

PTCL: PSEUDO-LABEL TEMPORAL CURRICULUM LEARNING FOR LABEL-LIMITED DYNAMIC GRAPH

Anonymous authors

Paper under double-blind review

ABSTRACT

Dynamic node classification is critical for modeling evolving systems like financial transactions and academic collaborations. In such systems, dynamically capturing node information changes is critical for dynamic node classification, which usually requires all labels at every timestamp. However, it is difficult to collect all dynamic labels in real-world scenarios due to high annotation costs and label uncertainty (e.g., ambiguous or delayed labels in fraud detection). In contrast, final timestamp labels are easier to obtain as they rely on complete temporal patterns and are usually maintained as a unique label for each user in many open platforms, without tracking the history data. To bridge this gap, we propose a pioneering method **PTCL (Pseudo-label Temporal Curriculum Learning)**, combining the variational EM (Expectation Maximization) framework with a novel Temporal Curriculum Learning strategy to effectively leverage both final timestamp labels and pseudo-labels. We also contribute a new academic dataset **CoOAG (Co-authorship graph derived from Open Academic Graph)**, capturing long-range research interest in dynamic graph. Experiments across real-world scenarios demonstrate PTCL’s consistent superiority over other methods adapted to this task. Beyond methodology, we propose a unified framework **FLiD (Framework for Label-Limited Dynamic Node Classification)**, consisting of a complete preparation workflow, training pipeline, and evaluation standards, and supporting various models and datasets. Code details can be found in supplementary materials.

1 INTRODUCTION

Graph-structured data is widespread in domains such as social networks (Ying et al., 2018; Newman et al., 2002; Feng et al., 2022), biological systems (Zitnik et al., 2018; Li et al., 2022a), financial transactions (Huang et al., 2022; Li & Yang, 2023), and academic collaborations (Hu et al., 2021; Zhou et al., 2022). Graphs effectively capture relationships between entities, enabling powerful modeling of complex systems (Bronstein et al., 2017). A key task is node classification, which assigns labels to nodes based on features and structure (Xiao et al., 2022; Bhagat et al., 2011; Rong et al., 2019), and has been extensively studied in static graphs. However, many real-world graphs are dynamic, with evolving nodes, edges, and labels over time (Rossi et al., 2020; Kumar et al., 2019; Xu et al., 2020). For instance, authors may shift research areas (Jia et al., 2017) or users may change behaviors in transaction networks (Huang et al., 2022), highlighting the need for dynamic node classification that accounts for temporal label changes.

Ideally, training on a complete dynamic trajectory yields a strong classifier. However, dynamic node classification is challenged by the high cost of acquiring dynamic labels, requiring continuous monitoring, manual annotation, and coping with uncertainty (e.g., delayed or ambiguous fraud labels). Existing dynamic datasets often provide weak or rarely changing labels (Kumar et al., 2019), limiting their ability to reflect evolving node behavior. While labeling nodes at every timestamp is difficult, obtaining a final label (e.g., fraud status) at the end of a period is more feasible, as illustrated in Figure 1 (Huang et al., 2022). Many platforms, such as OAG (Open Academic Graph) (Sinha et al.; Zhang et al., b;a; Tang et al.), also offer only final static labels (e.g., fixed research interests). This motivates our core task: **label-limited dynamic node classification**. The goal is to classify nodes in dynamic graphs using limited label information, especially final timestamp labels, which implicitly summarize long-term behavior. Effectively leveraging unlabeled historical data thus requires robust modeling of temporal dynamics.

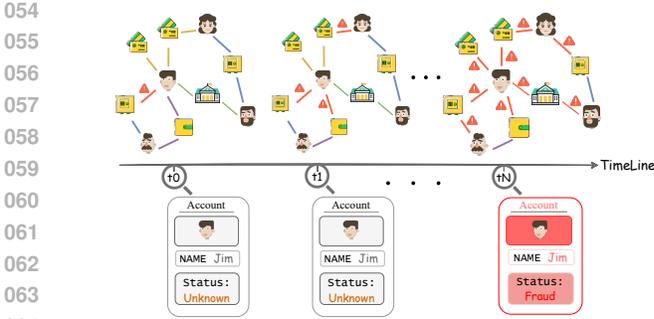


Figure 1: A representation of a dynamic financial system. The graph models entities such as users, payment cards, and financial institutions as nodes, with edges denoting transactional relationships. Over time, user behavior evolves through sequences of transactions, forming dynamic interaction patterns. Some users’ labels (e.g., account status) may eventually be revealed as fraudulent based on long-term behavioral signals.

