CODE: Towards Partial Label Graph Learning via Coupled Dual Separation

Yiyang Gu*
Peking University
Beijing, China
yiyanggu@pku.edu.cn

Taian Guo*
Peking University
Beijing, China
taianguo@stu.pku.edu.cn

Hang Zhou
University of California, Davis
Davis, CA, USA
hgzhou@ucdavis.edu

Zihao Chen
Peking University
Beijing, China
g.e.challenger@pku.edu.cn

Zhiping Xiao†
University of Washington
Seattle, WA, USA
patxiao@uw.edu

Yifang Qin*
Peking University
Beijing, China
qinyifang@pku.edu.cn

Xiao Luo†
University of California, Los Angeles
Los Angeles, CA, USA
xiaoluo@cs.ucla.edu

Wei Ju*
Peking University
Beijing, China
juwei@pku.edu.cn

Yifan Wang
University of International Business and Economics
Beijing, China
yifanwang@uibe.edu.cn

Ming Zhang*†
Peking University
Beijing, China
mzhang_cs@pku.edu.cn

Abstract

Graph classification is a fundamental machine learning problem with extensive applications in multimedia and biochemical analysis. Contemporary graph classification models usually require precise graph labels for supervision, even after self-supervised pre-training. However, in practical applications, the extensive precise annotation of graphs could be expensive or impractical. To exploit data efficiently, this work studies partial label graph learning, in which each graph is linked to a set of candidate labels but only one of them is accurate. Label ambiguity would bring difficulties in extracting graph semantics and the risk of overfitting noisy partial labels. Here, we present a novel approach called Coupled Dual Separation (CODE). To improve graph semantics mining under label ambiguity, our CODE contains a message passing branch as well as a graph kernel branch, which explore graph semantics implicitly and explicitly, respectively. To facilitate information exchange, we utilize one branch to separate partially labeled graphs into an informative set and an uninformative set, which provides guidance for the optimization of the other branch. Furthermore, to mitigate the risk of overfitting, parameters in coupled branches are partitioned into critical and non-critical ones for separated optimization procedures. Extensive experiments on several benchmark datasets validate the effectiveness of the proposed CODE.

CCS Concepts

• **Computing methodologies** → *Learning latent representations*; • **Mathematics of computing** → *Graph algorithms.*

Keywords

Partial Label Learning, Graph Classification, Graph Neural Networks, Graph Kernels

ACM Reference Format:
Yiyang Gu, Taian Guo, Hang Zhou, Zihao Chen, Zhiping Xiao, Yifang Qin, Xiao Luo, Wei Ju, Yifan Wang, and Ming Zhang. 2025. CODE: Towards Partial Label Graph Learning via Coupled Dual Separation. In *Proceedings of the 33rd ACM International Conference on Multimedia (MM '25), October 27–31, 2025, Dublin, Ireland.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3746027.3755812

1 Introduction

In the field of multimedia analysis, data usually has complicated correlations across domains such as images, videos and texts [35]. Due to the capacity of depicting these correlations, graphs have received increasing attention in different multimedia applications such as multimedia recommendation [58], cross-modal information

*National Key Laboratory for Multimedia Information Processing, School of Computer Science, PKU-Anker LLM Lab, Peking University.
†Corresponding authors.

MM '25, Dublin, Ireland

ACM ISBN 979-8-4007-2035-2/2025/10
https://doi.org/10.1145/3746027.3755812

retrieval [5] and multimedia event extraction [12, 32]. Among various graph-based problems, graph classification aims to assign labels to the whole graphs [47, 73], which is a fundamental graph machine problem with extensive multimedia applications including social network analysis [22, 66], multimodal analysis [36, 48, 73], text mining [47, 73], and molecular property prediction [14, 52, 72]. Graph neural networks (GNNs) have shown impressive performance in learning high-quality node and graph representations [23, 54, 57, 60, 64, 70], which have been applied to solve graph classification. These GNN approaches usually adopt the message passing mechanism to extract node characteristics and topological information into graph representations. In particular, they aggregate neighborhood information to update node representations iteratively and then summarize all node representations into graph representations using different pooling operators for downstream classification.

Despite their great success, most approaches are data-hungry and require abundant labeled graphs for the graph classification tasks. Unsupervised graph representation learning methods [26, 33, 68] also rely on extensive graph labels to train classifiers for the downstream graph classification tasks. However, in real-world applications, it would be extremely expensive to obtain accurate annotation for all training graph samples. For instance, determining the properties of chemical compounds requires costly density functional theory (DFT) calculations [1, 6]. To lower the training cost, an economic approach is to utilize automatic annotation tools instead of human labor, which could inevitably introduce noise and label ambiguity, due to potential inconsistency in their annotations. Another practical example is that proteins have hierarchical label structures, while we could only have access to coarse super-class labels and lack accurate subclass labels in some scenarios, leading to difficulties in fine-grained prediction with multiple ambiguous subclass labels [3, 69]. To tackle this problem, we study partial label graph learning in which every graph example is assigned a candidate set of ambiguous labels while only one of them is correct.

However, developing an effective partial label graph learning framework is a non-trivial problem, which requires us to solve the following two challenges. (1) ***How to extract discriminative topological information from graphs without sufficient supervision information?*** Previous graph neural network approaches [8, 18, 64] usually utilize the message passing mechanism to extract graph structural information implicitly under supervision. However, in our scenarios, label supervision is not adequate, which could generate low-quality graph representations and fail the classification task eventually. (2) ***How to reduce the influence of noisy candidate labels?*** In real-world scenarios, partial label learning may introduce numerous noisy and ambiguous labels to graph data, which could mislead the optimization of GNNs. Therefore, it is highly expected to enhance the generalization capacity of GNNs, which can get rid of the overfitting of these noisy labels for high classification performance.

To tackle the above challenges, in this work, we propose a novel approach named Coupled Dual Separation (CODE). The core of the proposed CODE is to introduce a message passing branch and a graph kernel branch for complementary graph topology mining, which interact with the dual separation of parameters and samples. In particular, different from the message passing branch for implicit graph topology mining, our graph kernel branch introduces learnable hidden graphs and random walk kernels to explore graph topology explicitly. To encourage the interaction of these coupled branches, CODE utilizes one branch to generate an informative set by measuring the confidence and cross-branch discrepancy, which can guide the optimization of the other branch. Compared with traditional pseudo-labeling, our cross-branch supervision can promote the knowledge exchange between two branches while reducing the potential error accumulation. In addition, to reduce the risk of overfitting noisy candidate labels, we separate the whole parameters into crucial ones and noncrucial ones. The crucial ones would be optimized with standard gradient descent while the noncrucial ones are encouraged to shrink. In this way, we can utilize fewer parameters in our model with higher generalization capacity. Extensive experiments on four benchmark datasets validate the superiority of the proposed CODE in comparison to a variety of baselines. We also involve extensive ablation studies and visualization to demonstrate the effectiveness of our proposed CODE. The main contribution of this work are three points as below:

- *Underexplored Problem.* We investigate an underexplored but practical problem of partial label graph learning, which is of great significance for various real-world multimedia and biochemical applications in challenging low-resource scenarios.
- *Novel Methodology.* We propose a novel approach containing a message passing branch and a graph kernel branch to jointly explore complementary graph semantic information, and design a novel coupled dual separation mechanism to facilitate organic information exchange between the two branches and alleviate overfitting to noisy candidate labels.
- *Comprehensive experiments.* Comprehensive experiments on four graph benchmark datasets demonstrate the effectiveness of our CODE when comparing against extensive competing methods.

