APPENDIX

A LEARNING AND OBTAINING GRAPH EMBEDDINGS WITH CLEP

We summarize the training algorithm of CLEP in Algorithm 1. After training CLEP, the embeddings of each graph can be obtained as in Algorithm 2.

Algorithm 1 The training algorithm of CLEP, for one batch.

Data: Graph batch $\mathcal{B} = \{\mathcal{G}_i \mid \mathcal{G}_i := (\mathbf{X}_i, \mathbf{A}_i), i \in [1, |\mathcal{B}|]\}$.
Modules: Variational encoder $h_{\mathcal{V}}(\cdot; \boldsymbol{\phi})$, soft edge assignment module $\mathrm{SEA}(\cdot)$, community-specific graph encoders $\{h_{\mathcal{G}}^{(k)}(\cdot; \boldsymbol{\theta}) \mid k \in [1, K]\}$, global graph encoder $h_{\mathcal{G}}(\cdot; \boldsymbol{\theta})$, projectors $\{m^{(k)}(\cdot; \boldsymbol{\theta}) \mid k \in [1, K]\}$.

$\mathcal{B}', \mathcal{B}'' \leftarrow \mathrm{perturb}(\mathcal{B})$;
for $i \leftarrow 1$ **to** $|\mathcal{B}|$ **do**
  $\mathbf{K}_i, \boldsymbol{\Lambda}_i \leftarrow h_{\mathcal{V}}(\mathbf{X}_i, \mathbf{A}_i; \boldsymbol{\phi})$;
  sample $\mathbf{Z}_i$ with $\mathbf{K}_i, \boldsymbol{\Lambda}_i$, as in Equation (11);
  $\mathbf{h}_i \leftarrow h_{\mathcal{G}}(\mathbf{X}_i, \mathbf{A}_i)$;
  for $k \leftarrow 1$ **to** K **do**
    $\mathbf{A}_i'^{(k)} \leftarrow \mathrm{SEA}(\mathbf{A}_i', k, \{\mathbf{Z}_i^{(k)} \cdot \mathbf{Z}_i^{(k)\intercal}\}_{k=1,K})$;
    $\mathbf{A}_i''^{(k)} \leftarrow \mathrm{SEA}(\mathbf{A}_i'', k, \{\mathbf{Z}_i^{(k)} \cdot \mathbf{Z}_i^{(k)\intercal}\}_{k=1,K})$;
    $\mathbf{h}_i'^{(k)} \leftarrow h_{\mathcal{G}}^{(k)}(\mathbf{A}_i'^{(k)}, \mathbf{X}_i'; \boldsymbol{\theta})$;
    $\mathbf{h}_i''^{(k)} \leftarrow h_{\mathcal{G}}^{(k)}(\mathbf{A}_i''^{(k)}, \mathbf{X}_i''; \boldsymbol{\theta})$;
    $\mathbf{f}_i'^{(k)} \leftarrow m^{(k)}(\mathbf{h}_i'^{(k)}; \boldsymbol{\theta})$;
    $\mathbf{f}_i''^{(k)} \leftarrow m^{(k)}(\mathbf{h}_i''^{(k)}; \boldsymbol{\theta})$
  end for
  compute $\mathbb{E}_{\mathbf{Z}_i \sim Q_{\boldsymbol{\phi}}(\mathbf{Z}_i)}\big[\log p_{\boldsymbol{\theta}}(i \mid \mathbf{Z}_i, \mathbf{A}_i, \mathbf{X}_i)\big]$ by Equation (5);
  compute $\mathbb{E}_{\mathbf{Z}_i \sim Q_{\boldsymbol{\phi}}(\mathbf{Z}_i)}\big[\log p(\mathbf{A}_i \mid \mathbf{Z}_i)\big]$ with the generative model defined in Section 3.1;
  compute $D_{\mathrm{KL}}(Q_{\boldsymbol{\phi}}(\mathbf{Z}_i) \,\|\, P(\mathbf{Z}_i))$ as in Equation (12);
  $\ell_i \leftarrow \mathbb{E}_{\mathbf{Z}_i \sim Q_{\boldsymbol{\phi}}(\mathbf{Z}_i)}\big[\log p_{\boldsymbol{\theta}}(i \mid \mathbf{Z}_i, \mathbf{A}_i, \mathbf{X}_i) + \log p(\mathbf{A}_i \mid \mathbf{Z}_i)\big] - D_{\mathrm{KL}}(Q_{\boldsymbol{\phi}}(\mathbf{Z}_i) \,\|\, P(\mathbf{Z}_i))$;
end for
$\mathcal{L}_{\mathrm{CLEP}} \leftarrow \mathbb{E}_{\mathbb{B}} \sum_{i=1}^{|\mathbb{B}|} \ell_i$;
$\boldsymbol{\phi} \leftarrow \boldsymbol{\phi} + \eta_{\boldsymbol{\phi}} \cdot \nabla_{\boldsymbol{\phi}} \mathcal{L}_{\mathrm{CLEP}}$;
$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \eta_{\boldsymbol{\theta}} \cdot \nabla_{\boldsymbol{\theta}} \mathcal{L}_{\mathrm{CLEP}}$;

