Uncertainty Quantification on Graph Learning: A Survey

CCS Concepts: • **Do Not Use This Code → Generate the Correct Terms for Your Paper**; *Generate the Correct Terms for Your Paper*; Generate the Correct Terms for Your Paper; Generate the Correct Terms for Your Paper.

Additional Key Words and Phrases: Do, Not, Us, This, Code, Put, the, Correct, Terms, for, Your, Paper

ACM Reference Format:
. 2018. Uncertainty Quantification on Graph Learning: A Survey. In . ACM, New York, NY, USA, **??** pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

2 PRELIMINARIES

In this section, we briefly introduce two key families of models for graphs, the Probabilistic Graphical Model and the Graph Neural Network.

2.1 Probabilistic Graphical Model

A probabilistic graphical model (PGM) [**? ?**] is widely used to represent the probability distribution of random variables, whose relations can be depicted in a graph. PGMs aim to minimize the cost of establishing compatible dependency relationships among all the variables. There are two distinct categories of PGMs, depending on whether the graph's edges are directed or undirected. Specifically, a Bayesian Network is employed for modeling a directed graph, where edges represent causality. Conversely, a Markov Random Field (MRF) is utilized to model an undirected graph, where edges signify the correlation between nodes. In an MRF, each node is conditionally independent of all other nodes, except for its immediate (e.g., 1-hop) neighbors.

Formally, we denote a graph $G = (V, E)$ consisting of a set of n random variables $V = \{X_1, \dots, X_n\}$ and E implies the relationship between any two variables. Each variable $X_i \in V$ may take values in $\{1, \dots, C\}$ where C is the number of classes. An MRF factories the joint distribution $P(V)$ as

$$P(V) = \frac{1}{Z} \prod_{X_i \in V} \phi(X_i = x_i) \prod_{X_j \in \mathcal{N}(X_i)} \psi(X_i = x_i, X_j = x_j), \tag{1}$$

where Z normalizes the product to a probability distribution. $\mathcal{N}(X_i)$ represents the neighbors of X_i within the graph G. $\phi(X_i = x_i)$ denotes the prior distribution of X_i taking on the value x_i, independent of other variables in the graph. The compatibility $\psi(X_i = x_i, X_j = x_j)$ encodes the likelihood of the pair (X_i, X_j) jointly taking the value (x_i, x_j), and capture the dependencies between variables. The marginal distribution $b(X_i)$, also known as belief, for any $X_i \in V$ is naturally given by

$$b(X_i = x_i) = \sum_{X_1} \cdots \sum_{X_{i-1}} \sum_{X_{i+1}} \cdots \sum_{X_n} P(X_1, X_2, \dots, X_i = x_i, \dots, X_n). \tag{2}$$

However, computing $b(X_i = x_i)$ for any node $X_i \in V$ by Eq. (**??**) is exponential complexity in the worst case. We show that Belief Propagation (BP) [**?**] infers the beliefs much more efficiently. First, the message $m_{i\to j}(X_j = x_j)$ from X_i to X_j is defined by

$$\frac{1}{Z_j}\sum_{X_i=x_i}\left[\psi(X_i = x_i, X_j = x_j)\phi(X_i = x_i)\prod_{k\in\mathcal{N}(X_i)\setminus\{X_j\}} m_{k\to i}(X_i = x_i)\right], \tag{3}$$

where Z_j is a normalization factor so that $m_{i\to j}$ is a probability distribution of X_j. The messages in both directions on all edges $(X_i, X_j) \in E$ will be updated iteratively. As the message updating converges (guaranteed when G is acyclic [**?**]). The belief $b(X_i = x_i)$ is given by

$$b(X_i = x_i) \propto \phi(X_i = x_i)\prod_{X_j\in\mathcal{N}(X_i)} m_{j\to i}(X_i = x_i). \tag{4}$$

For simplicity, we will omit X_i in $\phi(X_i = x_i)$ and use $\phi(x_i)$, and similarly for $\psi(x_i, x_j)$, $m_{j\to i}(x_i)$ and $b(x_i)$ if there is no ambiguity.