2 Related Work

2.1 Graph Classification

Graph classification problem aims to assign class labels to an entire graph. It is useful in applications such as social network analysis [28, 67], and chemical molecule property analysis [14, 72]. At an earlier stage, graph kernel methods, which learn the graphs' similarity by analyzing their subgraphs, are frequently used [2, 50, 51]. Recently, it is more popular to use graph neural networks (GNNs) for graph learning [18, 23, 39, 54, 60, 61]. These models typically use pre-defined node features, followed by multiple convolution layers that each aggregates messages from the nodes' neighborhoods. To have graph-level representation, graph-pooling is essential [10, 17, 31, 45]. Graph pooling aims to reduce the number of nodes in a graph while preserving its information. Most of these approaches fall into two categories: flat pooling (i.e. graph readout) that directly concludes a representation in one step, and hierarchical pooling that iteratively coarsens the graph thus preserves graph structure information better [34]. For instance, SAGPool [31] is a typical type of hierarchical pooling approach, which uses a self-attention mechanism to identify crucial nodes from graphs. Some works integrate message passing and graph kernels for graph learning. GNTK [11] models the GNN as a kernel method, focusing on understanding GNNs from a kernel perspective by analyzing their behavior in the infinite-width regime. GCKN [4] uses the

Nyström method for approximating feature mappings at each layer, which can be seen as a novel GNN where filters are unsupervisedly learned through kernel approximation. These methods for solving the graph classification task generally have promising results, but on the other hand, they heavily rely on end-to-end supervised optimization, which requires a large number of reliable graph-level ground-truth labels. Unsupervised graph representation learning methods such as GraphCL [68] and GraphACL [40] also rely on extensive graph labels to train classifiers for the downstream graph classification tasks. In reality, graph-level labeling can be very costly. Therefore, we propose to explore the graph classification problem under the partial label learning setting, which would be more practical since it tolerates more ambiguity and noise in graph data.

2.2 Partial Label Learning

Partial label learning assumes that each training sample is assigned a group of candidate labels, among which only one is correct [20, 25, 65]. Labels are thus assigned ambiguously. Many early works employ average-based algorithms, where all candidate labels are given equal importance when determining the training objective [9, 19, 71]. However, treating all candidate labels equally may lead to a misleading objective, limiting model performance. A natural improvement is to identify the most likely true label among all. We refer to these as identification-based methods [15, 42, 59, 74]. One such example is PLDA [59], which combines similarity-based modeling with prototype-guided learning to enhance feature discrimination. More recently, contrastive learning has been incorporated into partial label models to further improve performance. For example, PiCO [56], which achieves strong results in vision tasks, learns a class prototype embedding to guide label disambiguation through contrastive learning. While these methods are designed for Euclidean data, their extension to non-Euclidean graph structures remains underexplored. A recent attempt, DEER [21], applies distribution divergence-based graph contrast to enhance partial label learning on graphs, but it lacks explicit modeling of high-order structural semantics. To address this, we propose CODE, a dual-branch framework that combines message passing and learnable graph kernels for complementary graph topology mining, enabling more robust graph learning under label ambiguity.

3 Methodology

Problem Definition. *(Partial label graph learning)* An attribute graph can be denoted as $G = (\mathcal{V}, \mathcal{E}, \mathbf{X})$, where $\mathcal{V}$ is the set of nodes, $\mathcal{E}$ is the set of edges, $\mathbf{X} \in \mathbb{R}^{|\mathcal{V}| \times d}$ is the matrix of node attributes with d dimension. Let $\mathcal{D} = \{G_i, Y_i\}_{i=1}^{n}$ denote a training set, where each graph G_i is annotated with a candidate label set $Y_i \subset \mathcal{Y} = \{1, 2, \cdots, C\}$. C is the number of graph categories. The true label $y_i \in Y_i$ is included within the candidate set, yet unknown during the training. The goal of our problem is to learn a reliable model to predict the unique ground-truth category for each graph in the test set accurately.

Framework Overview. This work provides a novel framework CODE for partial label graph learning. Our CODE contains a message passing branch and a graph kernel branch, which are integrated into our dual separation optimization paradigm. The message passing branch utilizes the neighboring aggregation mechanism to explore graph topology implicitly while the graph kernel branch introduces hidden graphs to explore topology explicitly. Then, we identify informative data using confidence and cross-branch discrepancy for information exchange. In addition, the whole network parameters are separated into crucial ones and noncrucial ones with different optimization procedures. The overview of the proposed framework CODE is illustrated in Figure 1, and we will elaborate on the details below.

3.1 Message Passing Branch

Exploiting the impressive success of message passing neural networks (MPNNs) [64] in capturing graph characteristics and semantics, we follow the typical message passing mechanism to encode the graphs in the first branch. By iteratively aggregating information from neighboring nodes to update node representations, this branch mines the graph topology implicitly. To be more specific, for each node $v \in \mathcal{V}$, information is first aggregated from the embedding of neighboring nodes in the last layer. Then, the node v's embedding in the current layer is obtained by merging its embedding in the last layer and the information aggregated from the neighborhood. Formally, the node v's embedding $\mathbf{h}_v^{(l)}$ in the l^{th} layer can be computed as:

$$\mathbf{h}_{N(v)}^{(l)} = \mathrm{AGG}_\theta^{(l)}\left(\left\{\mathbf{h}_u^{(l-1)}, \forall u \in N(v)\right\}\right), \tag{1}$$

$$\mathbf{h}_v^{(l)} = \mathrm{COM}_\theta^{(l)}\left(\mathbf{h}_v^{(l-1)}, \mathbf{h}_{N(v)}^{(l)}\right), \tag{2}$$

where $N(v)$ represents the node v's neighborhood, AGG_θ denotes the aggregation function, and COM_θ denotes the combination function. After L iterative propagations, the graph-level representation $\boldsymbol{z}^{(1)}$ can be obtained by summarizing all the node representations in the L^{th} layer as follows:

$$\boldsymbol{z}^{(1)} = \sum_{v \in \mathcal{V}} \mathbf{h}_v^{(L)}. \tag{3}$$

Finally, the predicted label distribution $\boldsymbol{p}^{(1)} \in [0, 1]^C$ is generated through a multi-layer perceptron (MLP) with a softmax activation function,

$$\boldsymbol{p}^{(1)} = f_{MPB}(G; \Theta) = \mathrm{MLP}_\theta(\boldsymbol{z}^{(1)}), \tag{4}$$

where Θ is the parameter set for the message passing branch.

3.2 Graph Kernel Branch

However, the message passing branch only implicitly captures graph structure through the propagation of information along edges under supervision, which could generate inferior representation with inadequate annotation and fail to explore diverse high-order substructures. Towards this end, inspired by graph kernels that can count various substructures such as paths and subtrees in a graph [30], we introduce a differentiable random walk kernel function to explicitly capture topological information within the graph. First, we adopt M hidden graphs $\{G'_m\}_{m=1}^{M}$ parameterized by trainable adjacency matrices and node attribute matrices. We further assume them to be undirected graphs without self-loops to reduce parameter complexity. It is expected that these hidden graphs can learn substructures helpful in distinguishing graphs from different categories. Subsequently, we utilize a differentiable random walk kernel function to calculate the similarity between the input graph

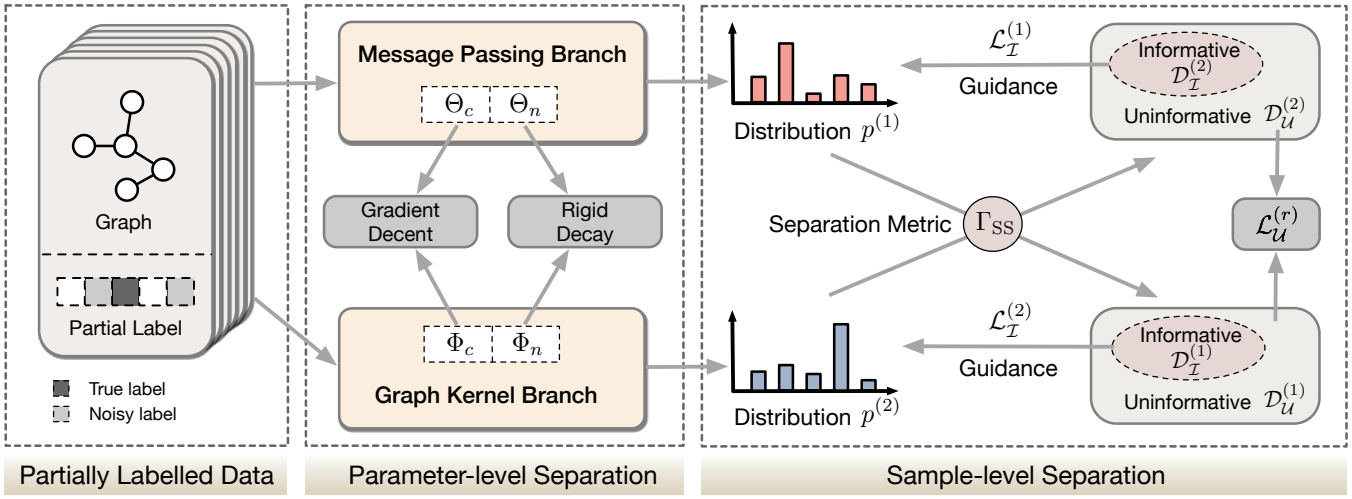


Figure 1: An overview of the proposed framework CODE for partial label graph learning. Our CODE first leverages a message passing branch and a graph kernel branch to explore graph topology complementarily. Then, a coupled dual separation mechanism is employed for label disambiguation and robust optimization. Sample-level separation facilitates information exchange between the two branches, while parameter-level separation mitigates the risk of overfitting ambiguous labels.

and each hidden graph according to the number of common random walks within both graphs [27, 43].

