Distribution Free Prediction Sets for Node Classification

Jase Clarkson [1]

Abstract

Graph Neural Networks (GNNs) are able to achieve high classification accuracy on many important real world datasets, but provide no rigorous notion of predictive uncertainty. Quantifying the confidence of GNN models is difficult due to the dependence between datapoints induced by the graph structure.

We leverage recent advances in conformal prediction to construct prediction sets for node classification in inductive learning scenarios. We do this by taking an existing approach for conformal classification that relies on *exchangeable* data and modifying it by appropriately weighting the conformal scores to reflect the network structure. We show through experiments on standard benchmark datasets using popular GNN models that our approach provides tighter and better calibrated prediction sets than a naive application of conformal prediction. The code is available at this link.

1. Introduction

Machine learning on graph structured data has seen a boom of popularity in recent years, with applications ranging from recommendation systems to biology and physics. Graph neural networks are quickly maturing as a technology; many state of the art models are commoditised in frameworks such as Pytorch Geometric (Fey and Lenssen, 2019) and DGL (Wang et al., 2019). Despite their overwhelming popularity and success, very little progress has been made towards quantifying the uncertainty of the predictions made by these models, a vital step towards robust real world deployments.

In related areas of machine learning such as computer vision, conformal prediction (Vovk et al., 2005) has emerged as a promising candidate for uncertainty quantification (Angelopoulos et al., 2020). Conformal prediction is a very appealing approach as it is compatible with any black box machine learning algorithm and dataset as long as the data is statistically exchangeable. The most wide-spread method, so called *split-conformal*, also requires trivial computational overhead when compared to model fitting.

Networks, or graph structured data is in general not exchangeable and so the guarantees provided by conformal prediction in its naive form do not hold. Recent work by Barber et al. (2022) extends conformal prediction to the non-exchangeable setting and provides theoretical guarantees on the performance of conformal prediction in this setting. We leverage insights from (Barber et al., 2022) to apply conformal prediction in the node classification setting. The key insight is that for a homophilous graph, the model calibration should be similar in a neighbourhood around any given node. We leverage this insight to localise the calibration of conformal prediction. We show that our method improves calibration of predictive uncertainty and provides tighter prediction sets when compared with a naive application of conformal prediction across several state of the art models applied to popular node classification datasets.

This paper is structured as follows; we begin by reviewing related work in Section 2. In Section 3 we give an introduction to the relevant background material on conformal prediction. We introduce our method for adapting conformal prediction to networks in Section 4. We then detail our experimental setup and provide marginal coverage statistics in Section 6, and discuss the conditional coverage properties of our method in Section 7. We provide an ablation study to better understand our results in 8 and finally give our conclusions and discuss avenues for future work in Section 9.

2. Related Work

There is no standard approach to estimating the predictive uncertainty of neural network models. Many works take a Bayesian approach (Goan and Fookes, 2020), where the goal is to learn a distribution over the network weights and represent uncertainty via the posterior distribution. Bayesian approaches however become quickly computationally intractable for large models and datasets, which has lead to approximations of Bayesian learning such as Deep Ensembles (Lakshminarayanan et al., 2017) and Variational Inference

[1] Department of Statistics, University of Oxford. Correspondence to: Jase Clarkson <jason.clarkson@stats.ox.ac.uk>.

Proceedings of the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023. Copyright 2023 by the author(s).

(Welling and Kingma, 2014). These approximations too come with practical drawbacks, such as the need to explore distinct regions of the parameter space in Deep Ensembles.

In the graph setting, several variants of Bayesian GNNs have been proposed (Hasanzadeh et al., 2020; Chandra et al., 2021; Zhang et al., 2019), all of which require modifications to either model architecture or the model training procedure. The methods introduced in this work are deployed *after* model training and have trivial computational overhead compared to model fitting.

Conformal prediction (Vovk et al., 2005) has seen a surge in popularity in recent years, especially amongst the deep learning community (Stutz et al., 2022; Einbinder et al., 2022; Teng et al., 2022). In the exposition below we will focus on conformal classification as introduced in (Romano et al., 2020) as that is the object of study in this work, but note that other approaches to conformal classification exist such as that introduced in (Sadinle et al., 2019). Conformal prediction may also be used to construct prediction intervals for regression (Lei et al., 2018; Romano et al., 2019) or to control more general risk functions (Angelopoulos et al., 2022).

While we are not aware of prior work extending conformal prediction to networks, very recent work has considered the application of conformal prediction to time series, in which observations are in general not exchangeable. These algorithms often assume the distribution can shift over time, even adversarially, and as such utilise online learning (Gibbs and Candes, 2021; Zaffran et al., 2022; Gibbs and Candès, 2022) or game theoretic (Bastani et al., 2022) techniques. In contrast, we assume that the data is non-exchangeable but that the adjacency matrix of the graph is indicative of the dependency structure between data points.

3. Conformal Prediction

Conformal prediction is a family of algorithms that generate finite sample valid prediction intervals or sets from an arbitrary black box machine learning model. In this work we employ a convenient approach known as *split* conformal prediction (Papadopoulos et al., 2002; Lei et al., 2018). Split conformal prediction may be thought of as a ”wrapper” around a fitted model that uses a set of exchangeable held out data to calibrate prediction sets. Amazingly, the predictive model does not even need be well specified for these guarantees to hold (although the prediction intervals or sets may not be useful in this case). An excellent tutorial is provided by Angelopoulos and Bates (2021).

