A COMPLEXITY ANALYSIS

We represent the overall algorithmic flow of the model as follows. Furthermore, the time complexity of our model is analyzed.

A.1 TIME COMPLEXITY OF LOW-HIGH FREQUENCY SIGNAL DISENTANGLEMENT

For the low- and high-frequency filter module, the number of nodes N, the number of edges $|\mathcal{E}|$, the feature dimension F of each graph, and the operation of each layer are considered when calculating the time complexity. For the operation of each layer, the time complexity is $O(|\mathcal{E}| + N \times F^2))$, and the model has L layers, so the overall time complexity is $O\big(L \times (|\mathcal{E}| + N \times F^2)\big)$. It can be seen that the overall time complexity of the low- and high-frequency filter module is mainly related to the structure of the graph, that is, it is positively correlated with the number of nodes and the feature dimension.

A.2 TIME COMPLEXITY OF LOW-FREQUENCY INTRA-CLASS CONSISTENCY

For the part of low-frequency intra-class consistency, the calculation of time complexity mainly involves the number of samples N_s and N_t of the source domain and the target domain, and the number of classification classes C of the task, and the overall time complexity is $O(N_s \times C + N_t \times C)$.

A.3 TIME COMPLEXITY OF HIGH-FREQUENCY CONTRASTIVE LEARNING

For high-frequency contrastive learning, the computational time complexity mainly involves calculating the similarity matrix and the cyclic traversal to find positive and negative samples. For the number of source domain and target domain graphs are N_s and N_t respectively, the time complexity of computing the similarity matrix is $O(N_s \times N_t \times F)$, and the time complexity of cyclic traversal of positive and negative samples is $O\big((N_s + N_t) \times max(N_s, N_t)\big)$.

Algorithm 1 The training process of SnLH model

Input: The labeled graph in the source domain $\mathcal{D}_s$; Unlabeled graph in the target domain $\mathcal{D}_t$.
Output: All the predicted values of the target domain graph along with the accuracy.

1: Initialize the parameters of the model randomly.
2: **while** the model is not convergence **do**
3: Sample batches of data from $\mathcal{D}_s$ and $\mathcal{D}_t$, respectively;
4: The sampled data is fed into a low- and high-frequency filter and a graph-level representation is obtained by a readout function;
5: Maximizing cross-domain low-frequency mutual information and contrastive learning of cross-domain high-frequency Information;
6: Calculate the overall loss function $\mathcal{L} = \mathcal{L}_{ce} + \mathcal{L}_{high}^{cl} + \mathcal{L}_{low}^{kd}$, and backpropagation, and update the model parameters.
7: **end while**

B BASELINES

The baseline models for all comparisons are introduced as follows:

- **WL subtree:** The method is based on the Weisfeiler-Lehman algorithm, and the main idea is to construct the feature representation of a node by recursively aggregating the information of the node and its neighbors.
- **GCN:** The GCN model continuously updates the node information by aggregating the information of neighbors and uses an iterative way to generate coding vectors to capture cross-domain information.
- **GIN**: GIN is an architecture for graph neural networks that enhances graph representation by designing a specific aggregation mechanism that enables it to capture more complex graph structural information.

- **GMT**: GMT is a deep learning method for graph learning that combines the advantages of graph neural networks and Transformer architectures to enhance graph representation and matching accuracy.
- **CIN:** CIN aims to mitigate cross-domain differences by extending the traditional Weisfeiler-Lehman algorithm to handle fine-grained graph structures.
- **CDAN:** CDAN is a method for cross-domain learning, and its core idea is to reduce the distribution difference between the source domain and the target domain through conditional adversarial training.
- **ToAlign:** ToAlign is a deep learning method for cross-domain alignment, which aims to solve the feature distribution mismatch problem in the domain adaptation task.
- **MetaAlign:** MetaAlign is a meta-learning method for cross-domain adversarial learning, which aims to solve the feature alignment problem in domain adaptation.
- **DUA:** DUA is a cross-domain learning algorithm that improves the generalization ability of the model by considering the information of the source domain and the target domain at the same time, which aims to solve the problem of effective learning in the case of mismatched data distribution of the source domain and the target domain.
- **DEAL:** DEAL is an algorithm suitable for cross-domain learning, which uses adaptive perturbation and performs adversarial training with the domain discriminator to solve the problem of domain difference.
- **CoCo:** The CoCo method uses coupled branches and ensemble contrastive learning techniques to reduce the inter-domain differences and improve the performance of the model on cross-domain problems.
- **To-UGDA:** The TO-UGDA method aims to solve the problem of insufficient labeled data in the target graph domain by combining domain invariant features, adversarial alignment, and meta-pseudo-label techniques.
- **A2GNN:** The A2GNN model derives the generalization bound of multi-layer GNN and combines the constraint of maximizing the Mean difference (MMD) to reduce the difference between domains.

C Experiment Details

In this part, we will further describe some experiment-related details as follows.

C.1 Main Result Details

In the main experiment, our hyperparameter settings are as follows: the ratio of low- and high-frequency information λ is 0.8, the number of layers is 4, the dimension of the hidden layer is 64, the temperature coefficient of the cross-domain low-frequency mutual information maximization module τ_{kd} is 2.0, the temperature coefficient of the cross-domain high-frequency information contrast learning module τ_{cl} is 0.2, and the learning rate is 2e-3. Furthermore, we conducted several random experiments to obtain the mean and standard deviation of the output results as the final results. In the comparison experiment with the performance of the latest methods, the A2GNN model is mainly applied to the node classification task. To make a fair comparison, we processed the node feature output of A2GNN with the same processing as our model through the readout function, but the result is not ideal and cannot extract good graph representations.

C.2 Additional Experimental Details

For the experimental study and the experiment of low- and high-frequency information influence, we conduct multiple experiments and record the average of the results as the final result. For the sensitivity analysis of the ratio parameter λ of low- and high-frequency information, we make several experiments and record the mean and standard deviation as our final results.