In formal, given two graphs $G = (\mathcal{V}, \mathcal{E}, \mathbf{X})$ and $G' = (\mathcal{V}', \mathcal{E}', \mathbf{X}')$, their graph direct product $G_\times = (\mathcal{V}_\times, \mathcal{E}_\times)$ is a graph where $\mathcal{V}_\times = \{(v, v') \,|v \in \mathcal{V} \wedge v' \in \mathcal{V}'\}$ and $E_\times = \{\{(v, v'), (u, u')\} \,|\{v, u\} \in \mathcal{E} \wedge \{v', u'\} \in \mathcal{E}'\}$. Then, a simultaneous walk on graphs G and G' can be achieved by a random walk on $G_\times$. Therefore, the q-step ($q \in \mathbb{N}$) random walk kernel between G and G' that counts the number of simultaneous walks of length q can be calculate as:

$$k^{(q)}\left(G, G'\right) = \sum_{i=1}^{|\mathcal{V}_\times|} \sum_{j=1}^{|\mathcal{V}'_\times|} \left[\mathbf{A}_\times^q\right]_{ij}, \tag{5}$$

where $\mathbf{A}_\times$ denotes the adjacency matrix of $G_\times$. To further take the node attributes into account, we connect each node (v, v') of $G_\times$ with the similarity between the attributes of v and v'. Formally, the similarity scores for all the nodes of $G_\times$ can be calculated as $\mathbf{s} = \mathrm{Flatten}(\mathbf{X}\mathbf{X}'^T) \in \mathbb{R}^{|\mathcal{V}||\mathcal{V}'|}$, and then the kernel function can be extended as:

$$k^{(q)}\left(G, G'\right) = \sum_{i=1}^{|\mathcal{V}_\times|} \sum_{j=1}^{|\mathcal{V}'_\times|} \mathbf{s}_i \mathbf{s}_j \left[\mathbf{A}_\times^q\right]_{ij}. \tag{6}$$

Finally, given the set of parameterized hidden graphs $\{G'_m\}_{m=1}^M$ and a sequence of increasing lengths $\{0, \ldots, Q\}$, we can obtain a kernel matrix $\mathbf{K} \in \mathbb{R}^{M \times (Q+1)}$, where $\mathbf{K}_{mq} = k^{(q)}(G, G'_m)$ for an input graph G. Let $\boldsymbol{z}^{(2)} = \mathrm{Flatten}(\mathbf{K})$ as learnable and interpretable features explicitly mining the input graph's structure, the predicted label distribution $\boldsymbol{p}^{(2)}$ of this branch can be attained through another MLP with a softmax activation function,

$$\boldsymbol{p}^{(2)} = f_{GKB}\left(G; \Phi\right) = \mathrm{MLP}_\phi(\boldsymbol{z}^{(2)}), \tag{7}$$

where Φ is the parameter set for the graph kernel branch.

3.3 Coupled Dual Separation for Disambiguation and Robust Optimization

Ambiguity and noise in the candidate label set may lead to error accumulation within each branch, thereby compromising performance. To mitigate the impact of this ambiguity and fully exploit the graph mining capabilities of both branches, we formalize a coupled dual separation framework, which contains both sample-level and parameter-level separation for effective disambiguation and robust optimization.

Sample-level Separation. Given that the two branches exhibit distinct learning capabilities and graph mining patterns, and can filter out different types of errors introduced by ambiguous labels, it is promising to facilitate organic information exchange between the two branches. To this end, we propose to identify *informative* data from each branch to guide the optimization of the other branch. This allows the reduction of error accumulation through mutual guidance between the two branches. Specifically, to identify informative samples, we design a novel metric considering three aspects: 1) sufficient training, 2) high confidence, and 3) high discrepancy [62] between the two branches. Formally, let $r \neq r' \in \mathcal{R} = \{1, 2\}$ denote the index for the two branches, this metric can be written as:

$$\begin{aligned}\Gamma_{\mathrm{SS}}^{(r)}(G_i) =& \ell_{\mathrm{CE}}(\tilde{\boldsymbol{p}}_i^{(r)}, \boldsymbol{p}_i^{(r)}) + \ell_{\mathrm{E}}(\hat{\boldsymbol{p}}_i^{(r)}) - \mathrm{JS}(\boldsymbol{p}_i^{(r)} \| \boldsymbol{p}_i^{(r')}) \\ =& -\tilde{\boldsymbol{p}}_i^{(r)} \log(\boldsymbol{p}_i^{(r)}) - \sum_{c \in \mathcal{Y}} \hat{\boldsymbol{p}}_i^{(r)}[c] \log(\hat{\boldsymbol{p}}_i^{(r)}[c]) \\ & - \frac{1}{2}\mathrm{KL}\left(\boldsymbol{p}_i^{(r)} \| \frac{\boldsymbol{p}_i^{(r)} + \boldsymbol{p}_i^{(r')}}{2}\right) \\ & - \frac{1}{2}\mathrm{KL}\left(\boldsymbol{p}_i^{(r')} \| \frac{\boldsymbol{p}_i^{(r)} + \boldsymbol{p}_i^{(r')}}{2}\right),\end{aligned} \tag{8}$$

$$\hat{\boldsymbol{p}}_{ic}^{(r)} = \begin{cases} \frac{\boldsymbol{p}_i^{(r)}[c]}{\sum_{c' \in Y_i} \boldsymbol{p}_i^{(r)}[c']} & \text{if} \quad c \in Y_i \\ 0 & \text{otherwise,} \end{cases} \tag{9}$$

where ℓ_{CE} denotes the cross-entropy loss, ℓ_{E} denotes the entropy, $\hat{\boldsymbol{p}}_i^{(r)}$ is the re-normalized distribution by clearing the probabilities outside the candidate set, and $\tilde{\mathbf{p}}_i^{(r)}$ is the temporally assembled distribution attained by the exponential moving average (EMA) strategy. JS and KL denote Jensen-Shannon divergence and Kullback–Leibler divergence, respectively. The three terms of $\Gamma_{\mathrm{SS}}^{(r)}$ correspond to the three aspects mentioned above: 1) The first term encourages that the current prediction $\boldsymbol{p}_i^{(r)}$ tends to the temporal assembling of the re-normalized distribution $\tilde{\boldsymbol{p}}_i^{(r)}$, suggesting that the predicted distribution for G_i is relatively stable and concentrated within the candidate set, thus demonstrating the sufficient training of this sample. 2) The second term is the entropy of the re-normalized predicted distribution $\hat{\boldsymbol{p}}_i^{(r)}$, reflecting the confidence of this prediction. 3) The third term is the negative Jensen-Shannon (JS) divergence between the predicted distributions from the two branches, identifying graph samples with high inter-branch discrepancy. Then, we separate $\mathcal{D}$ into an informative set $\mathcal{D}_{\mathcal{I}}^{(r)}$ and an uninformative set $\mathcal{D}_{\mathcal{U}}^{(r)}$ using $\Gamma_{\mathrm{SS}}^{(r)}$ for each branch,

$$\mathcal{D}_{\mathcal{I}}^{(r)} = \left\{ G_i \mid \Gamma_{\mathrm{SS}}^{(r)}(G_i) \leq \epsilon(\alpha) \right\}, \mathcal{D}_{\mathcal{U}}^{(r)} = \mathcal{D} / \mathcal{D}_{\mathcal{I}}^{(r)}, \tag{10}$$

where $\epsilon(\alpha)$ is determined by the α-percentile of informative scores $\Gamma_{\mathrm{SS}}^{(r)}$ for all the samples.