3.1. The Exchangeable Case

In a $K-$class classification model suppose that we have a fitted model $\hat{f}:\mathcal{X}\to[0,1]^K$ that outputs the probability of each class. Given an exchangeable set of held-out calibration datapoints $(X_1,Y_1),\ldots,(X_n,Y_n)$ (held out meaning they were not used to fit the model) and a new evaluation point (X_{n+1},Y_{n+1}), conformal prediction constructs a *prediction set* $\hat{C}(X_{n+1})$ that satisfies

$$1-\alpha\le\mathbb{P}\left(Y_{n+1}\in\hat{C}\left(X_{n+1}\right)\right)\le1-\alpha+\frac{1}{n+1}\quad(1)$$

for a user specified error rate $\alpha\in[0,1]$. Conformal prediction relies on a *score function* $S:\mathcal{X}\times\mathcal{Y}\to\mathbb{R}$, which measures the calibration of the prediction at a given datapoint. Given a score function S, the procedure for constructing a prediction set is very simple; for each datapoint (X_i,Y_i) in the calibration set, compute the score $s_i=S(X_i,Y_i)$. Define $1-\hat{\alpha}$ to be the $\lceil(n+1)(1-\alpha)\rceil/n$ empirical quantile of the scores $s_1,\ldots,s_n$, and finally create the prediction set

$$\hat{C}\left(X_{n+1}\right)=\left\{y:S\left(X_{n+1},y\right)\le1-\hat{\alpha}\right\}.$$

A popular conformal prediction procedure for classification problems is known as Adaptive Prediction Sets (APS, (Romano et al., 2020)). To motivate the APS score function, suppose we have access to an *oracle* classifier π that exactly matches the true conditional distribution (i.e. $\pi(x)=\mathbb{P}(Y_{n+1}|X_{n+1}=x)$). Then to construct a $1-\alpha$ prediction set from the oracle, we simply sort the probabilities into descending order, and add labels to the set until the cumulative probability exceeds $1-\alpha$ (with appropriate tie breaking to ensure exact coverage).

Let

$$\pi_y(x)=\mathbb{P}(Y=y\mid X=x)$$

for all $y\in\mathcal{Y}$ and denote the order statistics of the oracle classifier probabilities by

$$\pi_{(1)}(x)\ge\pi_{(2)}(x)\ge\ldots\ge\pi_{(K)}(x).$$

For any $\tau\in[0,1]$ define the generalised conditional quantile as

$$\begin{aligned}L(x;\pi,\tau)=\min\{k\in\{1,\ldots,K\}:\\ \pi_{(1)}(x)+\pi_{(2)}(x)+\ldots+\pi_{(k)}(x)\ge\tau\}.\end{aligned}$$

One can now define the set valued function

$$\mathcal{S}(x,u;\pi,\tau)=\begin{cases}\text{Labels of the}\\ L(x;\pi,\tau)-1\text{ largest }\pi_y(x),\\ \qquad\text{if }u\le V(x;\pi,\tau),\\ \\ \text{Labels of the}\\ L(x;\pi,\tau)\text{ largest }\pi_y(x),\\ \qquad\text{otherwise}\end{cases}$$

where

$$V(x;\pi,\tau)=\frac{1}{\pi_{(L(x;\pi,\tau))}(x)}\left[\sum_{c=1}^{L(x;\pi,\tau)}\pi_{(c)}(x)-\tau\right].$$

The oracle prediction set is then be defined as

$$C_{\alpha}^{\text{oracle}}(x)=\mathcal{S}(x,U;\pi,1-\alpha)$$

where $U\sim$ Uniform(0,1) is independent of everything else. The above is saying one should break ties proportional to the gap between the cumulative sum of the ordered probabilities until the true label is included and the desired level τ.

In practice the probabilities given by a fitted classifier $\hat{f}(x)$ will usually not be exactly equal to $\mathbb{P}(Y_{n+1}|X_{n+1}=x)$. APS instead measures the deviation from the oracle procedure required to achieve the desired level of coverage on the calibration data; the conformal score is defined as

$$S(x,y)=\sum_{j=1}^{k}\hat{f}(x)_{(j)},\text{ where } y=k. \tag{2}$$

To give a concrete example, suppose we want to construct prediction sets that contain the true label 90% of the time (so $\alpha=0.1$). It could be the case that, on our held out data, if we simply add up the ordered softmax outputs until their cumulative sum exceeds 0.9 we actually get 85% coverage, due to the model being miss-calibrated. Using APS we might calculate that if we construct prediction sets using $1-\hat{\alpha}=0.94$ we get 90% coverage, and by exchangeability this translates to 90% coverage on any new test point. We would therefore use the level $\hat{\alpha}=0.06$ to construct our new prediction sets.

3.2. Beyond Exchangeability

Conformal prediction in the form presented above relies on the assumption that the data points $Z_i=(X_i,Y_i)$ are exchangeable. The exchangeable form of conformal prediction provides no guarantee if these assumptions are violated, however *non-exchangeable conformal prediction* was introduced in the pioneering work of Barber et al. (2022).

Formally, the non-exchangeable conformal prediction procedure assumes a choice of deterministic fixed weights $w_1,\ldots,w_n\in[0,1]$ (normalized as detailed in (Barber et al., 2022)). As before, one computes the scores $s_1,\ldots,s_n$ but now defines the prediction set in terms of the *weighted quantiles* of the score distribution

$$\begin{aligned}&\widehat{C}_n\left(X_{n+1}\right)\\&=\ \{y\in\mathcal{Y}:S\left(X_{n+1},y\right)\leq\\&\qquad \mathrm{Q}_{1-\alpha}\left(\sum_{i=1}^{n}w_i\cdot\delta_{s_i}+w_{n+1}\cdot\delta_{+\infty}\right)\Bigg\}\end{aligned}\tag{3}$$

where $\mathrm{Q}_{\tau}(\cdot)$ denotes the τ-quantile of a distribution and δ_x denotes a point mass at x.

Non-exchangeable conformal prediction also comes with performance guarantees; the authors define the *coverage gap*

$$\text{Coverage gap }=(1-\alpha)-\mathbb{P}\left\{Y_{n+1}\in\widehat{C}_n\left(X_{n+1}\right)\right\} \tag{4}$$

as the loss of coverage when compared to the exchangeable setting, and show that this can be bounded as follows: let $Z=((X_1,Y_1),\ldots,(X_{n+1},Y_{n+1}))$ be the full dataset and define Z^i as the same dataset after swapping the test point and the i^{th} training point

$$\begin{aligned}Z^i=(&(X_1,Y_1),\ldots,(X_{i-1},Y_{i-1}),(X_{n+1},Y_{n+1}),\\&(X_{i+1},Y_{i+1}),\ldots,(X_n,Y_n),(X_i,Y_i))).\end{aligned}$$

Then the coverage gap in Equation (4) can be bounded as (Theorem 2a, Barber et al. (2022)):

$$\text{Coverage gap }\leq\frac{\sum_{i=1}^{n}w_i\cdot\mathrm{d}_{TV}\left(Z,Z^i\right)}{1+\sum_{i=1}^{n}w_i} \tag{5}$$

where d_{TV} is the total variation distance. To make this bound small one would like to place a large weight w_i on datapoints that are drawn from a similar distribution to the test point (X_{n+1},Y_{n+1}).

