A Additional Theoretical Results and Proofs

In this section, we present all the omitted proofs as well as some additional theoretical results.

A.1 Proof of Theorem 3.5

Proof. Without loss of generality, we only consider the case where $n = 3k + 2$ for some odd number $k > 10^4$.

We define G as the following tree graph:

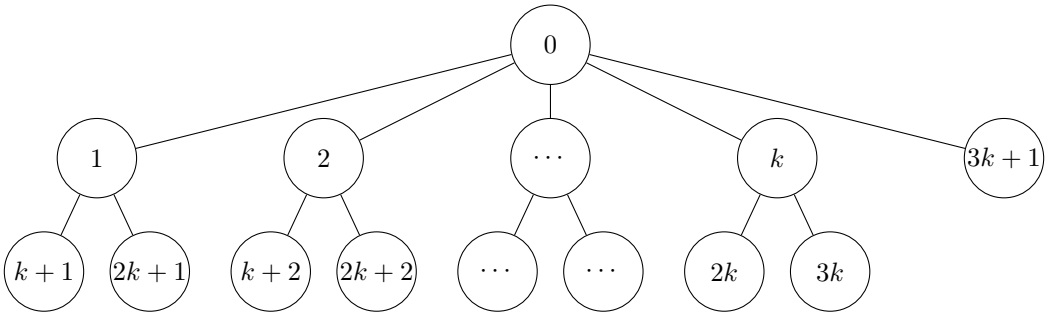

Figure 2: **The tree construction** in the proof of Theorem 3.5

In the graph, 0 is the root node with $k + 1$ children. One of the $k + 1$ children is a leaf node. Each of the other k children has two children.

Now it suffices to show that there exists $\boldsymbol{y} \in \mathcal{C}^n$ satisfying the strong majority voting pattern on G, such that unless $Pb = \Omega(n)$,

$$\sup_{f_G \in \mathcal{F}_G} \frac{1}{n} \sum_{i \in V} \mathbb{I}[f_G(\boldsymbol{X})_i = \boldsymbol{y}_i] \leq \frac{5}{6} + 0.001 \tag{5}$$

Let

$$\mathcal{Y} = \left\{ \boldsymbol{y} \in \{0,1\}^n \middle| \boldsymbol{y}_i = \boldsymbol{y}_{k+i} = \boldsymbol{y}_{2k+i} (\forall i \in [k]);\ \boldsymbol{y}_0 = \boldsymbol{y}_{3k+1} = \left[2 \sum_{j=1}^{k} \boldsymbol{y}_j - 1 \right] \right\}. \tag{6}$$

Then one can check that any $\boldsymbol{y} \in \mathcal{Y}$ has the strong majority voting pattern on G. Based on this observation, we can arbitrarily choose $\boldsymbol{y}_1, \cdots, \boldsymbol{y}_k$ from $\{0,1\}$, and then $\boldsymbol{y}_0, \boldsymbol{y}_{k+1}, \cdots, \boldsymbol{y}_{3k+1}$ can be uniquely determined to obtain some $\boldsymbol{y} \in \mathcal{Y}$.

We apply the *probabilistic method* to prove the existence of the desired $\boldsymbol{y} \in \mathcal{Y}$. Assume that $\boldsymbol{y}$ is sampled from $\mathrm{Unif}(\mathcal{Y})$. Then it suffice to show that the event (5) happens with positive probability unless $Pb = \Omega(n)$.

Note that $\boldsymbol{y}_1, \cdots, \boldsymbol{y}_k \sim$ i.i.d. $\mathrm{Unif}(\{0,1\})$ under our assumption. For a given $f_G \in \mathcal{F}_G$, the predicted label $f_G(\boldsymbol{X}) \in \mathcal{C}^n$ is a deterministic vector. Therefore, $\mathbb{I}[f_G(\boldsymbol{X})_1 = \boldsymbol{y}_1], \cdots, \mathbb{I}[f_G(\boldsymbol{X})_k = \boldsymbol{y}_k] \sim$ i.i.d. $\mathrm{Unif}(\{0,1\})$.