The informative information is exchanged [24, 41] between the two branches via generating one-hot pseudo labels of these informative samples for guidance:

$$\mathcal{L}_{\mathcal{I}}^{(r)} = \ell_{\mathrm{CE}}(\hat{\boldsymbol{y}}_i^{(r')}, \boldsymbol{p}_i^{(r)}) = -\frac{1}{|\mathcal{D}_{\mathcal{I}}^{(r')}|} \sum_{i=t}^{|\mathcal{D}_{\mathcal{I}}^{(r')}|} \hat{\boldsymbol{y}}_i^{(r')} \log(\boldsymbol{p}_i^{(r)}), \tag{11}$$

where $\hat{\boldsymbol{y}}_i^{(r')} = \max_{c \in Y_i} \left\{ \hat{\boldsymbol{p}}_i^{(r')}[c] \right\}$ is the pseudo label for G_i generated by the $(r')^{th}$ branch. For the uninformative samples, we minimize the probabilities outside the candidate set for more sufficient training,

$$\mathcal{L}_{\mathcal{U}}^{(r)} = -\frac{1}{|\mathcal{D}_{\mathcal{U}}^{(r)}|} \sum_{i=r}^{|\mathcal{D}_{\mathcal{U}}^{(r)}|} \sum_{c \notin Y_i} \log(1 - \boldsymbol{p}_i^{(r)}[c]), \tag{12}$$

In summary, the overall loss objective for each branch is summarized into:

$$\mathcal{L}^{(r)} = \mathcal{L}_{\mathcal{I}}^{(r)} + \mathcal{L}_{\mathcal{U}}^{(r)}, r \in \{1, 2\}. \tag{13}$$

Parameter-level Separation. The overparameterization of deep networks can lead to overfitting of noisy and ambiguous labels [37, 55, 63]. Drawing inspiration from the "lottery ticket" hypothesis [16], which posits that generalization only depends on a key parameter subset, we propose to separate critical parameters from non-critical parameters and apply different update rules on them to alleviate the overfitting of noisy labels in the candidate set. Specifically, we assess the importance of parameters from two perspectives: 1) numerical magnitude. Critical parameters, playing a significant role in network propagation, generally exhibit larger values, as indicated in recent network pruning works [38]; 2) gradient magnitude. In the initial stages of training, parameters exhibiting greater gradient values are more likely to shape generalization behavior [7], making them more likely to induce intrinsic semantic patterns from ambiguous data. Hence, we use the product of a parameter's value and its gradient as a metric for identifying critical parameters. Formally, the importance of a parameter $\omega \in \Theta(\Phi)$ can be formalized as follows:

$$\Gamma_{\mathrm{PS}}^{(r)}(\omega) = \left| \omega \cdot \frac{\partial \mathcal{L}^{(r)}}{\partial \omega} \right|. \tag{14}$$

With the metric Γ_{PS}, we can separate the parameters in each branch respectively. In what follows, we take the message passing branch as an example to introduce the rules for the separation and updating of different parameters. A dynamic threshold is introduced to separate the parameters into the critical set Θ_c and the non-crucial set Θ_n, i.e.,

$$\Theta_c = \left\{ \omega \mid \Gamma_{\mathrm{PS}}^{(1)}(\omega) \geq \epsilon(\beta) \right\}, \Theta_n = \Theta / \Theta_c, \tag{15}$$

where $\epsilon(\beta)$ is determined by the β-percentile of importance scores $\Gamma_{\mathrm{PS}}^{(1)}$ for all the parameters in the message passing branch. For the critical parameter $\omega \in \Theta_c$, we update it following the common rule of the gradient descent,

$$\omega \leftarrow \omega - \lambda \frac{\partial \mathcal{L}^{(1)}}{\partial \omega}, \tag{16}$$

where λ represents the learning rate. As for the non-critical parameter $\omega \in \Theta_n$, we employ rigid decay to shrink it,

$$\omega \leftarrow \omega - \lambda \operatorname{sgn}(\omega), \tag{17}$$

where sgn is the standard sign function. Compared to the pruning methods [16], the shrinking strategy enables our model to adaptively retain some non-critical parameters softly, mitigating precision loss. Similar procedures can be adopted to optimize the graph kernel branch. Through the parameter-level separation and corresponding update strategies, each branch can fully exploit the critical parameters to better induce intrinsic graph semantic patterns associated with potential ground-truth labels in the candidate sets. This results in the effects of label disambiguation and robust optimization, enhancing the model's generalization capability. The overall training algorithm of CODE is provided in Appendix B.

Theoretical Analysis. We derive the theoretical guarantee for the proposed coupled dual separation framework. Denote $\omega^* = \arg\min_\omega \mathcal{L}(\omega)$ and the derivatives of the loss function with respect to crucial and non-crucial parameters by $\nabla_c \mathcal{L}$ and $\nabla_n \mathcal{L}$, respectively. In what follows, the parameter at step t is denoted by ω_t; the lower scripts c in $\omega_{t,c}$ and ω_c^* stands for the crucial parts of ω_t and ω^*; and the lower scripts n in $\omega_{t,n}$ and ω_n^* stands for the non-crucial parts of ω_t and ω^*, respectively. We shall make the following assumptions:

Assumption 1. *There exists a $T \in \mathbb{N}^+$ and positive constants $C > 0, \lambda > 0$ such that $\|\omega_t - \omega^*\|_\infty \leq C\lambda$ for all parameters ω_t and $t \geq T$, which means the parameter updates stabilize within a small range after sufficient iterations. The distribution of each non-critical parameter $\omega_{t,n}$ is identically distributed around ω_n^*. $\mathcal{L}$ is convex in the local domain. In other words, ω^* is the global minimizer for analysis. $\nabla\mathcal{L}$ is L-Lipschitz continuous, i.e. there exists $L > 0$ such that for all $\omega_1, \omega_2 \in \Theta$,*

$$\|\nabla\mathcal{L}(\omega_1) - \nabla\mathcal{L}(\omega_2)\|_2 \leq L\|\omega_1 - \omega_2\|_2. \tag{18}$$

Table 1: Classification accuracy (mean%±std%) is reported across four standard graph benchmarks, with the best performance shown in bold and the second-best underlined. The parameter q reflects the extent of label ambiguity.

Dataset	ENZYMES			Letter-High			CIFAR10			COIL-DEL		
Methods	$q=0.1$	$q=0.3$	$q=0.5$	$q=0.1$	$q=0.3$	$q=0.5$	$q=0.1$	$q=0.3$	$q=0.5$	$q=0.02$	$q=0.05$	$q=0.1$
GCN	61.33±2.85	48.44±2.06	40.22±2.93	50.09±0.70	44.00±1.08	35.94±1.82	47.18±1.09	43.68±0.68	41.35±0.65	60.77±1.71	50.43±1.07	41.63±1.74
GAT	58.22±3.03	49.11±2.93	34.67±3.87	73.39±1.41	61.33±3.48	53.04±3.06	57.56±0.65	52.93±1.22	48.54±0.46	69.11±2.86	59.77±1.97	46.63±1.54
GIN	59.78±4.58	47.11±4.59	34.22±1.78	55.83±4.28	50.43±1.92	35.59±3.75	47.29±0.61	43.91±0.45	41.24±0.52	55.94±1.69	46.23±0.88	37.29±1.04
GraphSAGE	60.89±1.09	47.33±3.03	39.33±3.11	78.20±1.17	70.96±1.48	60.35±1.83	57.22±0.67	51.92±0.26	47.44±0.83	71.40±2.15	58.91±1.92	49.23±1.90
TopKPool	53.11±4.12	44.22±2.76	36.00±4.80	67.07±1.60	55.25±2.74	43.83±5.21	55.26±0.85	48.97±1.24	42.87±1.31	55.80±4.86	44.83±2.19	34.63±2.08
SAGPool	56.89±5.37	46.67±2.53	37.11±5.00	67.42±1.91	55.71±4.71	39.30±5.49	54.23±0.53	50.01±0.68	45.16±0.36	52.94±2.59	41.89±4.28	30.17±1.85
EdgePool	58.67±2.67	51.11±3.06	33.33±1.99	70.49±3.29	64.17±2.44	55.36±2.16	55.09±0.61	50.17±0.64	45.90±0.44	68.74±1.85	56.74±3.98	45.89±1.30
ASAP	60.89±2.67	44.44±3.06	31.56±3.34	71.25±1.44	65.04±1.22	52.75±4.41	54.56±0.66	50.10±0.63	44.81±1.57	59.03±3.09	46.20±4.08	34.94±3.02
Graph Transplant	61.56±2.86	51.78±2.39	43.78±3.41	80.75±0.60	74.84±1.44	66.78±1.86	56.87±1.28	53.79±1.11	48.95±1.47	80.09±0.75	66.57±1.60	57.11±1.03
PiCO	61.08±6.67	46.88±2.76	35.78±3.02	81.27±1.60	73.56±1.71	64.63±4.35	57.70±0.82	53.47±1.14	46.04±1.20	84.88±1.09	76.25±1.66	63.69±1.42
TGNN	62.44±3.01	53.33±3.51	42.22±4.39	78.55±0.78	70.43±0.97	59.83±1.32	OOM	OOM	OOM	70.49±0.87	62.28±1.05	50.20±1.17
GraphCL	61.78±1.51	54.22±5.14	39.78±5.09	78.43±0.85	72.00±2.01	62.49±2.00	57.62±0.56	53.57±0.87	48.10±0.61	78.83±1.06	69.94±2.31	60.17±2.96
GraphACL	58.22±1.51	54.44±2.33	44.89±4.75	81.04±1.01	69.80±1.01	57.68±2.85	57.65±0.21	53.25±0.62	47.86±0.65	80.66±0.41	71.40±0.95	60.29±3.04
CODE (Ours)	**64.00**±3.19	**61.56**±2.29	**50.67**±2.80	**83.89**±0.62	**81.04**±2.22	**73.07**±1.97	**60.30**±0.64	**58.03**±0.50	**54.62**±0.90	**86.49**±1.33	**82.46**±0.95	**73.69**±2.27

