ON ESTIMATING LINK PREDICTION UNCERTAINTY USING STOCHASTIC CENTERING

Puja Trivedi[1]*, Danai Koutra*[1]*, Jayaraman J. Thiagarajan*[2]

[1]University of Michgan, [2]Lawrence Livermore National Laboratory

ABSTRACT

Accurate confidence estimates are crucial for safe graph neural network (GNN) deployment, yet link prediction (LP) calibration is understudied. We provide novel insights into LP calibration by highlighting the importance of meaningful node-level uncertainties. In response, we propose E-ΔUQ, an architecture-agnostic framework leveraging stochastic centering to incorporate epistemic uncertainty into GNNs. Our work provides principles and three E-ΔUQ variants to improve trust in LP models, while introducing minimal overhead. Key results demonstrate properly handling node-level uncertainty improves edge calibration. We evaluate E-ΔUQ variants on citation networks and find that intermediate stochastic layers outperform alternatives by producing better node uncertainties. E-ΔUQ reduces calibration error by 15-50% and maintains comparable prediction fidelity.

Index Terms— Graph Neural Networks, uncertainty, link prediction, auxiliary tasks

1. INTRODUCTION

Graph neural networks (GNNs) are becoming highly prevalent in high-impact link prediction (LP) [1, 2] tasks ranging from product recommendation [3] to biological network completion (e.g., gene-gene interaction or drug-drug interactions) [4, 5]. Most often, the predicted links are used to invoke expensive actions or time-consuming experiments. Consequently, in addition to obtaining accurate predictions, it is important that practitioners are able to trust a model's confidence in its predictions [6]. This has led to the emergence of a large class of calibration techniques [7, 8, 9, 10]. Calibration is the process of adjusting output probabilities or confidence scores produced by a model to ensure that they accurately reflect the true likelihood associated with a specific prediction [11, 8]. While most existing studies on GNN calibration have extensively focused on node [12, 13, 14] or graph classification [15], the calibration behavior of LP models remains considerably less studied.

At a high level, link prediction architectures contain an encoder, which produces node-level features, and a light-weight decoder, which aggregates two node representations $(v_{\text{i}}, v_{\text{j}})$ to predict whether a given edge $(e_{(i,j)})$ is plausible. Indeed, it is challenging to directly extend state-of-the-art approaches for improving calibration from node or graph classification literature to LP settings. Though one can systematically estimate uncertainties from the encoder module, the lack of any node-level task makes it challenging to ensure that those uncertainties are well calibrated. Furthermore, it is highly non-trivial to model the interaction between node uncertainties for different LP decoder choices (e.g., node feature concatenation, scalar dot product). For example, it is unclear if an edge $(v_{\text{i}}, v_{\text{j}})$ between two nodes with high (node-level) uncertainties in their features is always guaranteed to have higher (edge-level) uncertainty compared to another edge $(v'_{\text{i}}, v'_{\text{j}})$ with only one node with high uncertainty.

Recently, [16] proposed a Bayesian approach for link prediction, which places an explicit prior over node features (in each layer) and uses a hierarchical Gaussian process (GP) to combine node-level priors to obtain edge-level predictions. While such an approach allows for closed-form aggregation of node-level uncertainties, it has a number of challenges. First, it requires a specific link-predictor structure (hierarchical edge-GP), which may not be compatible with the different decoder choices used in practice. Second, given the high computational costs associated with GP inferencing, this can be especially problematic when scaling to larger, production-scale datasets. Third, it is not straightforward to integrate any additional (or auxiliary) node-level tasks that can help better calibrate the node features [17].

To circumvent these challenges, we propose a non-parametric, architecture-agnostic, LP uncertainty estimator based on the recently proposed stochastic centering framework [18, 19]. We choose this framework for its flexibility to be adopted to any architecture, as well as, its strong generalization behavior under challenging distribution shifts [15].

Extending stochastic centering to edge-level uncertainty (Sec. 3): We first extend stochastic centering to link prediction networks by considering the node features to be deterministic and enabling uncertainty estimation only in the decoder module. This variant (**E-ΔUQ (v1)**) directly models the epistemic uncertainties arising from the sampling of edges in different parts of the (node) feature space.