2.2 Graph Neural Networks

Graph Neural Network (GNN) is a deep learning-based method that article graphicx

UQML

RUI XU

CCS Concepts: • **Do Not Use This Code → Generate the Correct Terms for Your Paper**; *Generate the Correct Terms for Your Paper*; Generate the Correct Terms for Your Paper; Generate the Correct Terms for Your Paper;

Additional Key Words and Phrases: Do, Not, Us, This, Code, Put, the, Correct, Terms, for, Your, Paper

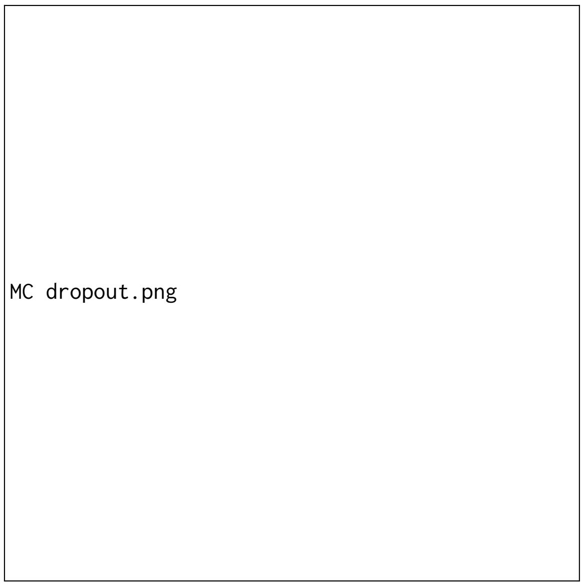
ACM Reference Format:
Rui Xu. 2018. UQML. In . ACM, New York, NY, USA, **??** pages. https://doi.org/XXXXXXX.XXXXXXX

3 REVIEWS

3.1 Calibrating Uncertainty Models for Steering Angle Estimation - 2019 IEEE

This paper reviews three main kinds of UQ methods in machine learning: Monte Carlo Dropout, Bootstrap Model, and Gaussian Mixture Model. Evaluation metrics are also introduced. Prediction accuracy, usually chosen as RMSE or MAE, is used to see if the predicted value is close to the training data. Calibration, on the other hand, is to measure if the predicted distribution is similar to the training data distribution.

3.1.1 Monte Carlo Dropout. To construct the dropout variant based on the baseline model, dropout layers are added after every fully connected layer but the last. Dropout is implemented by sampling a Bernoulli random variable with



probability p. Neuron k in layer l thus gets dropped with probability $z^{l;k} \sim$ Bernoulli(p).

Inference for an unknown sample x is performed by doing multiple stochastic forward passes and averaging the result. The randomness comes from generating new dropout weight matrices $\hat{W}_t$ from the Bernoulli distribution for every forward pass t. A forward pass with network weights $\hat{W}_t$ is denoted as $f^{\hat{W}_t}(x)$. The predicted mean steering angle can then be calculated as

$$E[\hat{y}] = \frac{1}{T}\sum_{t=1}^{T}\hat{y}_t = \frac{1}{T}\sum_{t=1}^{T} f^{\hat{W}_t}(x)$$

We can also get the predictive variance of the stochastic forward passes, seen as our uncertainty estimate, by calculating the variance over all T results plus the inverse model precision τ^{-1} as

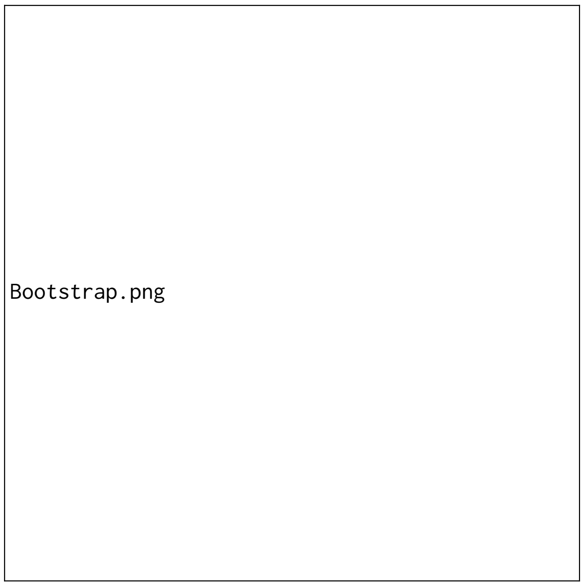
$$Var[\hat{y}] = \tau^{-1} + \frac{1}{T}\sum_{t=1}^{T}(f^{\hat{W}_t}(x))^2 - (E[\hat{y}])^2$$

The model precision τ^{-1} can be interpreted as label noise and thus representing the aleatoric uncertainty. Being a fixed constant it implies homoscedastic aleatoric uncertainty. We omit the model precision and use the variance of the outputs as predictive uncertainty.

3.1.2 Bootstrap Model. We would have to train models separately and make inferences in those networks K in parallel, we rely on a simplification instead: multiple models are fused into one model by sharing the visual encoder network.

The process of sampling K subsets of the training data for each subnetwork is done by generating a mask for every sample indicating which head it is passed to during training.