$\nabla \mathcal{L}_n(\omega_{t,n})^\top \operatorname{sgn}(\omega_{t,n}) \geq 0$, *which means the non-critical parameters tend to decay through the gradient descent w.r.t. the loss function.* $\mathcal{L}(\omega_{t+1}) \leq \mathcal{L}(\omega_t) + \lambda$ *during the training; otherwise, early stopping is adopted.*

For the updating strategy employed, the following theorem shows that our proposed CODE can achieve a solution with fewer yet crucial parameters while ensuring the difference in the loss function is bounded.

Theorem 1. *Under Assumptions 1, then*

$$\mathcal{L}\left(\omega_t\right) \xrightarrow{\text{a.s.}} \mathcal{L}\left(\omega^*\right). \tag{19}$$

The proof of Theorem 1 can be found in Appendix C. In the case of partial label graph learning, common gradient descent may converge to an optimal solution ω^* w.r.t. noisy observations, but it may be not the best solution for the underlying true data, because it has a large number of parameters that can potentially overfit to noisy labels [16, 55]. Our CODE's strategy does not aim to directly converge to ω^*, but instead obtains a solution with fewer activated parameters while ensuring the difference in the loss function is bounded, which potentially approaches the underlying true model $\hat{\omega}$ more closely by alleviating overfitting to noisy candidate labels.

3.4 Computational Complexity Analysis

Let L be the layer number of message passing branch, d is the feature dimension, Q is the longest length of random walk, and M is the number of hidden graphs. The message passing branch takes $O\left(L|\mathcal{E}|d + L|\mathcal{V}|d^2\right)$ computional time while the graph kernel branch takes $O\left(Qd\left(M|\mathcal{V}'|(|\mathcal{V}'| + |\mathcal{V}|) + |\mathcal{E}|\right)\right)$ for each graph. The coupled dual separation process takes $O\left(d\right)$ for each graph. Since $|\mathcal{V}'|$, Q, and M are small, which can be regarded as constant terms, the total complexity of CODE is linearly related to $|\mathcal{E}|$, same as most graph learning methods.

4 Experiments

4.1 Experimental Setup

Datasets. We evaluate our framework on four graph benchmarks: ENZYMES [49], Letter-High [46], CIFAR10 [13], and COIL-DEL [46]. Following [21], we introduce partial label noise by adding false positive labels with probability q. We set $q \in \{0.1, 0.3, 0.5\}$ for ENZYMES, Letter-High, and CIFAR10, and $q \in \{0.02, 0.05, 0.1\}$ for COIL-DEL. More details of the datasets are in Appendix D.

Baseline Methods. We compare our proposed CODE with a variety of competitive baselines: (a) Graph neural network methods: GCN [60], GAT [54], GIN [64], and GraphSAGE [23]; (b) Hierarchical graph pooling methods: TopKPool [17], SAGPool [31], EdgePool [10], and ASAP [45]; (c) Graph augmentation method: Graph Transplant [44]; (d) Unsupervised graph learning method: GraphACL [40]; (e) Weakly-supervised graph learning method: TGNN [27]; (f) Partial label learning method in computer vision: PiCO [56], which we equip PiCO with a GraphSAGE-based encoder for a fair comparison with our method. More details about the baselines can be found in Appendix E.

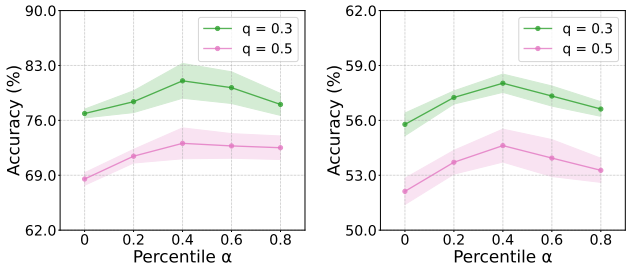
Implementation Details. All the methods are implemented utilizing the PyTorch. For the proposed CODE, We employ GraphSAGE [23] as the backbone of the message passing branch. The number of GraphSAGE layers is set to 2, and the hidden dimension is set to 512. For our graph kernel branch, we empirically set the number of hidden graphs to 8 and their size to 5 nodes. The maximum length of the random walk is set to 3. We use Adam [29] optimizer to optimize both two branches and set the batch size to 64. The total number of epochs is set to 100. We report the mean classification accuracy and standard deviation over five runs.

4.2 Performance Comparison

Table 1 reports the results of our proposed CODE compared to competitive baselines under various settings of label ambiguity q. The following observations can be made: (1) Overall, our CODE exhibits significant improvements over previous state-of-the-art

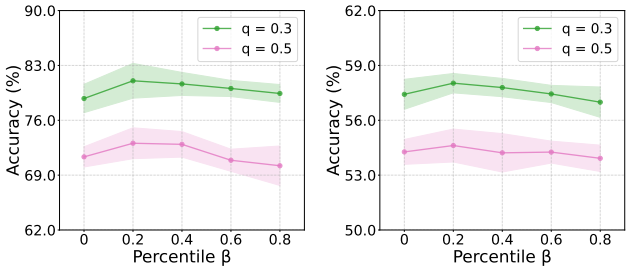
Table 2: Ablation study for CODE's key components. MPB, GKB, SLS, and PLS correspond to the message-passing branch, the graph kernel branch, the sample-level separation, and the parameter-level separation, respectively.

	Correlations				ENZYMES		Letter-High		CIFAR10		COIL-DEL	
	MPB	GKB	SLS	PLS	$q = 0.3$	$q = 0.5$	$q = 0.3$	$q = 0.5$	$q = 0.3$	$q = 0.5$	$q = 0.05$	$q = 0.1$
M_1	√				52.89	39.56	71.65	60.52	50.98	47.20	59.49	48.91
M_2		√			53.56	42.44	73.91	63.01	50.85	44.49	56.26	44.97
M_3	√	√			55.11	47.11	75.01	64.52	53.08	50.34	63.00	51.97
M_4	√	√	√		60.67	49.56	78.43	71.33	57.42	54.27	79.69	72.09
M_5 (Full model)	√	√	√	√	**61.56**	**50.67**	**81.04**	**73.07**	**58.03**	**54.62**	**82.46**	**73.69**



(a) Letter-High (b) CIFAR10

Figure 2: Performance *w.r.t.* the sample-level separation percentile α on Letter-High and CIFAR10.



(a) Letter-High (b) CIFAR10

Figure 3: Performance *w.r.t.* the parameter-level separation percentile β on Letter-High and CIFAR10.

methods in all scenarios, with substantial gains. For instance, compared to the latest graph contrastive learning method GraphACL, our CODE achieves an improvement of 11.9% and 15.4% for the average accuracy on ENZYMES and Letter-High, respectively. This suggests the effectiveness of our coupled dual separation mechanism in fully exploiting weak supervision from partial labels. (2) When compared to the representative partial label learning method PiCO in the computer vision domain, our proposed CODE shows higher average classification accuracy on CIFAR10, and COIL-DEL by 10.5% and 8.5%, respectively. This improvement is attributed to our CODE's effective exploration of complementary graph semantic patterns through the organic combination of the coupled branches, although these datasets are extracted from the computer vision domain by superpixels. (3) The performance of all the methods tends to decrease as the label ambiguity q increases, due to weaker and noisier supervision signals. However, as q grows, our proposed CODE demonstrates a slower decline in accuracy compared with other methods, showcasing its significant robustness to label ambiguity, owing to the alleviation of overfitting to ambiguous labels through the parameter-level separation. More comparison results can be found in Appendix F.