By Hoeffding's inequality, for $\varepsilon = 0.0001$, we have

$$\Pr\left[\frac{1}{k} \sum_{i=1}^{k} \mathbb{I}[f_G(\boldsymbol{X})_i = \boldsymbol{y}_i] > \frac{1}{2} + \varepsilon \right] \leq \exp\left(-2k\varepsilon^2\right).$$

Recall that $k > 10^4$ and $\varepsilon = 0.0001$, we have

$$\begin{aligned} & \frac{1}{k} \sum_{i=1}^{k} \mathbb{I}[f_G(\boldsymbol{X})_i = \boldsymbol{y}_i] \leq \frac{1}{2} + \varepsilon \\ \Rightarrow \quad & \frac{1}{n} \sum_{i \in V} \mathbb{I}[f_G(\boldsymbol{X})_i = \boldsymbol{y}_i] \leq \frac{\left(\frac{1}{2} + \varepsilon\right) k + (2k+1)}{n} \leq \frac{5}{6} + 0.001. \end{aligned}$$

Equivalently,

$$\frac{1}{n}\sum_{i\in V}\mathbb{I}[f_G(\boldsymbol{X})_i=\boldsymbol{y}_i]>\frac{5}{6}+0.001 \quad \Rightarrow \quad \frac{1}{k}\sum_{i=1}^{k}\mathbb{I}[f_G(\boldsymbol{X})_i=\boldsymbol{y}_i]>\frac{1}{2}+\varepsilon.$$

Therefore,

$$\Pr\left[\frac{1}{n}\sum_{i\in V}\mathbb{I}[f_G(\boldsymbol{X})_i=\boldsymbol{y}_i]>\frac{5}{6}+\varepsilon\right]\leq \Pr\left[\frac{1}{k}\sum_{i=1}^{k}\mathbb{I}[f_G(\boldsymbol{X})_i=\boldsymbol{y}_i]\right]\leq \exp\left(-2k\varepsilon^2\right).$$

Applying the union bound over $\mathcal{F}_G$, we have

$$\Pr\left[\exists\ f_G\in\mathcal{F}_G,\ \frac{1}{n}\sum_{i\in V}\mathbb{I}[f_G(\boldsymbol{X})_i=\boldsymbol{y}_i]>\frac{5}{6}+\varepsilon\right]\leq |\mathcal{F}_G|\exp\left(-2k\varepsilon^2\right).$$

Recall that $\mathcal{F}_G$ is a GNN class with P parameters and b-bit precision. Thus, $|\mathcal{F}_G|\leq 2^{Pb}$. If $Pb<n\varepsilon^2$, then

$$\begin{aligned}&\Pr\left[\exists\ f_G\in\mathcal{F}_G,\ \frac{1}{k}\sum_{i\in k}\mathbb{I}[f_G(\boldsymbol{X})_i=\boldsymbol{y}_i]>\frac{5}{6}+\varepsilon\right]<1\\ \Rightarrow\quad &\Pr\left[\sup_{f_G\in\mathcal{F}_G}\frac{1}{n}\sum_{i\in V}\mathbb{I}[f_G(\boldsymbol{X})_i=\boldsymbol{y}_i]\leq\frac{5}{6}+0.001\right]>0.\end{aligned}$$

Therefore, unless $Pb=\Omega(n)$, the desired $\boldsymbol{y}\in\mathcal{Y}$ exists, which concludes the proof. □

A.2 Extending Theorem 3.5 to anti-majority voting label patterns

In this subsection, we presents an additional theoretical result to show that LLIBEP can also perform well on label patterns other than the strong majority voting pattern, while the label-oblivious GNNs also fail. The *anti-majority voting pattern* has been defined in Definition 3.3. In the anti-majority voting label pattern, the neighbors' labels still contains critical information for predicting the label of a node, which is similar to the strong majority voting pattern.

Parallel to Theorem 3.5, we show that label patterns is hard to represent for any label-oblivious GNN class:

Theorem A.1. *Suppose $n,d\in\mathbb{N}^*$, and $\mathcal{F}_G$ is any given label-oblivious GNN class with P parameters and b-bit precision. There exists a rooted tree G with n nodes, such that for any node feature $\boldsymbol{X}\in\mathbb{R}^{n\times d}$, there exist labels $\boldsymbol{y}\in\mathcal{C}^n$ that satisfy the followings unless $Pb=\Omega(n)$:*

- *The label $\boldsymbol{y}$ has the anti-majority voting pattern on G;*
- *Any model in $\mathcal{F}_G$ makes non-trivial prediction error. Specifically,*

$$\sup_{f_G\in\mathcal{F}_G}\frac{1}{n}\sum_{i\in V}\mathbb{I}[f_G(\boldsymbol{X})_i=\boldsymbol{y}_i]\leq\frac{5}{6}+0.001.$$

Proof. The proof idea is the same as that of Theorem 3.5.