This ensures that every head sees only a subset of the whole training data. The data subsampling is realized by generating a binary mask $\{0, 1\}^K$ for every training sample indicating whether it is seen by head $k \in [1, K]$.



For sample t and head k, this results in variable $m_t^k \sim$ Bernoulli(p) with p = 0:5 specifying whether a head is trained on a certain sample.

The loss thus is defined as

$$L(\theta) = \sum_{i=1}^{N} \sum_{k=1}^{K} m_i^k \frac{1}{2} ||y_i - \hat{y}_i||^2$$

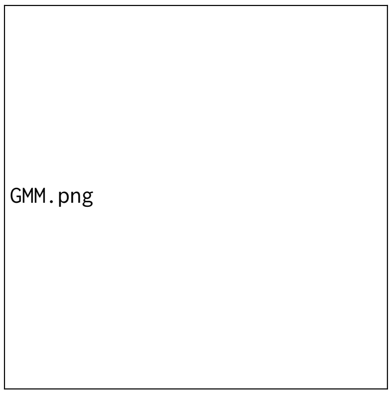
Inference is performed by forward-propagation through all heads resulting in K outputs. Note that only a single pass is required to get K outputs, instead of having to do K passes as in the dropout model. $f^k(x)$ is the prediction of head k for sample x.

$$E[\hat{y}] = \frac{1}{K} \sum_{k=1}^{K} f^k(x)$$

$$Var[\hat{y}] = \frac{1}{K} \sum_{k=1}^{K} (f^k(x))^2 - (E[\hat{y}])^2$$

Bootstrap model can not get aleatoric and epistemic uncertainty separated. Instead, it only yields the predictive variance which is used as predictive uncertainty. The inference step is computationally more expensive than inference in the baseline model as there are K more computations in the fully connected layer.

3.1.3 Gaussian Mixture Model. A Gaussian Mixture Model is composed of multiple Gaussians combined in a weighted sum. The parameters of a GMM are $\theta = \pi_i; \mu_i; \sigma_i^2; i \in [1, K]$ for K mixtures.



The mixture model is trained using the negative log-likelihood as a loss. A small constant $\epsilon = 1e - 6$ is added to the argument of the logarithm for numerical stability.

$$L(\theta) = -\frac{1}{N}\sum_{i=1}^{N} \log(p(y_i|x_i)) = \frac{1}{N}\sum_{i=1}^{N} \log(\sum_{j=1}^{K} \pi_j^{(i)} N(y_i|\mu_j^{(i)}, \sigma_j^{2(i)}) + \epsilon)$$

All mixture components are combined in a weighted sum and we get the following results for the expected value:

$$E[\hat{y}] = \sum_{j=1}^{K} \pi_j(x)\mu_j(x)$$

The total variance, in our context, called the predicted variance which is interpreted as predictive uncertainty, decomposes into the weighted sum of the variances and the weighted variances of the means,

$$Var[\hat{y}] = \sum_{j=1}^{K} \pi_j(x)\sigma_j^2(x) + \sum_{j=1}^{K} \pi_j(x)||\mu_j(x) - \sum_{k=1}^{K} \pi_k(x)\mu_k(x)||^2$$

The second term is also referred to as explained variance or epistemic uncertainty as it vanishes with more data. A decreasing aleatoric uncertainty can be interpreted as a decreasing variance of the mixture components. In turn, intuitively, a decreasing epistemic uncertainty equals to the means of the mixture components getting closer. However, this is only the case as long as there are no ambiguous situations as would be the case at intersections with multiple possible routes.

3.1.4 Prediction Accuracy. The prediction accuracy is usually chosen as root mean square error(RMSE) or mean absolute error(MAE) to measure the distance between the predicted values and true values.

3.1.5 Calibration. Intuitively, this means that the uncertainty estimations have to be a true probability and should reflect the true likelihood.

First, a z% confidence interval is computed for each sample in the data set based on the predicted mean and variance. The confidence interval (CI) specifies the lower and upper bound arranged symmetrically around the mean, containing z% of the given Gaussian distribution mass.

4 UNCERTAINTY QUANTIFICATION ON GRAPHS

4.1 Uncertainty Quantification in Probabilitic Graphical Models

4.2 Uncertainty Quantification in Graph Neural Networks

REFERENCES

[] Christopher M Bishop and Nasser M Nasrabadi. 2006. *Pattern recognition and machine learning*. Vol. 4. Springer.

[] Michael I Jordan. 2004. Graphical models. (2004).

[] Judea Pearl. 1988. *Probabilistic reasoning in intelligent systems: networks of plausible inference.* Morgan Kaufmann.

Received 20 February 2007; revised 12 March 2009; accepted 5 June 2009