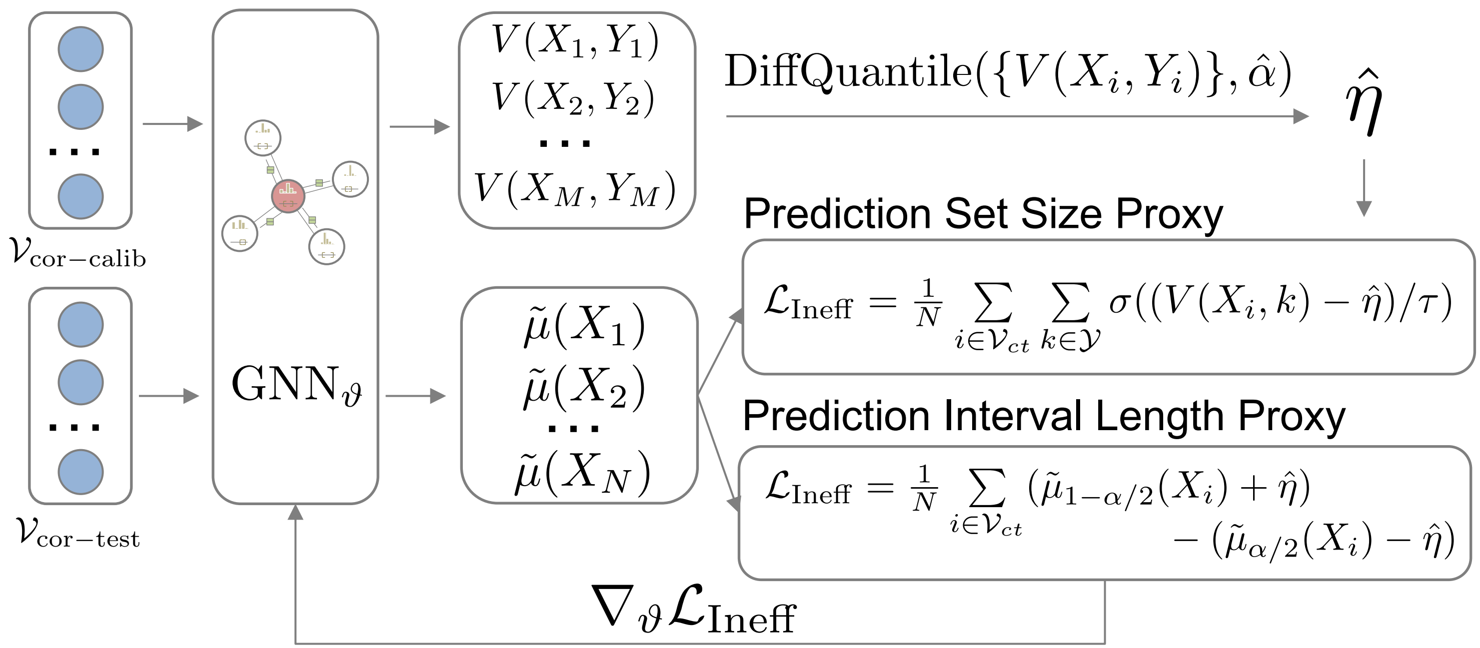
Step 3: quantile computation
Step 4: prediction set/interval construction

Prediction Interval Length Proxy

$$\mathcal{L}_{\text{Ineff}} = \frac{1}{N} \sum_{i \in \mathcal{V}_{ct}} (\tilde{\mu}_{1-\alpha/2}(X_i) + \hat{\eta}) - (\tilde{\mu}_{\alpha/2}(X_i) - \hat{\eta})$$



Overview



- Will the correction step affect coverage guarantees?
  - Because the update step is also a permutation invariant GNN.
  - Based on theorem 1, it still achieves coverage guarantees.