4. Conformal Prediction for Node Classification

Consider now the node classification setting: we are given a graph $G=(V,E)$, and for each node $i\in V$ we are given a node feature vector $X_i\in\mathbb{R}^F$ and a label $Y_i\in\mathcal{Y}$. A standard pipeline for node classification usually consists of a GNN model that produces a node embedding $h_i\in\mathbb{R}^H$ followed by a classifier $f:\mathbb{R}^H\to\mathcal{Y}$.

Here the data points $Z_i=(X_i,Y_i)$ are certainly not assumed to be exchangeable; the underlying principle of GNN models is that the adjacency matrix of G provides information about the dependency between datapoints (and hence neighbourhood information of G is aggregated and used for prediction). Barber et al. (2022) show in particular that non-exchangeable data can be navigated when the fitted model is a symmetric function of the test data. Our method is based on the observation that using only training data to fit the model trivially satisfies this assumption. In particular, this excludes the transductive setting.

We combine non-exchangeable conformal prediction with the information given by the adjacency matrix to produce an algorithm for constructing prediction sets for node classification, which we call Neighbourhood Adaptive Prediction

Table 1. Statistics for the Flickr, Reddit2 and Amazon Computers datasets, with notation as in Subsection 6.3.

Dataset	Nodes	Edges	# Feat	# Classes	# Test Nodes	$\lvert\mathcal{N}^{cal}\rvert$	$\hat{H}$	H_{rand}
Flickr	89,250	899,756	500	7	22313	5161	0.319	0.266
Reddit2	232,965	23,213,838	602	41	55334	22160	0.812	0.051
Amazon Comp.	13752	491,722	767	10	12000	11033	0.785	0.208

Sets (NAPS). The first variant simply localises the calibration to a neighbourhood of the network; we set the weights in Equation (3) to $w_i = 1$ if $i \in \mathcal{N}^K_{n+1}$, where $\mathcal{N}^K_{n+1}$ is the K-hop neighbourhood of node $n+1$. We then apply non-exchangeable conformal prediction with the APS scoring function in Equation (2).

The coverage gap of NAPS is bounded as

$$\text{Coverage gap} \;\leq\; \frac{\sum_{i\in\mathcal{N}^k_{n+1}} \mathrm{d}_{TV}\left(Z, Z^i\right)}{1+|\mathcal{N}^k_{n+1}|} \tag{6}$$

by simple substitution into Equation (5). This bound will be small if the k-hop neighbours of node $n+1$ are distributed similarly, which is otherwise known as *homophily* (McPherson et al., 2001).

Homophily is a key principle of many real world networks, where linked nodes often belong to the same class and have similar features, and is in crucial for good performance in many popular GNN architectures (although recent work has considered the heterophilic case, see (Zhu et al., 2021), which we will discuss in the future work section). This is also related to network *homogeneity*, where nodes in a neighbourhood play similar roles in the network and are considered interchangeable on average.

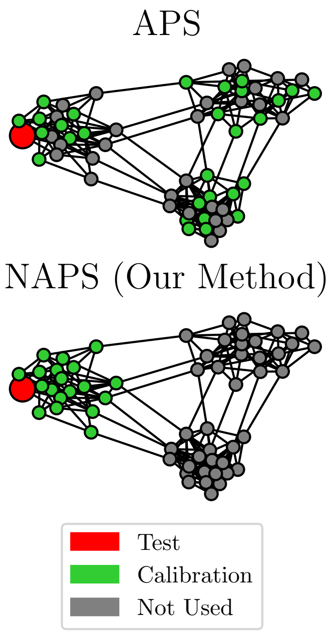

Figure 1. An illustration of the nodes used for calibrating conformal prediction via an ”out the box” application of APS (top panel), which randomly splits the data, and the nodes used in NAPS (bottom panel). NAPS localises the calibration nodes to a neighbourhood of the test node.

Motivated by the principle of homophily, we also explore two variants of NAPS that place more weight on closer neighbours than those further away. Formally we introduce a weighting function $w(k)$ setting $w_i = w(k)$ for a node i that is k hops from the test node $n+1$. In this setting, letting $\mathcal{N}^{k'}_{n+1}$ represent the nodes that are exactly k hops from node $n+1$, similarly to (6) we have the coverage gap bound

$$\text{Coverage gap} \;\leq\; \frac{\sum_{k=1}^{K} w(k) \sum_{i\in\mathcal{N}^{k'}_{n+1}} \mathrm{d}_{TV}\left(Z, Z^i\right)}{1+\sum_{k=1}^{K} w(k)|\mathcal{N}^{k'}_{n+1}|}. \tag{7}$$

The first variant, which we call NAPS-H, uses a hyperbolic decay rate $w(k) = k^{-1}$, and the second, which we call NAPS-G, uses a geometric decay rate $w(k) = 2^{-k}$.

Note that NAPS is not applicable in the transductive setting, as the fitted model $\hat{f}$ would depend on the node features in the test set, hence the conformal scores would no longer be exchangeable. It is however applicable in inductive settings where either the test set consists of multiple new graphs, or new nodes are added to an existing network.

The neighbourhood depth parameter K introduces a trade-off; expanding the neighbourhood increases the sample size for calibration, but introduces nodes that may be progressively less topologically similar. In the form presented here we recommend only applying NAPS to large homophilous networks with dense 1 or 2 hop neighbourhoods, but we will discuss extensions in future work.

5. A Case Study: The Stochastic Block Model

To gain further insight into when we might expect NAPS to outpeform APS, we perform a theoretical study inspired by the Stochastic Block Model (SBM), a parametric model from network science. Suppose we have a graph on $2n$ nodes partitioned into two distinct groups, C_1 and C_2. We introduce two parameters, p_{in}, the probability of connecting

Table 2. The test accuracy, empirical coverage, average prediction set size and average prediction set size conditional on coverage for all models considered on the Reddit2, Flickr and Amazon Computers datasets with $\alpha = 0.1$. Each column shows the median-of-means computed over 100 repetitions of the experiment, where each experiment is evaluated on a set of 1000 nodes. Bold indicates the best performing method.