Creating Meaningful Node-level uncertainties (Sec. 3): Despite the simplicity of the previous variant and its performance in practice, incorporating the node-level uncertainties can lead

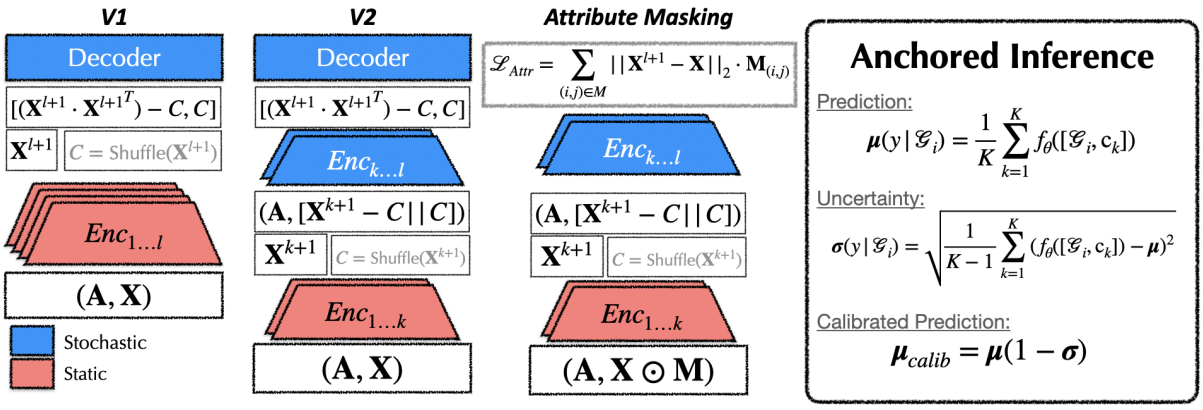

Fig. 1: **Overview of E-ΔUQ.** We propose three different stochastic centering variants that induce varying levels of stochasticity in the underlying GNN. Variant (**E-ΔUQ (v1)**) directly models the epistemic uncertainties arising from the sampling of edges in different parts of the (node) feature space. Variant (**E-ΔUQ (v2)**) performs stochastic centering in the encoder network itself and implicitly leverage those uncertainties to produce calibrated edge probabilities. **E-ΔUQ (v3)**) uses the auxiliary attribute masking task to first calibrate the node-level uncertainties and subsequently estimate the edge-level uncertainties similar to **E-ΔUQ (v2)**. We show the attribute masking task above and use shuffled node features as the anchoring distribution.

to richer LP models. Hence, we propose to invoke stochastic centering in the encoder network itself (**E-ΔUQ (v2)**) and implicitly leverage those uncertainties to produce calibrated edge probabilities. Finally, we consider a sophisticated variant (**E-ΔUQ (v3)**) where we leverage an auxiliary task to first calibrate the node-level uncertainties and subsequently estimate the edge-level uncertainties similar to **E-ΔUQ (v2)**.

Experimental Evaluation (Sec. 4): Using a suite of citation network datasets, we systematically evaluate the three proposed uncertainty estimation techniques and demonstrate their behavior in terms of both fidelity of the predicted links and calibration error metrics. This work, for the first time, delves into the important problem of appropriately handling node-level uncertainties in LP architectures.

2. PRELIMINARIES AND RELATED WORK

In this section, we briefly discuss the notations and related work relevant to problem setting and approach.

Notations. Let $\mathcal{G} = (\mathbf{X}, \mathbf{A})$ be a graph with node features $\mathbf{X} \in \mathbb{R}^{N \times d_\ell}$, and adjacency matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$, where N, m, d_ℓ, denotes the number of nodes, number of edges, and feature dimension. We wish to predict unobserved edges that are missing from $\mathbf{A}$ given the observed, training graph and nodes. Thus, we can define a LP GNN consisting of a node `Encoder` with ℓ message passing layers (`MPNN`), and `Decoder` which predicts whether an edge $e(i, j)$ exists between two nodes, given their representations (X_i, X_j):

$$\mathbf{X}^{\ell+1} = \texttt{Encoder}\left(\mathbf{X}^{\ell}, \mathbf{A}\right), \quad (1)$$

$$E_{(i,j)} = \texttt{Decoder}\left(\mathbf{X}_i^{\ell+1} \mathbf{X}_j^{\ell+1}\right), \quad (2)$$

$$(3)$$

where $\mathbf{X}^{\ell+1}$ is the intermediate node representations. Popular decoder architectures include taking the dot product or concatenating representations and then passing the resulting representation through some linear layers. Models are trained by treating LP as a binary classification task, where true edges in $\mathbf{A}$ are considered positive class samples and non-edges in $\mathbf{A}$ are considered negative class samples.

