Tokyo Institute of Technology

Learning GNNs with Noisy Labels

Hoang NT, Jun Jin Choong, Tsuyoshi Murata
Tokyo Institute of Technology

LEARNING WITH LIMITED LABELED DATA
ICLR 2019

Introduction

We study the robustness to symmetric label noise of GNNs training procedures. By combining the nonlinear neural message-passing models (e.g. Graph Isomorphism Networks, GraphSAGE, etc.) with loss correction methods, we present a noise-tolerant approach for the graph classification task. Our experiments show that test accuracy can be improved under the artificial symmetric noisy setting.

Bayesian Inference Model

We abbreviate the probability that a map cell m is occupied $p(m=1|x_*)$ as $p(m|x_*)$. As in standard occupancy grid mapping, the cell m is occupied with probability $p(m|x_*)$ and free with probability $1-p(m|x_*)$. That is, $m \sim Ber(\theta)$, with parameter $\theta = p(m|x_*)$. We seek to estimate the parameter θ. To do so, we use the nonparametric Bayesian inference model for exponential families in [1]. With $\theta \sim Beta(\alpha, \beta)$, the predicted mean and variance of θ are as follows:

$$\mathbf{E}[\theta] = \frac{\alpha}{\alpha+\beta} \tag{1}$$

$$\mathbf{Var}[\theta] = \frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)} \tag{2}$$

where α and β are hyperparameters. At each time step we can apply exact recursive updates to the hyperparameters

$$\alpha_t = \alpha_{t-1} + \bar{y} \tag{3}$$
$$\beta_t = \beta_{t-1} + \bar{k} - \bar{y} \tag{4}$$

in which $\bar{k}$ and $\bar{y}$ are the results of the following kernel computations:

$$\bar{y} = \sum_{i=1}^{N} k(x_i, x_*) y_i \tag{5}$$

$$\bar{k} = \sum_{i=1}^{N} k(x_i, x_*). \tag{6}$$

The quantities α_0 and β_0 represent prior pseudo-counts of the positive (occupied) and negative (free) classes respectively.

We also take advantage of variance predictions in a similar fashion to previous work [3]. We use the following model of state for cells in the environment

$$\text{state} = \begin{cases} \text{free}, & \text{if } p < p_{free}, \sigma^2 < \sigma^2_{th} \\ \text{occupied}, & \text{if } p > p_{occ}, \sigma^2 < \sigma^2_{th} \\ \text{unknown}, & \text{otherwise} \end{cases} \tag{7}$$

where $p = \mathbf{E}[\theta]$, and quantities p_{free} and p_{occ} are thresholds on the occupancy probability of cells deemed "free," and "occupied," respectively. The variance σ^2, is $\mathbf{Var}[\theta]$, which is thresholded by σ^2_{th} to filter out predictions with high variance as "unknown."

We opt to use the sparse kernel presented in [2] to enable exact updates in equations 3 and 4.

$$k(x, x') = \begin{cases} \sigma_0\left[\frac{2+\cos(2\pi\frac{d}{l})}{3}(1-\frac{d}{l}) + \frac{1}{2\pi}\sin(2\pi\frac{d}{l})\right] & \text{if } d < l \\ 0 & \text{if } d \geq l \end{cases} \tag{8}$$

where $\sigma_0 > 0$ is a constant parameter of the kernel, $l > 0$ is the scale, and d is the distance between x and x'.

References

[1] W.R. Vega-Brown, M. Doniec, and N.G. Roy, "Nonparametric Bayesian inference on multivariate exponential families," *Advances in Neural Information Processing Systems*, pp. 2546-2554, 2014.

[2] A. Melkumyan and F. Ramos, "A Sparse Covariance Function for Exact Gaussian Process Inference in Large Datasets," *Proceedings of the International Joint Conferences on Artificial Intelligence Organization*, vol. 9, pp. 1936-1942, 2009.

[3] J. Wang and B. Englot, "Fast, Accurate Gaussian Process Occupancy Maps via Test-Data Octrees and Nested Bayesian Fusion," *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 1003-1010, May 2016.