Dataset	Model	Accuracy	Coverage				Size				Size | Coverage			
		Top-1	APS	NAPS	NAPS-L	NAPS-G	APS	NAPS	NAPS-L	NAPS-G	APS	NAPS	NAPS-L	NAPS-G
Reddit2	GS-Mean	0.914	0.928	0.895	0.896	0.899	2.02	**1.59**	**1.59**	**1.59**	2.12	**1.72**	1.73	1.73
	GS-Max	0.771	0.918	0.903	0.904	0.906	3.97	**3.20**	**3.20**	3.21	3.82	**3.30**	3.31	3.31
	SD-SAGE	0.844	0.925	0.896	0.899	0.897	2.08	1.64	**1.63**	1.64	2.11	1.69	**1.67**	1.68
	SD-GCN	0.827	0.927	0.896	0.898	0.899	2.11	1.66	**1.64**	1.65	2.14	1.73	**1.70**	1.71
Flickr	GS-Mean	0.503	0.915	0.904	0.909	0.910	4.22	**4.01**	4.10	4.12	4.26	**4.06**	4.17	4.19
	GS-Max	0.501	0.910	0.903	0.907	0.909	4.32	**4.11**	4.21	4.23	4.34	**4.14**	4.24	4.27
	SD-SAGE	0.500	0.912	0.904	0.907	0.908	4.27	**4.07**	4.15	4.17	4.31	**4.11**	4.21	4.23
	SD-GCN	0.496	0.912	0.903	0.908	0.909	4.28	**4.09**	4.18	4.20	4.33	**4.12**	4.22	4.25
Amazon	GS-Mean	0.854	0.905	0.902	0.902	0.904	1.50	**1.44**	**1.44**	1.46	1.57	1.50	**1.49**	1.52
	GS-Max	0.765	0.902	0.903	0.902	0.902	2.15	**1.98**	2.01	2.01	2.18	**2.04**	2.06	2.07
	SD-SAGE	0.815	0.912	0.905	0.905	0.906	1.77	**1.66**	1.67	1.67	1.80	**1.72**	1.74	1.73
	SD-GCN	0.822	0.911	0.904	0.905	0.904	1.75	1.64	**1.63**	1.64	1.82	1.72	**1.71**	1.73

two nodes in the same group, and p_{out}, the probability of connecting two nodes in different groups (usually $p_{out} << p_{in}$). We assume for nodes $i \in C_1$ that $(X_i, Y_i) \sim \pi_1$ i.i.d for all $i \in C_1$ where π_1 is a probability distribution, and that for nodes $j \in C_2$ that $(X_j, Y_j) \sim \pi_2$ i.i.d where $\pi_2 \neq \pi_1$, a different probability distribution. Importantly, we assume the group membership of each node is not observable, only implicit in the model specification.

Assume we have already fit a classifier $\hat{f}$ on a distinct training set, and our test network follows the block-model structure described above. The goal is to construct prediction sets for each node in the test set. If the group membership for each node was known, the problem would be easy; one could simply calibrate amongst only nodes in the same community. These nodes are drawn i.i.d from the same probability distribution, hence are exchangeable and so the standard guarantees for conformal prediction hold. As community membership is not observable, one must use the edges to determine whether two nodes are likely to be in the same group or not. Intuitively if $p_{in} >> p_{out}$ then on average linked nodes are much more likely to be in the same community, hence if there are enough neighbours to ensure sufficient sample size one would want to calibrate amongst the neighbours.

We now provide a theoretical result quantifying this intuition, the proof of which is given in Appendix A.1.

Lemma 5.1. *Assume the test data has the block model structure described above. Let CG_{APS} be the coverage gap for prediction sets constructed using APS (i.e. calibrating using all available nodes), and CG_{NAPS} be the coverage gap attained by the unweighted variant of NAPS calibrated amongst the 1-hop neighbours of each test node. Then $\mathbb{E}[CG_{NAPS}] < \mathbb{E}[CG_{APS}]$ if*

$$\mathbb{E}\left[\frac{N_{out}}{N_{out} + N_{in} + 1}\right] < \frac{1}{2} \tag{8}$$

where $N_{in} \sim Bin(n-1, p_{in})$, $N_{out} \sim Bin(n, p_{out})$ are Binomial random variables.

Note if we mean field approximate the expectation in Equation (8) we obtain a more intuitive expression in terms of the data generating parameters, namely that $\mathbb{E}[CG_{\text{NAPS}}] < \mathbb{E}[CG_{\text{APS}}]$ (approximately) when

$$\frac{np_{out}}{np_{out} + (n-1)p_{in} + 1} < \frac{1}{2}.$$

.

6. Experiments

We now perform experiments with popular real world datasets and models to evaluate the performance of our procedure. Our experiments follow the following format: we split each graph into training, validation and test nodes (where the validation and test nodes are not available during model fitting i.e. an inductive node split). The training and validation sets are used for model fitting, and the test set is used to evaluate the conformal prediction procedure by constructing prediction sets and evaluating the empirical coverage. Details of the implementation are found in Appendix A.3.

6.1. Evaluating Conformal Prediction

The observed coverage in a single application of conformal prediction is a *random* quantity, where the randomness comes from the choice of which data points are used for calibration as well as the finite sample size of the calibration set (corresponding to the upper bound in Equation (1)). It is therefore important to pick a large enough number of calibration points, and also repeat the experiment many times with different calibration/evaluation splits.

For simplicity we follow the guidelines given in Angelopoulos and Bates (2021), which suggest using at least 1000 validation points, and we repeat each experiment 100 times with a different calibration/evaluation split; with this setup by the law of large numbers the probability of observing significant deviations from the true coverage is extremely low, and therefore we can evaluate the performance of our method with high confidence.

Conformal prediction in the exchangeable setting is usually deployed by splitting the data into a calibration set and an evaluation set. The calibration points are used to estimate the quantile of the score distribution, which is used to construct prediction sets for each evaluation point. In our setting, this corresponds to selecting the calibration and evaluation nodes randomly, which ignores the graph structure. The goal of our experimental setup is to study the improvement in the performance of conformal prediction when the graph structure is taken into account.

6.2. Experimental Setup

In each experiment, we sample a batch of evaluation nodes and construct a $1 - \alpha$ probability prediction set for each evaluation node using NAPS as described in Section 4, as well as using a naive application of APS calibrated among all the nodes not in the evaluation set. We then report the empirical coverage, average prediction set size and average size of the prediction set given that the set contains the true label across all nodes.