Calibration and Stochastic Centering. While several strategies have been proposed to improve calibration [7, 8, 9, 10] of vision models and GNN-based node classification calibration [12, 13, 14, 15], these methods are not suited for link prediction calibration because they cannot ensure reliable node-level calibration without node-level supervision (see Sec. 1) and often struggle to outperform the simple, but prohibitively expensive deep ensemble (DEns) [7] baseline. Unfortunately, DEns, which takes the mean prediction over a set of independently trained models, requires training and storing multiple models. Recently, however, [18] proposed a state-of-the-art, single model uncertainty estimation method, Δ-UQ, based on the principle of *anchoring*, which is capable of simulating the behavior of an ensemble through only a single model.

Conceptually, anchoring is the process of creating a relative representation for an input sample x in terms of a random anchor c (which is used to perform the *stochastic centering*), $[x - c, c]$. By choosing different anchors randomly in each training iteration, Δ-UQ emulates the process of sampling different solutions from the hypothesis space (akin to an ensemble). During inference, Δ-UQ aggregates multiple predictions obtained via different random anchors and produces uncertainty estimates.

Formally, given a trained stochastically centered model, $f_\theta : [\mathbf{X} - \mathbf{C}, \mathbf{C}] \rightarrow \hat{\mathbf{Y}}$, let $\mathbf{C} := \mathbf{X}_{train}$ be the anchor dis-

tribution, $x \in \mathbf{X}_{test}$ be a test sample, and anchor $c \in \mathbf{C}$ be anchor. Then, the mean target class prediction, $\boldsymbol{\mu}(y|\mathrm{x})$, and corresponding variance, $\boldsymbol{\sigma}(y|\mathrm{x})$ over K random anchors are computed as:

$$\boldsymbol{\mu}(y|\mathrm{x}) = \frac{1}{K}\sum_{k=1}^{K} f_\theta([\mathrm{x} - \mathrm{c}_k, \mathrm{c}_k]) \tag{4}$$

$$\boldsymbol{\sigma}(y|\mathrm{x}) = \sqrt{\frac{1}{K-1}\sum_{k=1}^{K}(f_\theta([\mathrm{x} - \mathrm{c}_k, \mathrm{c}_k]) - \boldsymbol{\mu})^2} \tag{5}$$

Since the variance over K anchors captures epistemic uncertainty by sampling different hypotheses, these estimates can be used to modulate the predictions: $\boldsymbol{\mu}_{\text{calib.}} = \boldsymbol{\mu}(1 - \boldsymbol{\sigma})$. The resulting calibrated predictions and uncertainty estimates have led to state-of-the-art performance on vision [18, 19] and graph classification tasks [15], while still only requiring a single model. Given its impressive performance and flexibility, we focus on adapting stochastic centering to our link prediction calibration setting. Namely, we discuss in detail design considerations that arise from aggregating node-level uncertainties into edge-level confidence estimates.

3. PROPOSED APPROACH

In this section, we introduce our proposed approach, and discuss the three variants, for improving GNN-based link prediction calibration using stochastic centering (see Fig. 1). Furthermore, we will demonstrate the importance of creating and aggregating meaningful node-level uncertainties. Indeed, as discussed in Sec. 1, on the one hand, LP calibration is difficult since node-level uncertainties must be correctly aggregated in order to produce consistent edge-level uncertainties. On the other hand, due to the lack of suitable node-level tasks, the uncertainties associated with the node features can themselves be poorly calibrated. Consequently, even a sophisticated aggregation strategy in LP decoders can lead to sub-par calibration of edge probabilities. Below, we introduce three different variants of our proposed method, where each variant gradually seeks to improve the characterization of node-level uncertainties, and subsequently leverages the stochastic centering framework to produce edge uncertainties.