We use the same tree construction as presented in App. A.1. For the labels, instead of considering samples from $\mathcal{Y}$ defined in Eq. (6) (which is a set of strong majority voting labels), we consider the following set:

$$\mathcal{Y}=\left\{\boldsymbol{y}\in\{0,1\}^n\middle|\boldsymbol{y}_{k+i}=\boldsymbol{y}_{2k+i}=1-\boldsymbol{y}_i(\forall i\in[k]);\ \boldsymbol{y}_0=\boldsymbol{y}_{3k+1}=1-\left[2\sum_{j=1}^{k}\boldsymbol{y}_j-1\right]\right\}. \tag{7}$$

It's easy to check that any $\boldsymbol{y}\in\mathcal{Y}$ has the anti-majority voting label pattern. The same probabilistic-method-based argument proves the existence of $\boldsymbol{y}\in\mathcal{Y}$ that has the desired property. □

A.3 Proof of Theorem 4.1

Proof. We first prove the case when $\boldsymbol{y}$ has the strong majority voting label pattern.

Recall that we assume binary classification ($c = 2$) and c hidden dimensions in the model. The weight matrices $\boldsymbol{W}_{\text{in}}, \boldsymbol{W}_{\text{out}}$, linear mappings $\{f_{\boldsymbol{\theta}_\ell}\}_{\ell=0}^{L_0}$, and layer normalization operations $\{\text{LayerNorm}_{\boldsymbol{\psi}_\ell}\}_{\ell=1}^{L_0}$ are constructed as follows:

- **Weight matrices.** $\boldsymbol{W}_{\text{in}} = \boldsymbol{W}_{\text{out}} = \boldsymbol{I}_2$ where $\boldsymbol{I}_k$ denotes the identidy matrix in $\mathbb{R}^{k\times k}$.
- **Linear mappings.** $f_{\boldsymbol{\theta}_0} : \boldsymbol{x} \mapsto \boldsymbol{0}$ and $f_{\boldsymbol{\theta}_\ell} : \boldsymbol{x} \mapsto \boldsymbol{x}/3$ for $\ell \in \{1, \cdots, L_0\}$.
- **Layer normalization operations.** For all the layer normalization, the bias parameters are set to $\boldsymbol{0}$ and the gain parameters are set to $\sqrt{2}\boldsymbol{1}$, where $\boldsymbol{0} \in \mathbb{R}^2$ and $\boldsymbol{1} \in \mathbb{R}^2$ are the all-zero and all-one vectors. Then we have $\text{LayerNorm}_{\boldsymbol{\psi}_\ell} : (a, b) \mapsto (\text{sign}(a - b), \text{sign}(b - a))$ for $\ell \in \{1, \cdots, L_0\}$.

Now we show that the above construction makes LLIBEP outputs the true label $\boldsymbol{y}$ given the node features $\boldsymbol{X}$ and known labels $\boldsymbol{y}_{\text{train}}$. Specifically, we prove the following claim:

Claim. Denote by $\text{depth}(i)$ the depth of node $i \in V$. Based on the above construction, for any $\ell \in \{0, 1, \cdots, L_0\}$ and any $i \in V$ satisfying $\text{depth}(i) \geq L_0 - \ell$, $\boldsymbol{B}^{(\ell)}$ satisfies that

- $\boldsymbol{B}_{i0}^{(\ell)} > \boldsymbol{B}_{i1}^{(\ell)}$ if $\boldsymbol{y}_i = 0$;
- $\boldsymbol{B}_{i0}^{(\ell)} < \boldsymbol{B}_{i1}^{(\ell)}$ if $\boldsymbol{y}_i = 1$.

The claim can be proved by induction.

For $\ell = 0$, note that any node of depth L_0 is in the training set V_{train}, since it is a leaf node and $V_{\text{leaf}} \subset V_{\text{train}}$. Besides, $\boldsymbol{W}_{\text{in}} = \boldsymbol{I}_2$ and $f_{\boldsymbol{\theta}_0}$ is a zero mapping. Thus, according to Eq. (2), $\boldsymbol{B}_i^{(0)}$ is the one-hot encoding of $\boldsymbol{y}_i$, which satisfies the condition in the claim.