For each experiment we sample 1000 nodes randomly from the nodes in the test set, and we perform 100 repetitions of the experiment (see Appendix 6.1 for a justification of this approach). We only apply our method to large connected components from the test set following the discussion in Section 4 (see Appendix A.2 for details on the datasets and the test set construction procedure).

6.3. Datasets and Models

We apply our method to three popular node classification datasets, namely Reddit2 and Flickr introduced in (Zeng et al., 2020) and Amazon Computers introduced in (Shchur et al., 2018). We apply two variants of two popular GNN models, namely GraphSAGE (Hamilton et al., 2017) with the mean and max aggregators, and the ShaDow (Zeng et al., 2021) subgraph sampling scheme with GraphSAGE and GCN (Kipf and Welling, 2017) layers.

NAPS relies on node homophily to minimize the coverage gap bound in Equation (5). Here we verify here that each of these networks is homophilous. We measure this via the node homophily ratio defined in (Pei et al., 2020) as

$$H = \frac{1}{|\mathcal{V}|} \sum_{v \in \mathcal{V}} \frac{|\{(w, v) : w \in \mathcal{N}(v) \wedge y_v = y_w\}|}{|\mathcal{N}(v)|}.$$

We define a homophilous network as one that has node homophily ratio larger than expected under a random assignment of labels. For a network with K classes, assume each node is assigned class k independently with probability p_k. Then for any $(v, w) \in \mathcal{V}$, we have

$$\mathbb{P}(y_v = y_w) = \sum_{k=1}^{K} p_k^2.$$

It follows that the expected homophily ratio under random class assignment is

$$\mathbb{E}[H] = \frac{1}{|\mathcal{V}|} |\mathcal{V}| \sum_{k=1}^{K} p_k^2 = \sum_{k=1}^{K} p_k^2.$$

In Table 1 we report both the observed homophily ratio $\hat{H}$ computed over the induced subgraph of the test nodes for each network as well as the expected node homophily under a random assignment of the labels H_{rand}, using the relative node label frequencies as the probabilities p_k. We see that Reddit2 and Amazon Computers are strongly homophilous, while Flickr is relatively weakly so.

The results for each dataset are displayed in Table 4. We see across all models on all three datasets, NAPS produces well calibrated, tight prediction sets, while the naive application of APS tends to overcover and produces wider prediction sets. The outperformance of NAPS over APS on the Flickr dataset shows that strong homophily is not required. We also see that this outperformance persists over a range of different GNN architectures. While the three NAPS variants perform similarly on Reddit2 and Amazon, NAPS outperforms the weighted variants by a significant margin on the Flickr dataset. This is likely due to the relatively weak node homophily ratio, implying that the increase in effective sample size is more valuable than down-weighting more distant neighbours.

7. Conditional Coverage and Network Topology

In this section we will examine the conditional coverage properties of NAPS via two metrics, one generic and one specific to the graph setting. A set-valued predictor $\mathcal{C} : \mathcal{X} \to 2^{\mathcal{Y}}$ is said to satisfy *exact conditional coverage* if

$$\mathbb{P}(Y \in \mathcal{C}(X) | X = x) = 1 - \alpha \text{ for all } x \in \mathcal{X}.$$

Table 3. The SSCV and PCCV (as described in Equations (9) and (10)) for all models considered on the Reddit2, Flickr and Amazon Computers datasets with $\alpha = 0.1$. Bold indicates the best performing method. All the NAPS variants use depth parameter $K = 2$.

Dataset	Model	SSCV				PCCV			
		APS	NAPS	NAPS-H	NAPS-G	APS	NAPS	NAPS-H	NAPS-G
Reddit2	GraphSAGE-Mean	0.087	0.074	**0.072**	0.074	0.084	**0.052**	0.054	0.054
	GraphSAGE-Max	0.089	0.075	**0.073**	0.076	0.083	0.052	**0.051**	0.053
	ShaDow-SAGE	0.075	0.056	**0.053**	0.055	0.087	0.054	**0.051**	0.055
	ShaDow-GCN	0.078	0.056	**0.054**	0.056	0.089	0.053	**0.052**	0.053
Flickr	GraphSAGE-Mean	0.113	**0.064**	0.074	0.079	0.080	**0.059**	0.065	0.065
	GraphSAGE-Max	0.102	**0.062**	0.073	0.075	0.084	**0.061**	0.067	0.067
	ShaDow-SAGE	0.100	**0.054**	0.085	0.087	0.089	**0.064**	0.066	0.067
	ShaDow-GCN	0.105	**0.054**	0.079	0.081	0.091	**0.062**	0.069	0.071
Amazon	GraphSAGE-Mean	0.082	0.073	**0.072**	0.079	0.076	0.046	0.042	**0.041**
	GraphSAGE-Max	0.081	**0.072**	**0.072**	0.074	0.078	0.045	**0.042**	**0.042**
	ShaDow-SAGE	0.065	0.057	0.058	**0.052**	1.03	0.056	0.053	**0.050**
	ShaDow-GCN	0.066	0.055	**0.054**	0.055	0.99	0.051	0.052	**0.049**

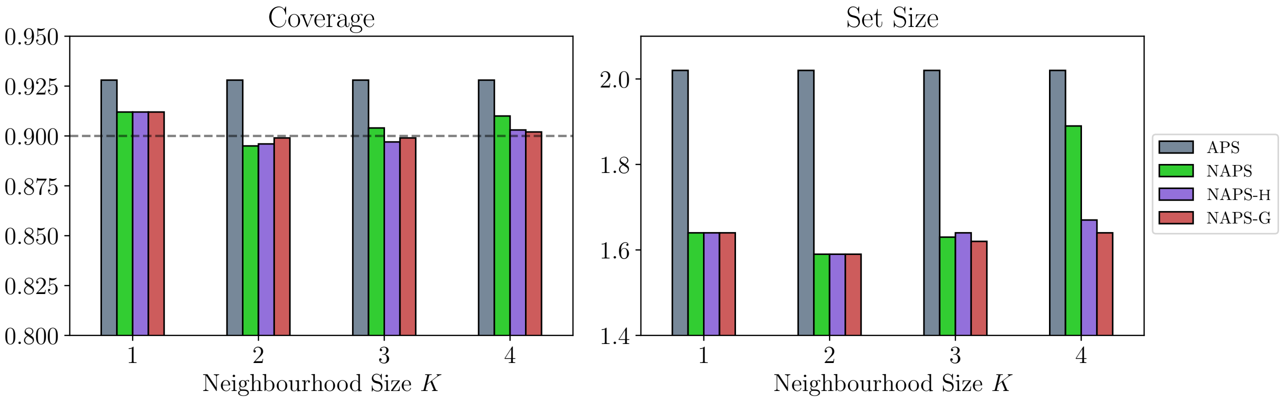

Figure 2. The coverage and size for the different conformal prediction procedures whilst varying K on the Reddit2 dataset. All the experiments use the GraphSAGE-Mean GNN architecture. Each column shows the median of means over 100 repetitions of the experiment for the given method using the methodology introduced in Section 6.2. The dashed black line shows the desired coverage level $\alpha = 0.9$.