E-ΔUQ (v1) – Deterministic node features: While stochastic centering can be performed at any layer of the GNN, in this variant, stochastic centering is only performed prior to decoder. In other words, the node representations are assumed to be deterministic and that the epistemic uncertainties arise from the non-uniform samples of edges in different parts of the feature space. While, this assumption indicates that we expect the least amount of change on the reliability of node-level features, we perform stochastic feature aggregation over pairs of nodes by utilizing the anchoring framework. Given that the LP decoder paramterizes an edge based on a chosen aggregation function on the given pair of nodes, through anchoring, we sample different possible aggregation hypotheses before marginalizing over anchors according to (7). In other words, given the features $\mathbf{x}_i$ and $\mathbf{x}_j$ for a node pair, we perform stochastic centering using a randomly chosen anchor $\mathbf{c}$ (from a pre-specified anchor distribution). Formally, we define this operation for dot-product and concatenation-style LP decoder modules as follows:

$$\texttt{Decoder}_{dot} : [(\mathbf{x}_i - \mathbf{c}) * (\mathbf{x}_j - \mathbf{c}), \mathbf{c}] \tag{6}$$

$$\texttt{Decoder}_{concat} : [(\mathbf{x}_i - \mathbf{c}) || (\mathbf{x}_j - \mathbf{c}), \mathbf{c}] \tag{7}$$

Similar to standard anchored model training, this tuple is taken as input by the LP decoder and trained using the standard cross entropy loss. Note, in each iteration of training, a different anchor $\mathbf{c}$ is randomly chosen.

E-ΔUQ (v2) – Partially stochastic encoder: As discussed earlier, it is reasonable to expect the node-level uncertainties can be utilized to improve the calibration of the edge probabilities obtained using LP. This is motivated by the fact that, the encoder architecture (implemented using GNNs) can be susceptible to epistemic uncertainties arising from the distribution of node attributes and hence, they can singificantly influence the subsequent predictions on pairs of nodes. Hence, in this variant, we invoke anchoring in the intermediate layers of the encoder architecture itself, and obtain stochastic node representations. Formally,

$$\begin{aligned}
\mathbf{X}^{r+1} &= \texttt{Encoder}^{1\ldots r}(\mathbf{X}, \mathbf{A}) \\
\mathbf{X}^{\ell+1} &= \texttt{Encoder}^{r+1\ldots \ell}\left([\mathbf{X}^{r+1} - \mathbf{C}, \mathbf{C}], \mathbf{A}\right) \\
\hat{E}_{(i,j)} &= \texttt{Decoder}\left(\mathbf{X}_i^{\ell+1}, \mathbf{X}_j^{\ell+1}\right)
\end{aligned}$$

However, it is important to note that this approach results in partially stochastic encoder model (i.e., first few layers of the encoder are deterministic) and the anchoring process leverages the structural information (through the message passing in GNN layers). By introducing anchoring in the encoder and performing end-to-end training of the LP model, we are able to effectively sample the hypothesis space for joint node feature learning and LP decoding. We expect this increased diversity to help improve the quality of our link-level predictions.

E-ΔUQ (v3) – Partially stochastic encoders + Node-level pretraining. While the aforementioned variants use stochastic centering to implicitly improve the aggregation of uncertainty over pairs of nodes, they do not explicitly improve the quality of node-level uncertainties. Indeed, this is difficult as node-level calibration supervision cannot be assumed on LP tasks. Therefore, we combine E-ΔUQ (v2) with an unsupervised node-level pretraining task to prime the encoder's node-level uncertainties before LP training. In particular, we pretrain the encoder with an auxillary node feature attribute masking task, and then train both the encoder and decoder with the standard link-prediction loss. Formally, assuming $\mathbf{M} \in [0, 1]^{N \times d}$ denotes a random binary mask, we use that to mask portions

of the input node feature matrix and define a self-supervised objective as follows:

$$\mathbf{X}^{r+1} = \texttt{Encoder}^{1\ldots r}(\mathbf{X} \odot \mathbf{M}, , \mathbf{A})$$
$$\mathbf{X}^{\ell+1} = \texttt{Encoder}^{r+1\ldots \ell}\left([\mathbf{X}^{r+1} - \mathbf{C}, \mathbf{C}], \mathbf{A}\right)$$
$$\mathcal{L}_{Attr} = \sum_{(i,j)\in\mathbf{M}} ||\mathbf{X}^{\ell+1} - \mathbf{X}||_2 \cdot \mathbf{M}_{(i,j)}$$

Here, $\|.\|_2$ denotes the ℓ_2 norm and the reconstruction loss is measures only using the masked parts of the feature matrix. After completing the attribute masking-based pretraining, the encoder is equipped to produce node-level uncertainties. Subsequently, both the encoder and decoder modules of the LP architecture are trained end-to-end, following E-ΔUQ (v2). While other pretraining tasks can be considered, we use attribute masking for effectiveness and ease of implementation.