Assume the claim holds for $\ell \in \{0, 1, \cdots, L_1 - 1\}$ ($1 \leq L_1 \leq L_0 - 1$) and consider the case where $\ell = L_1$. For any node i satisfying $\text{depth}(i) \geq L_0 - L_1$, assume that the label of i is 0 without the loss of generality. Let

$$\boldsymbol{p} = \boldsymbol{A}_i f_{\boldsymbol{\theta}_{L_1}}\left(\text{LayerNorm}_{\boldsymbol{\psi}_{L_1}}(\boldsymbol{B}^{(L_1-1)})\right) = \frac{1}{3}\boldsymbol{A}_i \text{LayerNorm}_{\boldsymbol{\psi}_{L_1}}(\boldsymbol{B}^{(L_1-1)})$$

Then $\boldsymbol{B}_i^{(L_1)} = \sigma\left(\boldsymbol{E}_i + \boldsymbol{p}\right)$

We consider the following two cases:

- **Case 1: i is a leaf node.** In this case, we have

  $$\boldsymbol{p} = \frac{1}{3}\text{LayerNorm}_{\boldsymbol{\psi}_{L_1}}(\boldsymbol{B}_{\text{pa}(i)}^{(L_1-1)}) \qquad \text{where } \text{pa}(i) \text{ denotes the parent of } i.$$

  Note that each entry of $\text{LayerNorm}_{\boldsymbol{\psi}_{L_1}}(\boldsymbol{B}_{\text{pa}(i)}^{(L_1-1)})$ takes value in $\{-1, 1, 0\}$. Besides, we have $\boldsymbol{E}_{i0} = 1$ and $\boldsymbol{E}_{i1} = 0$ since $i \in V_{\text{leaf}} \subset V_{\text{train}}$. Thus,

  $$\boldsymbol{E}_{i0} + \boldsymbol{p}_0 \geq \frac{2}{3};\ \boldsymbol{E}_{i1} + \boldsymbol{p}_1 \leq \frac{1}{3} \quad \Rightarrow \quad \sigma(\boldsymbol{E}_{i0} + \boldsymbol{p}_0) > \sigma(\boldsymbol{E}_{i1} + \boldsymbol{p}_1),$$

  indicating the condition in the claim holds.

- **Case 2: i is a non-leaf node.** In this case, we have

  $$\boldsymbol{p} = \frac{1}{3}\text{LayerNorm}_{\boldsymbol{\psi}_{L_1}}(\boldsymbol{B}_{\text{pa}(i)}^{(L_1-1)}) + \frac{1}{3}\sum_{j\in\text{Child}(i)} \text{LayerNorm}_{\boldsymbol{\psi}_{L_1}}(\boldsymbol{B}_j^{(L_1-1)}).$$

Note that $\text{depth}(j) \geq L_0 - L_1 + 1$ for any $j \in \text{Child}(i)$. Thus, by the induction assumption, the definition of strong majority voting label pattern, and the design of $\text{LayerNorm}_{\boldsymbol{\psi}_{L_1}}$, we have

$$\boldsymbol{p}_c = \frac{1}{3}\boldsymbol{q}_c + \frac{1}{3}\sum_{j\in\text{Child}(i)} \mathbb{I}[\boldsymbol{y}_j = c] \quad (c \in \{0,1\});$$
$$\sum_{j\in\text{Child}(i)} \mathbb{I}[\boldsymbol{y}_j = 0] - \sum_{j\in\text{Child}(i)} \mathbb{I}[\boldsymbol{y}_j = 1] \geq 2,$$

where $\boldsymbol{q} = \text{LayerNorm}_{\boldsymbol{\psi}_{L_1}}(\boldsymbol{B}^{(L_1-1)}_{\text{pa}(i)})$.