4.3 Ablation Study

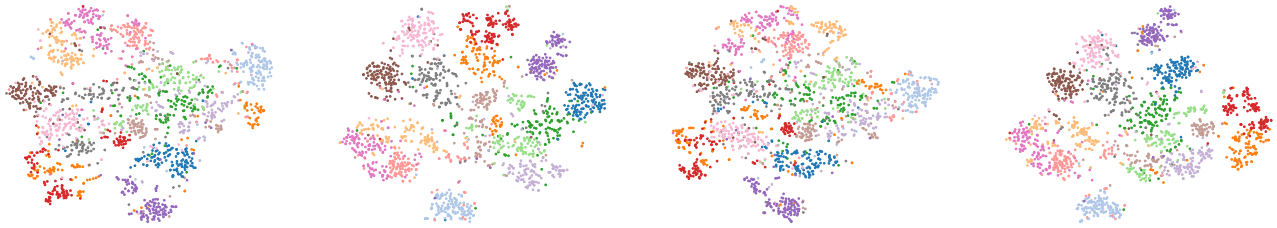
To investigate the effectiveness of various key components, we configure several variants of CODE for comparison: (1) M_1 only contains the message passing branch; (2) M_2 only contains the graph kernel branch; (3) M_3 employs both branches; (4) M_4 further introduces the sample-level separation; (5) The difference between M_5 (our full model) and M_4 lies in the usage of the parameter-level separation. The results are reported in Table 2. It can be found that M_3 outperforms both M_1 and M_2, which benefits from the complementary graph semantics mined from the two branches. After introducing the sample-level separation, M_4 achieves better results than M_3, because the separated informative samples guide the optimization of the other branch mutually, reducing the error accumulation caused by label ambiguity. By comparing the results of M_5 and M_4, we can observe that the parameter-level separation can further improve the performance of our model by mitigating the risk of overfitting noisy labels in the candidate set.

4.4 Hyper-parameter Analysis

We study the influence of the sample-level separation percentile α and the parameter-level separation percentile β on Letter-High and CIFAR10. From Figure 2, the accuracy reaches the high point as the sample-level separation percentile α equals 0.4 on all settings, since only a moderate number of samples are informative for guiding the optimization of the other branch under label ambiguity. Selecting too few informative samples will lead to insufficient optimization guidance for the other branch while selecting too many samples for information exchange will introduce the noise of uninformative samples. Besides, as we can observe from Figure 3, with the increase of the parameter-level separation percentile β, the performance of our proposed framework CODE first improves and then deteriorates, because the too-small β may suffer from overfitting the ambiguous and noisy labels in the candidate set, while too-large β leads to the loss of the model's expressive power with insufficient parameters retained.

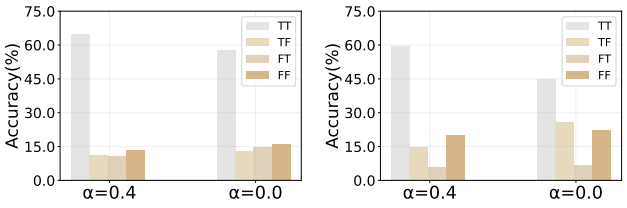
4.5 Analysis of Sample Separation Mechanism

We further investigate the power of sample-level separation by dividing the predictions into four categories: 1) TT, where both the message passing branch and the graph kernel branch give accurate predictions. 2) TF, where the message passing branch delivers a correct prediction, while the graph kernel branch fails. 3) FT, where the



(a) Baseline ($q = 0.3$) (b) CODE ($q = 0.3$) (c) Baseline ($q = 0.5$) (d) CODE ($q = 0.5$)

Figure 4: Visualization of graph representations generated by the message passing branch on Letter-High using t-SNE.



(a) Letter-High ($q = 0.5$) (b) COIL-DEL ($q = 0.1$)

Figure 5: Analysis of sample separation mechanism.

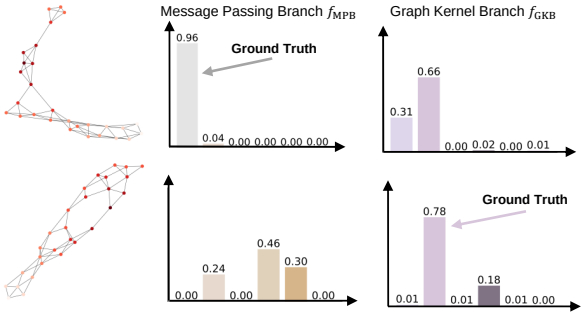


Figure 6: Case study for the complementarity of the two branches.

graph kernel branch offers a correct prediction, while the message passing branch fails. 4) FF, where neither branch succeeds in providing an accurate prediction. Figure 5 shows the results on Letter-High and COIL-DEL, with different sample separation percentiles. We can see that instead of treating all samples as non-informative ones ($\alpha = 0.0$), identifying moderate informative samples ($\alpha = 0.4$) from one branch to guide the other branch's optimization allows more accurate information exchange between the two branches and reduction of error accumulation within each branch.

4.6 Case Study and Visualization

We examine the ability of our two branches to extract complementary graph semantic information. Figure 6 illustrates two cases in ENZYMES, where it can be observed that the upper sample is only classified correctly by the message passing branch, while the bottom sample is only classified correctly by the graph kernel branch. This justifies the benefits of the complementary semantic patterns learned from the two branches under label ambiguity. We encourage organic information exchange between the two branches by the sample-level separation mechanism to leverage the complementary graph semantic information and reduce error accumulation

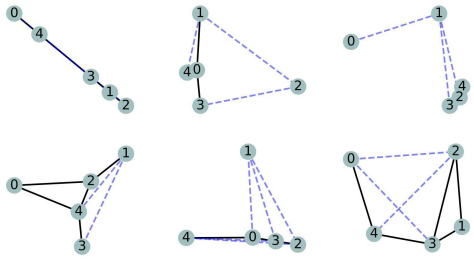


Figure 7: Visualization of hidden graphs learned by the graph kernel branch on Letter-High ($q = 0.5$). The solid lines correspond to edges whose weights are larger than 0.1, while the dashed lines correspond to edges whose weights are in the range of $[0.01, 0.1]$.

within each branch. We further explore the learned graph semantic patterns of the two branches on Letter-High by visualization. On the one hand, we employ t-SNE [53] to project the graph representations learned by our CODE's message passing branch and the best baseline Graph Transplant into a two-dimensional space. As depicted in Figure 4, our model exhibits clearer cluster structures and class boundaries, even in the scenario with high label ambiguity ($q = 0.5$). For example, the red points scatter to other classes in the baseline while gathering separately in our CODE. On the other hand, we visualize the hidden graphs learned by the graph kernel branch in Figure 7. It can be observed that this branch can capture various graph structures, thereby providing rich structural semantic information beneficial for classification.

5 Conclusion

This work studies partial label graph learning and thus introduces a new method called CODE for this problem. CODE contains a message passing branch and a graph kernel branch to explore graph semantics from implicit and explicit views, respectively. To enhance the message exchange from complementary views, we identify informative graphs using one branch to provide reliable guidance for the other branch. In addition, we separate network parameters in both branches to employ different optimization procedures for reducing the risk of overfitting. Extensive experiments on four graph benchmark datasets demonstrate the superiority of our proposed CODE. In future work, we would extend the proposed CODE to other practical scenarios such as semi-supervised graph learning and robust graph learning.

Acknowledgements

This paper is partially supported by the National Natural Science Foundation of China with Grant Number 62276002.

References

[1] Axel D Becke. 2014. Perspective: Fifty years of density-functional theory in chemical physics. *The Journal of chemical physics* 140, 18 (2014), 18A301.

[2] Karsten M Borgwardt and Hans-Peter Kriegel. 2005. Shortest-path kernels on graphs. In *Fifth IEEE international conference on data mining (ICDM'05)*. IEEE, 8–pp.

[3] Ricardo Cerri, Rodrigo C Barros, André C PLF de Carvalho, and Yaochu Jin. 2016. Reduction strategies for hierarchical multi-label classification in protein function prediction. *BMC bioinformatics* 17, 1 (2016), 1–24.