4. EXPERIMENTS

In this section, we experimentally validate the effectiveness of our three LP calibration variants. *Experimental Set-up:* We consider three different datasets (Cora, Citeseer, Pubmed) and use the publicly available train-test splits for evaluation. A 3-layer GraphSAGE [20] backbone is used for the encoder, with either a dot-product decoder (Cora) or a concatenation decoder (Citeseer, Pubmed) (Table 4). Ten anchors are used for all E-ΔUQvariants. Hyper-parameters are shared between vanilla and E-ΔUQmodels.[1] The AUPRC and expected calibration error are reported over 10 seeds. We report results for the best intermediate anchoring layer according to validation AUPRC. We make the following observations from Table 4.

Observation 1: Stochastic centering variants improves the calibration on all datasets over the vanilla model. Indeed, the improvement is particularly large on Cora, where ECE is decreased by 50%, and Citeseer, where ECE is decreased by 16%. While we do not see as large gains on Pubmed, we do note that no E-ΔUQ variants increases the calibration error. This clearly suggests that our stochastic centering approach is effective.

Observation 2: Stochastic centering variants perform comparably on AUPR, with E-ΔUQ variants performing the best on 2/3 datasets. Generally, we see that E-ΔUQ variants improve AUPR 4/9, though we suspect that E-ΔUQ performance could be further improved if we tuned method-specific hyper-parameters.

Observation 3: Amongst E-ΔUQ variants, E-ΔUQ (v3) obtains the best calibration on 2/3 datasets. This suggests there is value to our pretraining method, which seeks to improve node-level calibration. We suspect that E-ΔUQ (v3)'s performance could be further improved with better auxiliary tasks, but we leave the design of such tasks to future work.

Observation 4: E-ΔUQ (v2) induced better calibration than E-ΔUQ (v1) on 3/3 datasets, and has better AUPR on 2/3 datasets. This supports our argument better node-level calibration is critical for also improving LP calibration.

Table 1: **Dataset Statistics.**

Name	#nodes	#edges	#features
Cora	2,708	10,556	1,433
CiteSeer	3,327	9,104	3,703
PubMed	19,717	88,648	500

5. CONCLUSION

In this work, we proposed and evaluated three variants of stochastic centering for improving the calibration of graph neural networks for link prediction. Our key finding is that properly accounting for node-level uncertainty is critical for obtaining well-calibrated edge-level confidence estimates. Our experiments on three citation networks demonstrated that our proposed E-ΔUQ methods can substantially reduce the expected calibration error compared to vanilla models. Overall, this work provides novel insights into the importance of node-level uncertainty modeling for link prediction calibration and our proposed stochastic centering framework offers a flexible way to incorporate epistemic uncertainty into existing GNN architectures in a principled manner. An interesting direction for future work is exploring additional auxiliary pretraining objectives to further improve the meaningfulness of node uncertainties.

Table 2: **Link Prediction Calibration.**

Dataset	**Method**	**AUPR** (↑)	**ECE** (↓)
Citeseer	E-ΔUQ (v3)	0.8409 ±0.0115	**0.2591** ±0.0178
	E-ΔUQ (v2)	**0.8548** ±0.0076	0.2833 ±0.0075
	E-ΔUQ (v1)	0.8070 ±0.0218	0.3056 ±0.0109
	Vanilla	0.8236 ±0.0115	0.3002 ±0.0062
Cora	E-ΔUQ (v3)	0.8886 ±0.0042	**0.1554** ±0.0060
	E-ΔUQ (v2)	0.8888 ±0.0062	0.1731 ±0.0181
	E-ΔUQ (v1)	0.8598 ±0.0207	0.2640 ±0.0125
	Vanilla	**0.8936** ±0.0066	0.3503 ±0.0146
Pubmed	E-ΔUQ (v3)	0.8775 ±0.0098	0.1818 ±0.0048
	E-ΔUQ (v2)	0.8701 ±0.0016	**0.1538** ±0.0059
	E-ΔUQ (v1)	**0.9069** ±0.0063	0.1801 ±0.0117
	Vanilla	0.8897 ±0.0091	0.1980 ±0.0035

[1]Code will be released.