Algorithm 2 The algorithm to obtain the embeddings of each graph.

Data: Graph $\mathcal{G} := (\mathbf{X}, \mathbf{A})$.
Modules: Variational encoder $h_{\mathcal{V}}(\cdot; \boldsymbol{\phi})$, soft edge assignment module $\mathrm{SEA}(\cdot)$, community-specific graph encoders $\{h_{\mathcal{G}}^{(k)}(\cdot; \boldsymbol{\theta}) \mid k \in [1, K]\}$.
Output: Graph embedding $\mathbf{h}$.

$\mathbf{K}, \boldsymbol{\Lambda} \leftarrow h_{\mathcal{V}}(\mathbf{X}, \mathbf{A}; \boldsymbol{\phi})$;
sample $\mathbf{Z}$ with $\mathbf{K}, \boldsymbol{\Lambda}$, as in Equation (11);
for $k \leftarrow 1$ **to** K **do**
  $\mathbf{A}^{(k)} \leftarrow \mathrm{SEA}(\mathbf{A}, k, \{\mathbf{Z}^{(k)} \cdot \mathbf{Z}^{(k)\intercal}\}_{k=1,K})$;
  $\mathbf{h}^{(k)} \leftarrow h_{\mathcal{G}}^{(k)}(\mathbf{A}^{(k)}, \mathbf{X}; \boldsymbol{\theta})$;
end for
$\mathbf{h} \leftarrow [\mathbf{h}^{(1)}, \mathbf{h}^{(2)}, \cdots, \mathbf{h}^{(K)}]$.

B More Ablation Studies

In Equation (5), we use a set of learned weights to balance the K community-specific contrastive learning tasks. A simplification to this step is to replace these learned weights by directly setting $p_i^{(k)} = \frac{1}{K}$, which is referred to as "CLEP-average". We compare the unsupervised graph classification results obtained by regular CLEP, CLEP-average and their collective base model GraphCL. As shown in Table 3, the improvement from community-wise contrastive learning is consistent across all benchmarks, and the attention-like contrastive task balancing mechanism, as elaborated in Section 3.3, can effectively enhance the advantage of community-wise contrastive learning.

Table 3: Comparison of graph classification performance (average accuracy $\pm$ standard error).

Method	**MUTAG**	**PTC_MR**	**PROTEINS**	**NCI1**	**IMDB-B**	**IMDB-M**	**RDT-B**	**RDT-M5K**
GraphCL	86.8 $\pm$ 1.3	63.6 $\pm$ 1.8	74.4 $\pm$ 0.5	77.9 $\pm$ 0.4	71.1 $\pm$ 0.4	50.7 $\pm$ 0.4	89.5 $\pm$ 0.8	56.0 $\pm$ 0.3
CLEP-average	90.5$\pm$ 0.7	64.5 $\pm$ 1.3	75.8 $\pm$ 0.6	78.4 $\pm$ 0.3	73.4 $\pm$ 0.3	51.7 $\pm$ 0.4	86.5 $\pm$ 0.6	56.1 $\pm$ 0.4
CLEP	91.2 $\pm$ 0.8	65.1 $\pm$ 1.2	76.4 $\pm$ 0.4	78.5 $\pm$ 0.4	75.6 $\pm$ 0.4	52.0 $\pm$ 0.3	87.3 $\pm$ 0.5	56.4 $\pm$ 0.3