Note that $\boldsymbol{q} \in \{(0,0),(0,1),(1,0)\}$. Therefore, $|\boldsymbol{q}_0 - \boldsymbol{q}_1| \leq 1$, and

$$\boldsymbol{p}_0 - \boldsymbol{p}_1 = \frac{1}{3}\left(\sum_{j\in\text{Child}(i)} \mathbb{I}[\boldsymbol{y}_j = 0] - \sum_{j\in\text{Child}(i)} \mathbb{I}[\boldsymbol{y}_j = 1]\right) + \frac{1}{3}(\boldsymbol{q}_0 - \boldsymbol{q}_1) \geq \frac{1}{3}.$$

Besides, we also have $\boldsymbol{E}_{i0} \geq \boldsymbol{E}_{i1}$, no matter whether or not $i \in V_{\text{train}}$. Thus,

$$(\boldsymbol{E}_{i0} + \boldsymbol{p}_0) - (\boldsymbol{E}_{i1} + \boldsymbol{p}_1) \geq \frac{1}{3} \quad \Rightarrow \quad \sigma(\boldsymbol{E}_{i0} + \boldsymbol{p}_0) > \sigma(\boldsymbol{E}_{i1} + \boldsymbol{p}_1),$$

indicating the condition in the claim holds.

In either case, we show that $\boldsymbol{B}_i^{(L_1)}$ satisfies the condition in the claim. So we conclude that the claim is true by induction.

Set ℓ to L_0 in the claim and note that $\boldsymbol{W}_{\text{out}} = \boldsymbol{I}_2$, we obtain that for any $i \in V$

- $\boldsymbol{Y}_{i0} > \boldsymbol{Y}_{i1}$ if $\boldsymbol{y}_i = 0$;
- $\boldsymbol{Y}_{i0} < \boldsymbol{Y}_{i1}$ if $\boldsymbol{y}_i = 1$,

which implies that the model outputs the correct prediction and concludes the proof for the setting where $\boldsymbol{y}$ has the strong majority voting label pattern.

Second, we consider the case where $\boldsymbol{y}$ has the anti-majority voting label pattern. The construction and proof are almost the same as the previous case, except that we construct different mappings $\{f_{\boldsymbol{\theta}_\ell}\}_{\ell=1}^{L_0}$ as follows:

$$f_{\boldsymbol{\theta}_\ell} : (a,b) \mapsto \left(\frac{b}{3}, \frac{a}{3}\right) \quad (\ell \in \{1,\cdots,L_0\}).$$

Based on this construction, one can similarly prove the claim in App. A.3, which directly leads to a proof of this theorem. □

Discussions. The additional theoretical result shows that LLIBEP can perform well on graphs with anti-majority voting label patterns. Although the theorem does not present new proof techniques, it studies a very interesting setting, i.e., the heterophily setting. Under the assumption of anti-majority voting label patterns, a node tends to have different labels from its neighbors, exhibiting the heterophily property. The theoretical result shows that LLIBEP can also work well even under certain heterophily settings. This is in contrast to some of the existing methods (Huang et al., 2020; Wang and Leskovec, 2021) that achieves label-awareness by incorporating the label propagation algorithm (LPA) (Zhou et al., 2003), which implicitly assumes the homophily property.

B Detailed Experiment Settings

B.1 Experiments on synthetic datasets

Data generation. We generate trees as illustrated in Fig. 2. The feature of each node is sampled independently from the standard Gaussian distribution, and the feature dimensionality d is set to 16. We sample the labels of nodes $1, \cdots, k$ independently from $\mathrm{Unif}(\{0, 1\})$, and then decides the label of the other nodes based on Eq. (6) (strong majority voting pattern setting) or Eq. (7) (anti-majority voting pattern setting). According to Theorems 4.1, all the leaf nodes are used as training nodes. For the non-leaf nodes, we randomly split 50%/25%/25% them into the training/validation/test set.

Implementation details. In LLIBEP, we use layer normalization in the intermediate layers. For GCN, we test variants both with and without layer normalization, and select the model with better validation accuracy. In the message passing layer of both GCN and LLIBEP, we use the symmetrically-normalized adjacency matrix following (Kipf and Welling, 2017). Specifically, denote by $\boldsymbol{A}$ the original adjacency matrix. Then we use the following matrix in message passing:

$$\widetilde{\boldsymbol{A}} = \boldsymbol{D}^{-\frac{1}{2}}(\boldsymbol{A} + \boldsymbol{I}_n)\boldsymbol{D}^{-\frac{1}{2}} \quad \text{where } \boldsymbol{D} = \mathrm{diag}((\boldsymbol{A} + \boldsymbol{I}_n)\boldsymbol{1}).$$

This design choice is consistent in all our experiments (including those in Sec. 5.2 and App. C)

Training and evaluation recipes. All the models are trained with Adam as the optimizer (Kingma and Ba, 2015) for 1000 iterations. The peak learning rate is swept across $\{0.003, 0.001, 0.0003, 0.0001\}$. We also experiment with two learning rate schedulers: (1) constant and (2) linearly decaying to 0 during training. We set the dropout probability to 0.2. We do not apply weight decay in training. After training, we evaluate the models trained with different hyper-parameters on the validation set, select the model with the highest validation accuracy, and evaluate the selected model on the test set to obtain the accuracy.