Test Data Octrees

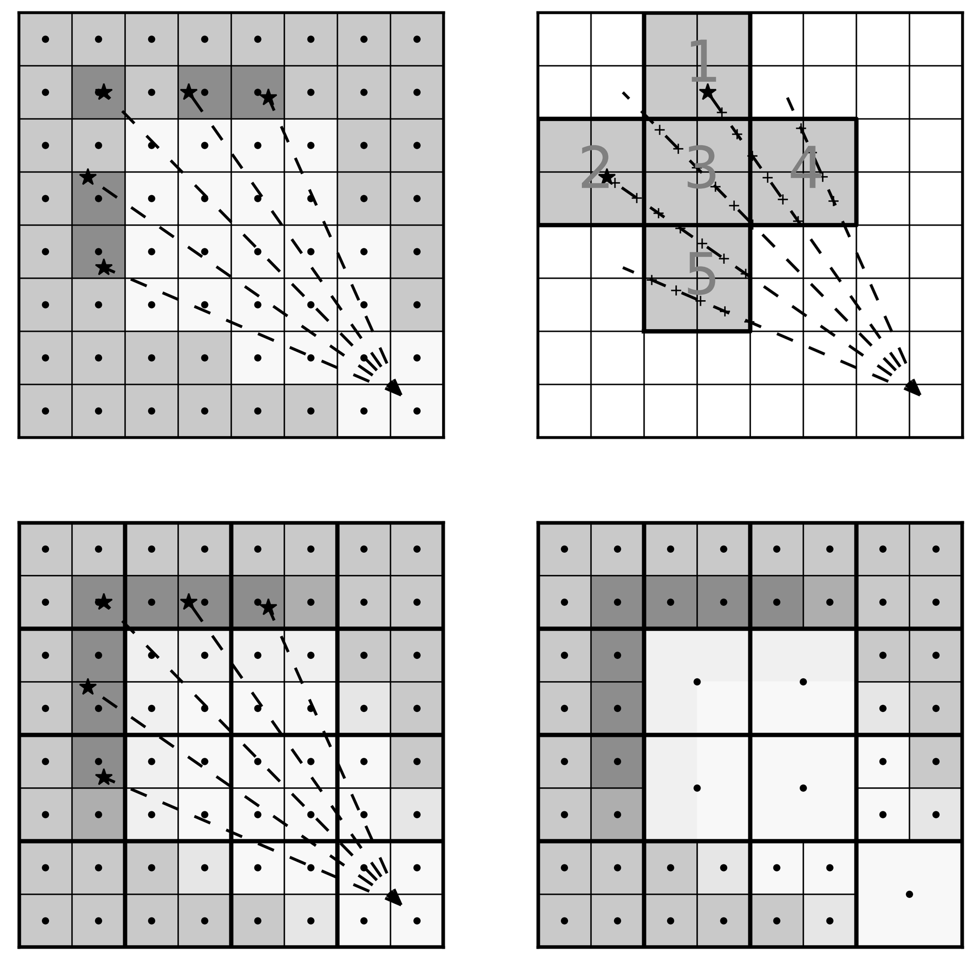
To the right we show the core data structure involved in our approach to occupancy maps: the "test data octree" from [3], which allows us to prune our environment representation to a lower resolution.

Top-left: Standard occupancy grid mapping.