[4] Dexiong Chen, Laurent Jacob, and Julien Mairal. 2020. Convolutional kernel networks for graph-structured data. In *International Conference on Machine Learning*. PMLR, 1576–1586.

[5] Dapeng Chen, Min Wang, Haobin Chen, Lin Wu, Jing Qin, and Wei Peng. 2022. Cross-modal retrieval with heterogeneous graph embedding. In *Proceedings of the 30th ACM International Conference on Multimedia*. 3291–3300.

[6] Henry Chermette. 1999. Chemical reactivity indexes in density functional theory. *Journal of computational chemistry* 20, 1 (1999), 129–154.

[7] Brian Chmiel, Liad Ben-Uri, Moran Shkolnik, Elad Hoffer, Ron Banner, and Daniel Soudry. 2020. Neural gradients are near-lognormal: improved quantized and sparse training. *arXiv preprint arXiv:2006.08173* (2020).

[8] Gabriele Corso, Luca Cavalleri, Dominique Beaini, Pietro Liò, and Petar Veličković. 2020. Principal neighbourhood aggregation for graph nets. *Advances in Neural Information Processing Systems* 33 (2020), 13260–13271.

[9] Timothee Cour, Ben Sapp, and Ben Taskar. 2011. Learning from partial labels. *The Journal of Machine Learning Research* 12 (2011), 1501–1536.

[10] Frederik Diehl. 2019. Edge contraction pooling for graph neural networks. *arXiv preprint arXiv:1905.10990* (2019).

[11] Simon S Du, Kangcheng Hou, Russ R Salakhutdinov, Barnabas Poczos, Ruosong Wang, and Keyulu Xu. 2019. Graph neural tangent kernel: Fusing graph neural networks with graph kernels. *Advances in neural information processing systems* 32 (2019).

[12] Zilin Du, Yunxin Li, Xu Guo, Yidan Sun, and Boyang Li. 2023. Training multimedia event extraction with generated images and captions. In *Proceedings of the 31st ACM International Conference on Multimedia*. 5504–5513.

[13] Vijay Prakash Dwivedi, Chaitanya K Joshi, Thomas Laurent, Yoshua Bengio, and Xavier Bresson. 2020. Benchmarking graph neural networks. *arXiv preprint arXiv:2003.00982* (2020).

[14] Xiaomin Fang, Lihang Liu, Jieqiong Lei, Donglong He, Shanzhuo Zhang, Jingbo Zhou, Fan Wang, Hua Wu, and Haifeng Wang. 2022. Geometry-enhanced molecular representation learning for property prediction. *Nature Machine Intelligence* 4, 2 (2022), 127–134.

[15] Lei Feng and Bo An. 2019. Partial Label Learning by Semantic Difference Maximization.. In *IJCAI*. 2294–2300.

[16] Jonathan Frankle and Michael Carbin. 2018. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635* (2018).

[17] Hongyang Gao and Shuiwang Ji. 2019. Graph u-nets. In *international conference on machine learning*. PMLR, 2083–2092.

[18] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. 2017. Neural message passing for quantum chemistry. In *International Conference on Machine Learning*.

[19] Chen Gong, Tongliang Liu, Yuanyan Tang, Jian Yang, Jie Yang, and Dacheng Tao. 2017. A regularization approach for instance-based superset label learning. *IEEE transactions on cybernetics* 48, 3 (2017), 967–978.

[20] Yiyang Gu, Binqi Chen, Zihao Chen, Ziyue Qiao, Xiao Luo, Junyu Luo, Zhiping Xiao, Wei Ju, and Ming Zhang. 2025. MATE: Masked Optimal Transport with Dynamic Selection for Partial Label Graph Learning. *Artificial Intelligence* (2025), 104396.

[21] Yiyang Gu, Zihao Chen, Yifang Qin, Zhengyang Mao, Zhiping Xiao, Wei Ju, Chong Chen, Xian-Sheng Hua, Yifan Wang, Xiao Luo, et al. 2024. DEER: Distribution divergence-based graph contrast for partial label learning on graphs. *IEEE Transactions on Multimedia* (2024).

[22] Hongyu Guo and Yongyi Mao. 2023. Interpolating Graph Pair to Regularize Graph Classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 37. 7766–7774.

[23] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. *Advances in neural information processing systems* 30 (2017).

[24] Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama. 2018. Co-teaching: Robust training of deep neural networks with extremely noisy labels. *Advances in neural information processing systems* 31 (2018).

[25] Eyke Hüllermeier and Jürgen Beringer. 2005. Learning from ambiguously labeled examples. In *Advances in Intelligent Data Analysis VI: 6th International Symposium on Intelligent Data Analysis, IDA 2005, Madrid, Spain, September 8-10, 2005. Proceedings 6*. Springer, 168–179.

[26] Wei Ju, Yiyang Gu, Xiao Luo, Yifan Wang, Haochen Yuan, Huasong Zhong, and Ming Zhang. 2023. Unsupervised graph-level representation learning with hierarchical contrasts. *Neural Networks* 158 (2023), 359–368.

[27] Wei Ju, Xiao Luo, Meng Qu, Yifan Wang, Chong Chen, Minghua Deng, Xian-Sheng Hua, and Ming Zhang. 2023. TGNN: A joint semi-supervised framework for graph-level classification. *arXiv preprint arXiv:2304.11688* (2023).

[28] Wei Ju, Junwei Yang, Meng Qu, Weiping Song, Jianhao Shen, and Ming Zhang. 2022. Kgnn: Harnessing kernel-based networks for semi-supervised graph classification. In *Proceedings of the fifteenth ACM international conference on web search and data mining*. 421–429.

[29] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[30] Nils M Kriege, Fredrik D Johansson, and Christopher Morris. 2020. A survey on graph kernels. *Applied Network Science* 5, 1 (2020), 1–42.

[31] Junhyun Lee, Inyeop Lee, and Jaewoo Kang. 2019. Self-attention graph pooling. In *International conference on machine learning*. PMLR, 3734–3743.

[32] Manling Li, Alireza Zareian, Qi Zeng, Spencer Whitehead, Di Lu, Heng Ji, and Shih-Fu Chang. 2020. Cross-media Structured Common Space for Multimedia Event Extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 2557–2568.

[33] Sihang Li, Xiang Wang, An Zhang, Yingxin Wu, Xiangnan He, and Tat-Seng Chua. 2022. Let invariant rationale discovery inspire graph contrastive learning. In *International conference on machine learning*. PMLR, 13052–13065.

[34] Chuang Liu, Yibing Zhan, Jia Wu, Chang Li, Bo Du, Wenbin Hu, Tongliang Liu, and Dacheng Tao. 2022. Graph pooling for graph neural networks: Progress, challenges, and opportunities. *arXiv preprint arXiv:2204.07321* (2022).

[35] Kang Liu, Feng Xue, Dan Guo, Peijie Sun, Shengsheng Qian, and Richang Hong. 2023. Multimodal graph contrastive learning for multimedia-based recommendation. *IEEE Transactions on Multimedia* (2023).

[36] Meng Liu, Ke Liang, Dayu Hu, Hao Yu, Yue Liu, Lingyuan Meng, Wenxuan Tu, Sihang Zhou, and Xinwang Liu. 2023. Tmac: Temporal multi-modal graph learning for acoustic event classification. In *Proceedings of the 31st ACM International Conference on Multimedia*. 3365–3374.

[37] Sheng Liu, Zhihui Zhu, Qing Qu, and Chong You. 2022. Robust training under label noise by over-parameterization. In *International Conference on Machine Learning*. PMLR, 14153–14172.

[38] Zhuang Liu, Mingjie Sun, Tinghui Zhou, Gao Huang, and Trevor Darrell. 2018. Rethinking the value of network pruning. *arXiv preprint arXiv:1810.05270* (2018).

[39] Junyu Luo, Yiyang Gu, Xiao Luo, Wei Ju, Zhiping Xiao, Yusheng Zhao, Jingyang Yuan, and Ming Zhang. 2024. Gala: Graph diffusion-based alignment with jigsaw for source-free domain adaptation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 46, 12 (2024), 9038–9051.

[40] Xiao Luo, Wei Ju, Yiyang Gu, Zhengyang Mao, Luchen Liu, Yuhui Yuan, and Ming Zhang. 2023. Self-supervised graph-level representation learning with adversarial contrastive learning. *ACM Transactions on Knowledge Discovery from Data* 18, 2 (2023), 1–23.