Exact Conditional coverage is known to be impossible to achieve for conformal prediction (Barber et al., 2019); it is nonetheless desirable to achieve approximate conditional coverage.

Firstly we consider a metric introduced specifically for classification problems in (Angelopoulos et al., 2020), namely Size-Stratified Coverage Violation (SSCV) metric. Let $\mathcal{S} = \{S_i\}_{i=1}^{i=s}$ be a disjoint stratification of the possible prediction set sizes such that $\bigcup_{j=1}^{s} S_i = \{1, \ldots, |\mathcal{Y}|\}$. Define the index set of all the datapoints with prediction set size falling into stratum S_j as

$$\mathcal{I}_j = \{i : |\mathcal{C}(X_i, Y_i, U_i)| \in S_j\}.$$

Then the SSCV is defined as

$$\mathrm{SSCV}\left(\mathcal{C}, \mathcal{S}\right) := \sup_{\mathrm{j}} \left| \frac{\left|\{\mathrm{i} : \mathrm{Y_i} \in \mathcal{C}\left(\mathrm{X_i}, \mathrm{Y_i}, \mathrm{U_i}\right), \mathrm{i} \in \mathcal{I}_{\mathrm{j}}\}\right|}{|\mathcal{I}_{\mathrm{j}}|} - (1 - \alpha) \right|. \quad (9)$$

Here prediction set size can be thought of as a proxy for the difficulty of the sample, so intuitively this measures the worst case coverage violation conditioned on the difficulty of the example.

In the graph setting, a desirable property for a conformal prediction method is that the coverage probability for a given node does not depend on its location within the graph. To give a trivial example of why this is important, suppose our test network consists of 100 nodes split into two communities, with 90 nodes in one community and 10 nodes in the other. To achieve 90% coverage, a conformal prediction method could predict the full label set on all nodes in the first community and the empty set on all the nodes in the second community, yet this would not be helpful in practice.

To measure the degree to which this issue is present in the constructed prediction sets, we introduce the Partition Conditional Coverage Violation (PCCV) metric. Given a partitioning of the node set into disjoint partitions $\mathcal{V} = \{\mathcal{V}_i\}_{i=1}^{N}$, we define the PCCV as

$$\mathrm{PCCV}\left(\mathcal{C}, \mathcal{V}\right) := \sup_{\mathrm{j}} \left| \frac{\left|\{\mathrm{i} : \mathrm{Y_i} \in \mathcal{C}\left(\mathrm{X_i}, \mathrm{Y_i}, \mathrm{U_i}\right), \mathrm{i} \in \mathcal{V}_{\mathrm{j}}\}\right|}{|\mathcal{V}_{\mathrm{j}}|} - (1-\alpha) \right|. \tag{10}$$

We estimate the SSCV using the experimental setup introduced in Section 6.2. We modify the experimental setup slightly for computing PCCV, instead taking half of the nodes (chosen at random) as calibration data and the other half as test data. We partition the data into disjoint neighbourhoods using a recursive approach; given the current set of test nodes, we choose a root node at random and take the 2-hop neighbourhood around it, then delete all the nodes in the neighbourhood from the current set and repeat until there are no neighbourhoods of size 30 nodes or more. We repeat this procedure 100 times and report the median over the different runs.

The experimental results are shown in Table 3. All NAPS variants outperform APS on all data sets and models. Within the NAPS variants, the linear weights tend to perform best on the Reddit2 data set, and the geometric weights perform very well on the Amazon dataset, while the unweighted NAPS version performs well on all data sets.

8. Ablation Study

The neighbourhood depth parameter K reflects a trade-off between sample size and similarity between datapoints. To better understand the influence of neighbourhood depth on the performance of NAPS, we repeat the Reddit2 experiment using the same setup introduced in Section 6.2 for $K = 1, \ldots, 4$.

The results for the GraphSAGE-Mean architecture are displayed in Figure 2. We see that for $K = 1$ (on which all three NAPS variants are equal) our method slightly overcovers, likely due to the low sample size. We see that as K increases (i.e. the neighbourhoods become large) the performance of the unweighted variant of NAPS tends towards that of APS as expected. The weighted variants are better able to manage the trade-off between sample size and relevance, maintaining similar rates of coverage and prediction set size as the neighbourhood size grows.

9. Conclusion and Future Work

In this work we have introduced NAPS, an approach for constructing prediction sets on graph structured data. NAPS is quick to train and deploy and comes with theoretical guarantees on the coverage. We have shown through extensive experiments that NAPS produces prediction sets that are both more efficient and have better conditional coverage properties than a naive application of conformal prediction.

In this work we have treated the neighbourhood size K as a hyper-parameter. Intuitively, one would like to select this value such that the calibration nodes are ”local” within the network while providing a large enough sample size to accurately estimate the quantile $1 - \hat{\alpha}$. It would be useful to further study the interplay between the optimal choice of K, the homophily level and the diameter of the network. NAPS could also be extended to heterophilic networks; in a heterophilic network nodes tend to be connected to dis-similar nodes. One could therefore calibrate among alternating neighbourhoods $\bigcup_{j=1}^{k} \mathcal{N}_{n+1}^{2j} \setminus \mathcal{N}_{n+1}^{2j-1}$.

NAPS may produce wide prediction sets when deployed on low density networks as the sample size for conformal calibration will be small, see Equation (1). An approach for conformal prediction in hierachical models was introduced in Dunn et al. (2022), where the quantiles are calibrated in different groups before being pooled. An approach similar to this could be applied for nodes in small connected components, where calibration on similar neighbourhoods or components could be pooled to provide a better estimate of the conformal quantile.