B.2 Experiments on real-world datasets

Training and evaluation recipes. Following (Platonov et al., 2023), we train our models with Adam as the optimizer (Kingma and Ba, 2015) for 1000 iterations on all the datasets. We sweep the peak learning rate is swept across $\{0.003, 0.001, 0.0003, 0.0001\}$. We experiment with two learning rate schedulers: (1) constant and (2) linearly decaying to 0 during training. The dropout probability and the weight decay parameter are swept across $\{0, 0.1, 0.2, 0.3, 0.5, 0.7\}$ and $\{0, 0.001, 0.00001\}$. The model depth is swept across $\{2, 3, 4, 5, 7\}$, and the model width is set to 512. The model uses GeLU as the activation function (Hendrycks and Gimpel, 2016).

For the label-aware GNN baselines, we also tune the method-specific hyper-parameters except GAMLP which contains more than 10 hyper-parameters. For label reuse, we sweep the number of reuse iterations R across $\{1, 2, 3\}$ (Wang et al., 2021). For GCN-LPA, we sweep the number of LPA iterations across $\{1, 2, 3, 5\}$ and the weight of the regularization loss term in $\{0.1, 0.01, 1, 3\}$. For C&S, we sweep the number of correction layers and the number of smoothing layers across $\{1, 4, 7, \cdots, 49\}$, and $\alpha_{\text{correction}}$ and $\alpha_{\text{smoothing}}$ across $\{0.2, 0.4, 0.6, 0.8\}$.

In out experiments, we first train the models with different hyper-parameters and then select the training recipe which maximizes the performance on the validation set. Then we train the model using the selected training recipe using 3 different random seeds, and report the mean and standard deviation of the test results. More detailed descriptions of the settings can be found in App. B.2.

Table 2: **Results of ablation experiments**. The left panel presents ablation experiments on the apriori node belief computation, comparing joint use of features and labels with computation based on only one term. The right panel presents ablation experiments on the learnable belief propagation layer, comparing propagation with and without apriori belief injection (i.e., $\boldsymbol{E}$ in Eq. (3)). The reported results are test accuracy on the two datasets.

	Apriori belief computation			Propagation layer	
	Feature + label	Feature only	Label only	With $\boldsymbol{E}$	Without $\boldsymbol{E}$
Roman-empire	87.91	85.83	28.89	87.91	54.25
Amazon-ratings	53.63	51.98	46.19	53.63	47.51

C Ablation study

We conduct ablation experiments on the roman-empire and amazon-ratings datasets to gain a deeper understanding of the contributions of the design choices in LLIBEP.

Regarding the apriori node belief computation. In Eq. (2), the node features and labels are jointly used to compute the apriori node belief. To understand the information of these two terms, we experiment with LLIBEP variants that only use one of these terms in the apriori node belief computation. The model configuration, training hyper-parameter search space, and the evaluation methods are the same as described in Sec. 5.2.

The experiment results are presented in the left panel of Table 2. We can see that jointly using both node features and known labels leads to the best performances. In particular, if only node features are used in apriori node belief computation, then the model becomes label-oblivious, which leads to significant accuracy drops especially on amazon-ratings. We also note that the accuracy drop varies across datasets, since it depends on how much the information in node feature and the information in label correlations complement each other.

Regarding the learnable linearized belief propagation layers. The design of our belief propagation layer is motivated from the LinBP update defined in Eq. (1). Interestingly, the update requires us to inject the apriori node belief $\boldsymbol{E}$ into all the intermediate layers. We conduct ablation experiments to test the necessity of this apriori node belief injection operation. The experiment results are presented in the right panel of Table 2. We can see that without this operation, the model performances on both dataset drop by more than 10 points. Note that the apriori node belief $\boldsymbol{E}$ contains information of the known labels. Thus, the results indicate that it can be critical for label-aware GNNs to inject the label information to the intermediate layers.