[41] Xiao Luo, Wei Ju, Meng Qu, Chong Chen, Minghua Deng, Xian-Sheng Hua, and Ming Zhang. 2022. Dualgraph: Improving semi-supervised graph classification via dual contrastive learning. In *2022 IEEE 38th International Conference on Data Engineering (ICDE)*. IEEE, 699–712.

[42] Gengyu Lyu, Songhe Feng, Tao Wang, and Congyan Lang. 2020. A self-paced regularization framework for partial-label learning. *IEEE Transactions on Cybernetics* 52, 2 (2020), 899–911.

[43] Giannis Nikolentzos and Michalis Vazirgiannis. 2020. Random walk graph neural networks. *Advances in Neural Information Processing Systems* 33 (2020), 16211–16222.

[44] Joonhyung Park, Hajin Shim, and Eunho Yang. 2022. Graph transplant: Node saliency-guided graph mixup with local structure preservation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 36. 7966–7974.

[45] Ekagra Ranjan, Soumya Sanyal, and Partha Talukdar. 2020. Asap: Adaptive structure aware pooling for learning hierarchical graph representations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 5470–5477.

[46] Kaspar Riesen and Horst Bunke. 2008. IAM graph database repository for graph based pattern recognition and machine learning. In *Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR)*. Springer, 287–297.

[47] François Rousseau, Emmanouil Kiagias, and Michalis Vazirgiannis. 2015. Text categorization as a graph classification problem. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. 1702–1712.

[48] Stevan Rudinac, Iva Gornishka, and Marcel Worring. 2017. Multimodal Classification of Violent Online Political Extremism Content with Graph Convolutional Networks. In *Proceedings of the on Thematic Workshops of ACM Multimedia 2017* (Mountain View, California, USA) *(Thematic Workshops '17)*. Association for

Computing Machinery, New York, NY, USA, 245–252.
[49] Ida Schomburg, Antje Chang, Christian Ebeling, Marion Gremse, Christian Heldt, Gregor Huhn, and Dietmar Schomburg. 2004. BRENDA, the enzyme database: updates and major new developments. *Nucleic acids research* 32, suppl_1 (2004), D431–D433.
[50] Nino Shervashidze, Pascal Schweitzer, Erik Jan Van Leeuwen, Kurt Mehlhorn, and Karsten M Borgwardt. 2011. Weisfeiler-lehman graph kernels. *Journal of Machine Learning Research* 12, 9 (2011).
[51] Nino Shervashidze, SVN Vishwanathan, Tobias Petri, Kurt Mehlhorn, and Karsten Borgwardt. 2009. Efficient graphlet kernels for large graph comparison. In *Artificial intelligence and statistics*. PMLR, 488–495.
[52] Hui Tang and Xun Liang. 2023. Where to Find Fascinating Inter-Graph Supervision: Imbalanced Graph Classification with Kernel Information Bottleneck. In *Proceedings of the 31st ACM International Conference on Multimedia*. 3240–3249.
[53] Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of machine learning research* 9, 11 (2008).
[54] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903* (2017).
[55] Haixin Wang, Huiyu Jiang, Jinan Sun, Shikun Zhang, Chong Chen, Xian-Sheng Hua, and Xiao Luo. 2023. DIOR: Learning to hash with label noise via dual partition and contrastive learning. *IEEE Transactions on Knowledge and Data Engineering* 36, 4 (2023), 1502–1517.
[56] Haobo Wang, Ruixuan Xiao, Yixuan Li, Lei Feng, Gang Niu, Gang Chen, and Junbo Zhao. 2022. Pico: Contrastive label disambiguation for partial label learning. In *Proceedings of the International Conference on Learning Representations*.
[57] Min Wang, Hao Yang, and Qing Cheng. 2022. GCL: Graph Calibration Loss for Trustworthy Graph Neural Network. In *MM '22: The 30th ACM International Conference on Multimedia, Lisboa, Portugal, October 10 - 14, 2022*, João Magalhães, Alberto Del Bimbo, Shin'ichi Satoh, Nicu Sebe, Xavier Alameda-Pineda, Qin Jin, Vincent Oria, and Laura Toni (Eds.). ACM, 988–996. https://doi.org/10.1145/3503161.3548423
[58] Qifan Wang, Yinwei Wei, Jianhua Yin, Jianlong Wu, Xuemeng Song, and Liqiang Nie. 2021. Dualgnn: Dual graph neural network for multimedia recommendation. *IEEE Transactions on Multimedia* 25 (2021), 1074–1084.
[59] Wei Wang and Min-Ling Zhang. 2022. Partial label learning with discrimination augmentation. In *Proceedings of the International ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 1920–1928.
[60] Max Welling and Thomas N Kipf. 2016. Semi-supervised classification with graph convolutional networks. In *J. International Conference on Learning Representations (ICLR 2017)*.
[61] Jinyong Wen, Shiming Xiang, and Chunhong Pan. 2023. Exploring Universal Principles for Graph Contrastive Learning: A Statistical Perspective. In *Proceedings of the 31st ACM International Conference on Multimedia* (<conf-loc>, <city>Ottawa ON</city>, <country>Canada</country>, </conf-loc>) *(MM '23)*. Association for Computing Machinery, New York, NY, USA, 3579–3589. https://doi.org/10.1145/3581783.3612229
[62] Xiaobo Xia, Bo Han, Yibing Zhan, Jun Yu, Mingming Gong, Chen Gong, and Tongliang Liu. 2023. Combating noisy labels with sample selection by mining high-discrepancy examples. In *Proceedings of the IEEE/CVF international conference on computer vision*. 1833–1843.
[63] Xiaobo Xia, Tongliang Liu, Bo Han, Chen Gong, Nannan Wang, Zongyuan Ge, and Yi Chang. 2020. Robust early-learning: Hindering the memorization of noisy labels. In *International conference on learning representations*.
[64] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2018. How Powerful are Graph Neural Networks?. In *International Conference on Learning Representations*.
[65] Ning Xu, Jiaqi Lv, and Xin Geng. 2019. Partial label learning via label enhancement. In *Proceedings of the AAAI Conference on artificial intelligence*, Vol. 33. 5557–5564.
[66] Nan Yin, Li Shen, Baopu Li, Mengzhu Wang, Xiao Luo, Chong Chen, Zhigang Luo, and Xian-Sheng Hua. 2022. DEAL: An Unsupervised Domain Adaptive Framework for Graph-level Classification. In *MM '22: The 30th ACM International Conference on Multimedia, Lisboa, Portugal, October 10 - 14, 2022*. ACM, 3470–3479.
[67] Jaemin Yoo, Sooyeon Shim, and U Kang. 2022. Model-agnostic augmentation for accurate graph classification. In *Proceedings of the ACM Web Conference 2022*. 1281–1291.
[68] Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. 2020. Graph contrastive learning with augmentations. In *Advances in Neural Information Processing Systems*.
[69] Guoxian Yu, Hailong Zhu, and Carlotta Domeniconi. 2015. Predicting protein functions using incomplete hierarchical labels. *BMC bioinformatics* 16, 1 (2015), 1–12.
[70] Jin Yuan, Feng Hou, Yangzhou Du, Zhongchao Shi, Xin Geng, Jianping Fan, and Yong Rui. 2022. Self-Supervised Graph Neural Network for Multi-Source Domain Adaptation. In *MM '22: The 30th ACM International Conference on Multimedia, Lisboa, Portugal, October 10 - 14, 2022*. ACM, 3907–3916.
[71] Min-Ling Zhang and Fei Yu. 2015. Solving the partial label learning problem: An instance-based approach.. In *IJCAI*. 4048–4054.
[72] Zaixi Zhang, Qi Liu, Hao Wang, Chengqiang Lu, and Chee-Kong Lee. 2021. Motif-based graph self-supervised learning for molecular property prediction. *Advances in Neural Information Processing Systems* 34 (2021), 15870–15882.
[73] Fei Zhao, Chunhui Li, Zhen Wu, Shangyu Xing, and Xinyu Dai. 2022. Learning from Different text-image Pairs: A Relation-enhanced Graph Convolutional Network for Multimodal NER. In *MM '22: The 30th ACM International Conference on Multimedia, Lisboa, Portugal, October 10 - 14, 2022*. ACM, 3983–3992.
[74] Yu Zhou, Jianjun He, and Hong Gu. 2016. Partial label learning via Gaussian processes. *IEEE transactions on cybernetics* 47, 12 (2016), 4443–4450.