CF-GNN achieves empirical coverage guarantee

Task	UQ Model	Cora	DBLP	CiteSeer	PubMed	Computers	Covered?
Node classif.	Temp. Scale.	0.946±.003 ✗	0.920±.009 ✗	0.952±.004 ✓	0.899±.002 ✗	0.929±.002 ✗	✗
	Vector Scale.	0.944±.004 ✗	0.921±.009 ✗	0.951±.004 ✓	0.899±.003 ✗	0.932±.002 ✗	✗
	Ensemble TS	0.947±.003 ✗	0.920±.008 ✗	0.953±.003 ✓	0.899±.002 ✗	0.930±.002 ✗	✗
	CaGCN	0.939±.005 ✗	0.922±.004 ✗	0.949±.005 ✗	0.898±.003 ✗	0.926±.003 ✗	✗
	GATS	0.939±.005 ✗	0.921±.004 ✗	0.951±.005 ✓	0.898±.002 ✗	0.925±.002 ✗	✗
	CF-GNN	0.952±.001 ✓	0.952±.001 ✓	0.953±.001 ✓	0.953±.001 ✓	0.952±.001 ✓	✓

Task	UQ Model	Anaheim	Chicago	Education	Election	Twitch	Covered?
Node regress.	QR	0.943±.031 ✗	0.950±.007 ✗	0.959±.001 ✓	0.956±.004 ✓	0.900±.015 ✗	✗
	MC dropout	0.553±.022 ✗	0.427±.015 ✗	0.423±.013 ✗	0.417±.008 ✗	0.448±.017 ✗	✗
	BayesianNN	0.967±.001 ✓	0.955±.003 ✓	0.957±.002 ✓	0.958±.009 ✓	0.923±.006 ✗	✗
	CF-GNN	0.957±.003 ✓	0.954±.002 ✓	0.951±.001 ✓	0.950±.001 ✓	0.954±.001 ✓	✓

CF-GNN enables drastic efficiency gain

Task	Dataset	CP $\longrightarrow$ CF-GNN
Node classif.	Cora	$3.80_{\pm.28} \xrightarrow{-53.61\%} 1.76_{\pm.27}$
	DBLP	$2.43_{\pm.03} \xrightarrow{-49.13\%} 1.23_{\pm.01}$
	CiteSeer	$3.86_{\pm.11} \xrightarrow{-74.27\%} 0.99_{\pm.02}$
	PubMed	$1.60_{\pm.02} \xrightarrow{-19.05\%} 1.29_{\pm.03}$
	Computers	$3.56_{\pm.13} \xrightarrow{-49.05\%} 1.81_{\pm.12}$
	Photo	$3.79_{\pm.13} \xrightarrow{-56.28\%} 1.66_{\pm.21}$
	CS	$7.79_{\pm.29} \xrightarrow{-62.16\%} 2.95_{\pm.49}$
	Physics	$3.11_{\pm.07} \xrightarrow{-62.81\%} 1.16_{\pm.13}$
Average Improvement		-53.75%

Task	Dataset	CP $\longrightarrow$ CF-GNN
Node regress.	Anaheim	$2.89_{\pm.39} \xrightarrow{-25.00\%} 2.17_{\pm.11}$
	Chicago	$2.05_{\pm.07} \xrightarrow{-0.48\%} 2.04_{\pm.17}$
	Education	$2.56_{\pm.02} \xrightarrow{-5.07\%} 2.43_{\pm.05}$
	Election	$0.90_{\pm.01} \xrightarrow{+0.21\%} 0.90_{\pm.02}$
	Income	$2.51_{\pm.12} \xrightarrow{-4.58\%} 2.40_{\pm.05}$
	Unemploy	$2.72_{\pm.03} \xrightarrow{-10.83\%} 2.43_{\pm.04}$
	Twitch	$2.43_{\pm.10} \xrightarrow{-1.36\%} 2.39_{\pm.07}$
Average Improvement		-6.73%

More results on conditional coverage, sensitivity analysis, other GNNs, etc. in the paper!

Lots of exciting follow-ups…

- Within the last 6 months
  - Extension to link prediction setting[1]
  - Extension to non-uniform split[2]
  - Extension to inductive setting[3]
  - Extension to edge exchangeability[3]
  - ……

1 Conformal Link Prediction to Control the Error Rate.
2 On the Validity of Conformal Prediction for Network Data Under Non-Uniform Sampling.
3 Conformal Inductive Graph Neural Networks.

Thank you!

- How to measure if an uncertainty estimation method is good?
- How to produce uncertainty estimates with reliability guarantees?
- How to produce reliable uncertainty estimates for graphs?

@KexinHuang5
kexinhuang.com
kexinh@cs.stanford.edu

Kexin Huang

Ying Jin

Emmanuel Candès

Jure Leskovec