6. ACKNOWLEDGMENTS

This work was performed under the auspices of the U.S. Department of Energy by the Lawrence Livermore National Laboratory under Contract No. DE-AC52-07NA27344, and is partially supported by the LLNL-LDRD Program under Project No. 22-SI-004 with IM release number LLNL-CONF-850978. This work is also supported by the National Science Foundation under CAREER Grant No. IIS 1845491, and Grant No. IIS 2212143. Any opinions, findings, and conclusions or recommendations expressed here are those of the author(s) and do not reflect the views of funding parties.

7. REFERENCES

[1] Muhan Zhang and Yixin Chen, "Link prediction based on graph neural networks," in *Proc. Adv. in Neural Information Processing Systems NeurIPS*, 2018.

[2] Muhan Zhang and Yixin Chen, "Link prediction based on graph neural networks," in *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett, Eds., 2018, pp. 5171–5181.

[3] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L. Hamilton, and Jure Leskovec, "Graph convolutional neural networks for web-scale recommender systems," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19-23, 2018*, Yike Guo and Faisal Farooq, Eds. 2018, pp. 974–983, ACM.

[4] Marinka Zitnik, Rok Sosič, and Jure Leskovec, "Prioritizing network communities," *Nature Communications*, 2018.

[5] Doyeong Hwang, Soojung Yang, Yongchan Kwon, Kyung-Hoon Lee, Grace Lee, Hanseok Jo, Seyeol Yoon, and Seongok Ryu, "Comprehensive study on molecular supervised learning with graph neural networks," *J. Chem. Inf. Model.*, 2020.

[6] Dan Hendrycks, Nicholas Carlini, John Schulman, and Jacob Steinhardt, "Unsolved problems in ML safety," *CoRR*, vol. abs/2109.13916, 2021.

[7] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles," in *Proc. Adv. in Neural Information Processing Systems (NeurIPS)*, 2017.

[8] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger, "On calibration of modern neural networks," in *Proc. of the Int. Conf. on Machine Learning, (ICML)*, 2017.

[9] Yarin Gal and Zoubin Ghahramani, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," in *Proc. Int. Conf. on Machine Learning (ICML)*, 2016.

[10] Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra, "Weight uncertainty in neural network," in *Proc. Int. Conf. on Machine Learning (ICML)*, 2015.

[11] Glenn W. Brier, "Verification of forecasts expressed in terms of probability," in *Monthly Weather Review*, 1950.

[12] Hans Hao-Hsun Hsu, Yuesong Shen, Christian Tomani, and Daniel Cremers, "What makes graph neural networks miscalibrated?," in *Proc. Adv. in Neural Information Processing Systems NeurIPS*, 2022.

[13] Xiao Wang, Hongrui Liu, Chuan Shi, and Cheng Yang, "Be confident! towards trustworthy graph neural networks via confidence calibration," in *Proc. Adv. in Neural Information Processing Systems NeurIPS*, 2021.

[14] Jian Kang, Qinghai Zhou, and Hanghang Tong, "Jurygcn: Quantifying jackknife uncertainty on graph convolutional networks," in *Proc. Int. Conf. on Knowledge Discovery & Data Mining, KDD*, 2022.

[15] Puja Trivedi, Mark Heimann, Rushil Anirudh, Danai Koutra, and Jayaraman J. Thiagarajan, "Estimating epistemic uncertainty of graph neural networks," in *Data Centric Machine Learning Workshop @ ICML*, 2023.

[16] Felix L. Opolka and Pietro Liò, "Bayesian link prediction with deep graph convolutional gaussian processes," in *International Conference on Artificial Intelligence and Statistics,AISTATS*, 2022.

[17] Weihua Hu, Bowen Liu, Joseph Gomes, Marinka Zitnik, Percy Liang, Vijay S. Pande, and Jure Leskovec, "Strategies for pre-training graph neural networks," in *ICLR*, 2020.

[18] Jayaraman J. Thiagarajan, Rushil Anirudh, Vivek Narayanaswamy, and Peer-Timo Bremer, "Single model uncertainty estimation via stochastic data centering," in *Proc. Adv. in Neural Information Processing Systems (NeurIPS)*, 2022.

[19] Aviv Netanyahu, Abhishek Gupta, Max Simchowitz, Kaiqing Zhang, and Pulkit Agrawal, "Learning to extrapolate: A transductive approach," in *Proc. Int. Conf. on Learning Representations (ICLR)*, 2023.

[20] William L. Hamilton, Zhitao Ying, and Jure Leskovec, "Inductive representation learning on large graphs," in *NeurIPS*, 2017.