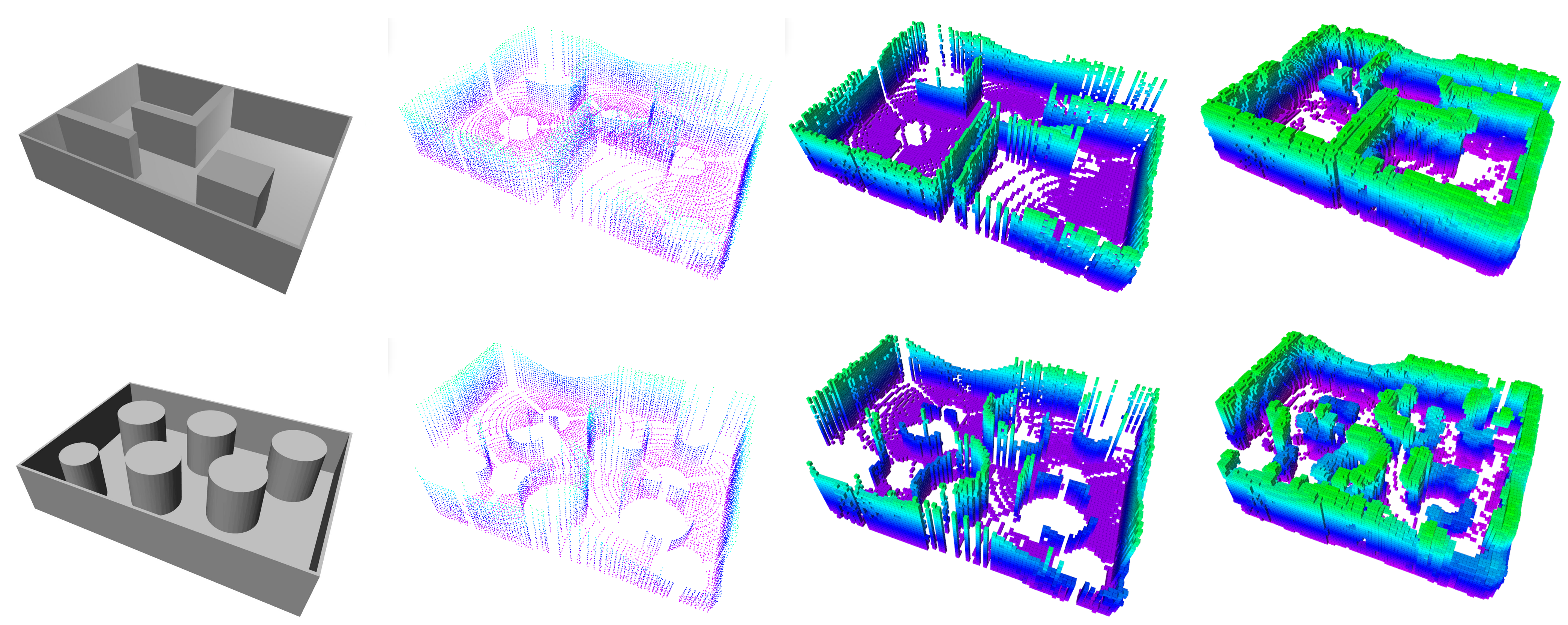
Top-right: Setup for prediction of the occupancy probability of all cells in block 3. The extended block consists of all blocks within distance l of block 3 that contain sensor data or sampled free-space points.

Bottom-left: For each block, the data from the corresponding extended block is aggregated and inference is performed.

Bottom-right: Neighboring cells within a block with the same occupancy state are pruned.