Finally, NAPS could be extended to model other types of graph structured data. Here we have only considered unweighted and undirected networks, but addressing both weighted and directed networks follow as natural extensions to the method. It would also be fairly straightforward to extend the method to node *regression* tasks simply by using a different conformal score function such as CQR (Romano et al., 2019).

10. Ethical Concerns

While we do not believe this work is likely to have any negative impact on society, as always with statistical methods care must be taken with the interpretation of the results. Just like any method that constructs prediction intervals, those obtained by NAPS indicate a statistical prediction, and as such if used in a high-stakes field such as healthcare must be interpreted as such.

References

Matthias Fey and Jan E. Lenssen. Fast graph representation learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019.

Minjie Wang, Da Zheng, Zihao Ye, Quan Gan, Mufei Li, Xiang Song, Jinjing Zhou, Chao Ma, Lingfan Yu, Yu Gai, Tianjun Xiao, Tong He, George Karypis, Jinyang Li, and Zheng Zhang. Deep graph library: A graph-centric, highly-performant package for graph neural networks. *arXiv preprint arXiv:1909.01315*, 2019.

Vladimir Vovk, Alex Gammerman, and Glenn Shafer. *Algorithmic Learning in a Random World*. Springer-Verlag, Berlin, Heidelberg, 2005. ISBN 0387001522.

Anastasios N Angelopoulos, Stephen Bates, Jitendra Malik, and Michael I Jordan. Uncertainty sets for image classifiers using conformal prediction. *arXiv preprint arXiv:2009.14193*, 2020.

Rina Foygel Barber, Emmanuel J. Candes, Aaditya Ramdas, and Ryan J. Tibshirani. Conformal prediction beyond exchangeability, 2022. URL https://arxiv.org/abs/2202.13415.

Ethan Goan and Clinton Fookes. Bayesian neural networks: An introduction and survey. In *Case Studies in Applied Bayesian Data Science*, pages 45–87. Springer International Publishing, 2020. doi: 10.1007/978-3-030-42553-1_3. URL https://doi.org/10.1007%2F978-3-030-42553-1_3.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in Neural Information Processing Systems*, 30, 2017.

Max Welling and Diederik P Kingma. Auto-encoding variational bayes. *ICLR*, 2014.

Arman Hasanzadeh, Ehsan Hajiramezanali, Shahin Boluki, Mingyuan Zhou, Nick Duffield, Krishna Narayanan, and Xiaoning Qian. Bayesian graph neural networks with adaptive connection sampling. In *International conference on machine learning*, pages 4094–4104. PMLR, 2020.

Rohitash Chandra, Ayush Bhagat, Manavendra Maharana, and Pavel N Krivitsky. Bayesian graph convolutional neural networks via tempered mcmc. *IEEE Access*, 9: 130353–130365, 2021.

Yingxue Zhang, Soumyasundar Pal, Mark Coates, and Deniz Ustebay. Bayesian graph convolutional neural networks for semi-supervised classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 5829–5836, 2019.

David Stutz, Krishnamurthy Dj Dvijotham, Ali Taylan Cemgil, and Arnaud Doucet. Learning optimal conformal classifiers. In *International Conference on Learning Representations*, 2022.

Bat-Sheva Einbinder, Yaniv Romano, Matteo Sesia, and Yanfei Zhou. Training uncertainty-aware classifiers with conformalized deep learning. In *Advances in Neural Information Processing Systems*, 2022.

Jiaye Teng, Chuan Wen, Dinghuai Zhang, Yoshua Bengio, Yang Gao, and Yang Yuan. Predictive inference with feature conformal prediction. *arXiv preprint arXiv:2210.00173*, 2022.

Yaniv Romano, Matteo Sesia, and Emmanuel Candes. Classification with valid and adaptive coverage. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 3581–3591. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/244edd7e85dc81602b7615cd705545f5-Paper.pdf.

Mauricio Sadinle, Jing Lei, and Larry Wasserman. Least ambiguous set-valued classifiers with bounded error levels. *Journal of the American Statistical Association*, 114 (525):223–234, 2019.

Jing Lei, Max G'Sell, Alessandro Rinaldo, Ryan J Tibshirani, and Larry Wasserman. Distribution-free predictive inference for regression. *Journal of the American Statistical Association*, 113(523):1094–1111, 2018.

Yaniv Romano, Evan Patterson, and Emmanuel Candes. Conformalized quantile regression. *Advances in Neural Information Processing Systems*, 32, 2019.

Anastasios N. Angelopoulos, Stephen Bates, Adam Fisch, Lihua Lei, and Tal Schuster. Conformal risk control, 2022. URL https://arxiv.org/abs/2208.02814.

Isaac Gibbs and Emmanuel Candes. Adaptive conformal inference under distribution shift. *Advances in Neural Information Processing Systems*, 34:1660–1672, 2021.

Margaux Zaffran, Olivier Féron, Yannig Goude, Julie Josse, and Aymeric Dieuleveut. Adaptive conformal predictions for time series. In *International Conference on Machine Learning*, pages 25834–25866. PMLR, 2022.

Isaac Gibbs and Emmanuel Candès. Conformal inference for online prediction with arbitrary distribution shifts, 2022. URL https://arxiv.org/abs/2208.08401.

Osbert Bastani, Varun Gupta, Christopher Jung, Georgy Noarov, Ramya Ramalingam, and Aaron Roth. Practical adversarial multivalid conformal prediction. In *Advances in Neural Information Processing Systems*, 2022.

Harris Papadopoulos, Kostas Proedrou, Volodya Vovk, and Alex Gammerman. Inductive confidence machines for regression. volume 2430, pages 185–194, 08 2002. ISBN 978-3-540-44036-9. doi: 10.1007/3-540-36755-1_29.

Anastasios N. Angelopoulos and Stephen Bates. A gentle introduction to conformal prediction and distribution-free uncertainty quantification, 2021. URL https://arxiv.org/abs/2107.07511.

Miller McPherson, Lynn Smith-Lovin, and James M Cook. Birds of a feather: Homophily in social networks. *Annual Review of Sociology*, 27(1):415–444, 2001. doi: 10.1146/annurev.soc.27.1.415. URL https://doi.org/10.1146/annurev.soc.27.1.415.

Jiong Zhu, Ryan A Rossi, Anup Rao, Tung Mai, Nedim Lipka, Nesreen K Ahmed, and Danai Koutra. Graph neural networks with heterophily. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 11168–11176, 2021.

Hanqing Zeng, Hongkuan Zhou, Ajitesh Srivastava, Rajgopal Kannan, and Viktor Prasanna. GraphSAINT: Graph sampling based inductive learning method. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=BJe8pkHFwS.

Oleksandr Shchur, Maximilian Mumme, Aleksandar Bojchevski, and Stephan Günnemann. Pitfalls of graph neural network evaluation, 2018. URL https://arxiv.org/abs/1811.05868.

Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper/2017/file/5dd9db5e033da9c6fb5ba83c7a7ebea9-Paper.pdf.

Hanqing Zeng, Muhan Zhang, Yinglong Xia, Ajitesh Srivastava, Andrey Malevich, Rajgopal Kannan, Viktor Prasanna, Long Jin, and Ren Chen. Decoupling the depth and scope of graph neural networks. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=d0MtHWY0NZ.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*, 2017.

Hongbin Pei, Bingzhe Wei, Kevin Chen-Chuan Chang, Yu Lei, and Bo Yang. Geom-gcn: Geometric graph convolutional networks. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=S1e2agrFvS.

Rina Foygel Barber, Emmanuel J. Candès, Aaditya Ramdas, and Ryan J. Tibshirani. The limits of distribution-free conditional predictive inference, 2019. URL https://arxiv.org/abs/1903.04684.

Robin Dunn, Larry Wasserman, and Aaditya Ramdas. Distribution-free prediction sets for two-layer hierarchical models. *Journal of the American Statistical Association*, 0(0):1–12, 2022. doi: 10.1080/01621459.2022.2060112. URL https://doi.org/10.1080/01621459.2022.2060112.

Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 12 2014.

A. Appendix

A.1. Proof of Lemma 5.1

We define $d := d_{TV}(\pi_1, \pi_2)$ as the total variation distance between the data distributions for the two communities.

A naive application of APS ignores the link structure, applying weight $w_j = 1$ to all nodes. For whichever node the prediction set is being constructed, n nodes will belong to the opposing community (hence occur a penalty of d in the Equation (5)) and $n-1$ nodes will belong to the same community so we may immediately apply Equation (5) to obtain

$$E[CG_{\text{APS}}] \leq \frac{nd}{(2n-1)+1} = \frac{d}{2} \tag{11}$$

Define $V_i := 1_{Y_i \in \hat{C}(X_i)}$ as the indicator function of the event that the i^{th} node is covered by the constructed prediction set. We can write the expected coverage for NAPS as

$$\mathbb{E}[C_{\text{NAPS}}] = \mathbb{E}\left[\frac{1}{2n}\sum_{i=1}^{2n} V_i\right] = \frac{1}{2n}\sum_{k=1}^{2n} \mathbb{E}[V_k].$$

We can write each term in this sum as

$$\mathbb{E}[V_k] = \sum_{i=1}^{n-1} \sum_{j=1}^{n} (\mathbb{E}[V_k \mid N_{in} = i, N_{out} = j] \\ Bin(i; n-1, p_{in}) Bin(j; n, p_{out})) \quad (12)$$

where N_{in}, N_{out} are random variables indicating the number of neighbours in the same community and opposing community to node k respectively, and $Bin(l; n, p)$ is the probability mass function of a Binomial random variable with parameters n and p evaluated at l.

The term $\mathbb{E}[V_k \mid N_{in} = i, N_{out} = j]$ is just the coverage of a prediction set constructed for node k in a fixed realisation of the network, so we may apply Equation (5) to obtain

$$\mathbb{E}[V_k \mid N_{in} = i, N_{out} = j] \geq (1 - \alpha) - \frac{jd}{i + j + 1}. \quad (13)$$

Lower bounding term-wise in Equation (12) we have that

$$\mathbb{E}[V_k] \geq (1 - \alpha) - d\mathbb{E}\left[\frac{N_{out}}{N_{out} + N_{in} + 1}\right] \quad (14)$$

Comparing the two bounds gives the result.

A.2. Dataset Details

For the experiments above we used the Flickr and Reddit2 datasets from (Zeng et al., 2020), and the Amazon Computers dataset introduced in (Shchur et al., 2018).

The Flickr dataset is constructed using images uploaded to the Flickr site, where the node features consist of the meta-data for each image and the label is the image tag. The Reddit2 dataset is constructed from posts on the social media site Reddit, with posts representing nodes. The node features are bag-of-word vectors from the post, and the label is the community (or sub-reddit) that the post belongs to. The Amazon Computers dataset consists of segments of an Amazon co-purchase graph, where nodes represent goods and links are added between nodes if they are frequently bought together.

Our train/validation/test splits for Flickr and Reddit2 were done using the splits given in the original papers (which are conveniently implemented in Pytorch Geometric (Fey and Lenssen, 2019)). For Amazon Computers we constructed our own split, using 752 nodes for training, 1000 for validation and the remaining 12000 for testing. As mentioned in the main text we tested our graph only on large connected components, which we chose as nodes with at least 50 2-hop neighbours in Flickr and Amazon Computers, and nodes with at least 1000 2-hop neighbours in Reddit2. We call this set of nodes $\mathcal{N}^{cal}$, and report the sizes of these sets as well as some summary statistics about each dataset in Table 1.

A.3. Model Training Details

We used the implementations of GraphSAGE and ShaDow provided by Pytorch Geometric (Fey and Lenssen, 2019). All models on all datasets used the same hyper-parameters. Each GNN used 2 layers with hidden dimension $H = 64$. We used the Adam optimiser (Kingma and Ba, 2014) with default hyper-parameters, learning rate $\eta = 0.1$, and used dropout probability $\delta = 0.5$. For the GraphSAGE neighbour sampling training we used 25 1-hop neighbours and 10 2-hop neighbours. We used early stopping based on the accuracy on the validation set. We made no effort to optimise any of these parameters as we are not trying to optimise for accuracy, we merely assess whether our method performs well with a variety of architectures.

Each experiment here took less than two hours in total on a single machine with an NVIDIA GeForce RTX 2060 SUPER GPU and an AMD Ryzen 7 3700X 8-Core Processor. One run of the conformal prediction procedure has trivial overhead when compared with model fitting (and actually NAPS is faster than APS as we use less data points to calibrate the procedure).