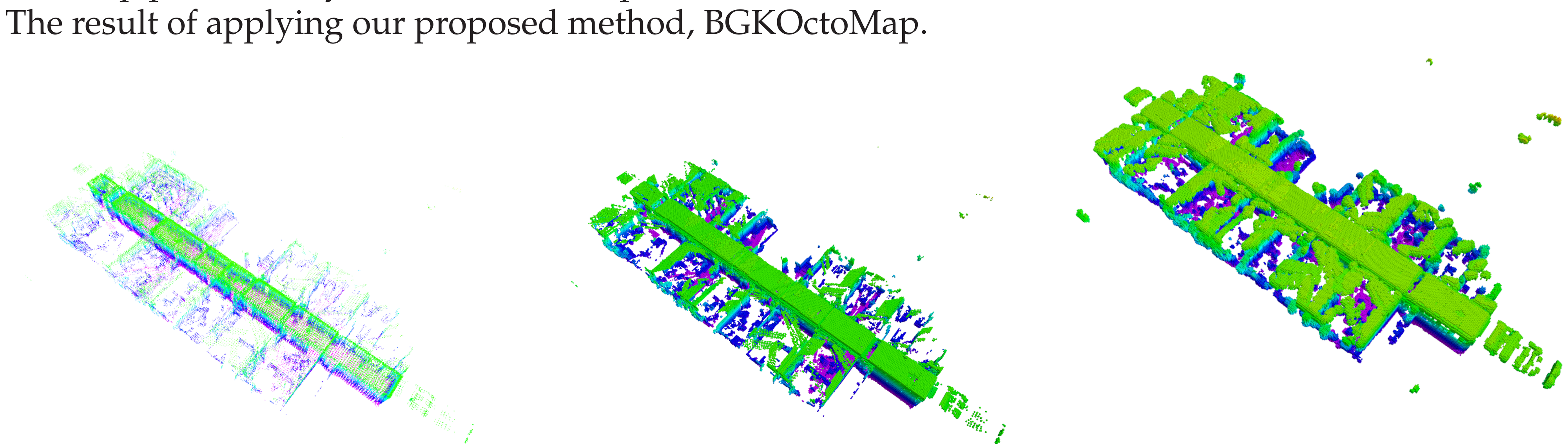


Experimental Results



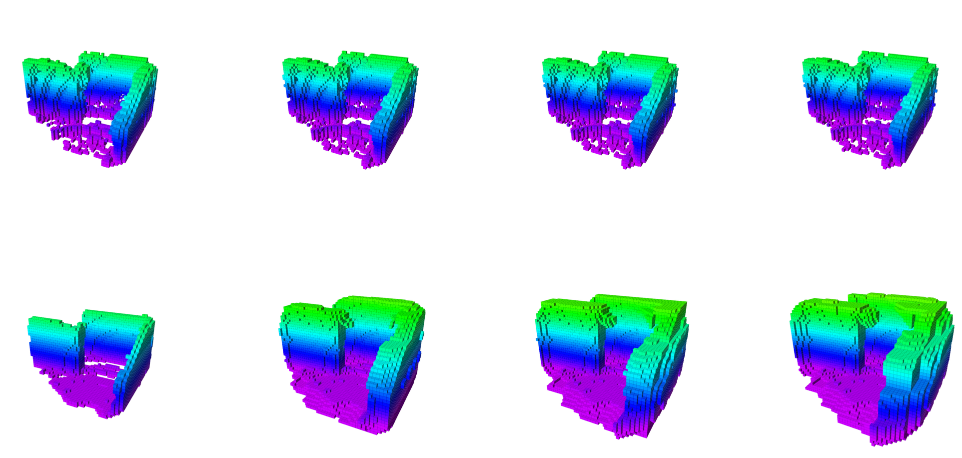
We tested our method "Bayesian Generalized Kernel OctoMap" (BGKOctoMap) in simulated *structured* (**top**) and *unstructured* (**bottom**) environments. We show (colored by height), from **left** to **right**:

1) The Gazebo simulation model for the environment
2) The simulated raw sensor data
3) The map produced by standard OctoMap
4) The result of applying our proposed method, BGKOctoMap.



We also evaluated our method qualitatively on real data from the University of Freiburg, shown above. From **left** to **right** we have:

1) The raw range-sensor data from University of Freiburg Corridor FR-079
2) The map produced by standard OctoMap
3) The map produced by BGKOctoMap



In this experiment, we simulate a robot keeping station, repeatedly scanning the same area.

Top: Our method, BGKOctoMap updated with information from 1, 15, 30, and 60 scans (**left** to **right**) containing the same data.

Bottom: The result after applying the online Gaussian process occupancy mapping method in [3] to the same data.

Qualitatively, our method (**top**) is more stable for long-term mapping scenarios with many repeat observations than the previous method (**bottom**) in which the occupied voxels tend to grow continuously with repeat observations.

We quantitatively evaluated the inference model in both simulated environments. On the **left** we show the receiver operating characteristic (ROC) curve of the classifier for the *structured* map, and for the *unstructured* map on the **right**. Each point on the curve shows the false-positive rate and true-positive rate at a particular threshold separating the positive class "occupied" and the negative class "unoccupied" (up and to the left is better). The area under the curve measures the overall accuracy of the classifier. We compare our method to GPOctomap-NBCM-P presented in [3] and standard OctoMap.