Building on the foundation of SSL (Semi-Supervised Learning) (Yang et al., 2022; Zhou et al., 2020; Zhu, 2005; Learning, 2006; Chapelle et al., 2006; Van Engelen & Hoos, 2020) and pseudo-labeling (Lee et al., 2013; Kage et al., 2024), we propose **PTCL** (Pseudo-label Temporal Curriculum Learning), which consists of a dynamic graph encoder (backbone) and a label predictor (decoder) (Yu et al., 2023; Rossi et al., 2020). Inspired by the variational EM framework (Qu, 2024; Qu et al., 2019; Zhao et al., 2022; Neal & Hinton, 1998; Dempster et al., 1977), we decouple their optimization: the decoder is trained solely on final timestamp labels to ensure label fidelity, then used to generate pseudo-labels for earlier timestamps. These pseudo-labels, combined with the final labels, guide the backbone training via a weighted loss, allowing it to learn temporal label dynamics effectively. To mitigate the impact of low-quality pseudo-labels in early EM iterations, we introduce a **Temporal Curriculum Learning** strategy inspired by **Curriculum Learning** (Bengio et al., 2009; Wang et al., 2021b; Soviany et al., 2021). Following the easy-to-hard principle, we assign higher weights to pseudo-labels near the final timestamps—where predictions are more reliable—at early stages of training. Over time, the model gradually incorporates earlier, harder timestamps, enabling progressive learning of temporal dynamics.

Extensive Experiments show that PTCL consistently outperforms other methods adapted to this task across multiple datasets, validating its effectiveness in capturing the temporal evolution of nodes. Additionally, we conduct a series of studies to verify the contribution of each design of PTCL.

To sum up, our contributions are as follows:

- **Pioneering Study:** To the best of our knowledge, this is a pioneering study to systematically investigate the problem of label-limited dynamic node classification. We formalize the task, identify its unique challenges, and propose a comprehensive method to address them.
- **Novel Method:** We introduce a new method PTCL, which captures the dynamic nature of nodes with limited labels, advancing dynamic graph study by constructing highly varying history information. Various experiments demonstrate the effectiveness of PTCL and the necessity of each design.
- **New Dataset:** We contribute a new dataset, CoOAG, which is derived from academic collaboration networks and designed for dynamic graph learning. It captures the dynamic nature of research interests, providing a rich testbed for evaluating PTCL.
- **Unified Framework:** We propose a unified framework, FLiD (**F**ramework for **L**abel-Limited **D**ynamic Node Classification), for our task, which includes a complete preparation workflow, a training pipeline, and evaluation protocols. Our framework supports various backbones and datasets, offering a flexible and extensible solution.

2 PROBLEM FORMULATION

A **dynamic graph with dynamic labels** can be mathematically represented as a sequence of chronologically ordered events: $\mathcal{G} = \{x(t_i)\} = \{(u_i, v_i, t_i)\}$, where $0 \leq t_1 \leq t_2 \leq \dots$. Each event $x(t_i)$ describes an interaction between a source node $u_i \in \mathcal{V}$ and a destination node $v_i \in \mathcal{V}$ at time t_i . And $y_{u_i}^{t_i}, y_{v_i}^{t_i} \in \mathcal{Y}$ are their respective labels at t_i . \mathcal{V} denotes the set of all nodes, and \mathcal{Y} is the class set of all nodes. For each node $u \in \mathcal{V}$, $\mathcal{T}_u = \{t_i \mid u = u_i \text{ or } u = v_i \text{ in } x(t_i) \in \mathcal{G}\}$ is the set of all timestamps at which u participates in any event in \mathcal{G} . The last occurrence time $T_u = \max \mathcal{T}_u$ is the most recent timestamp in \mathcal{T}_u . We further define $\mathcal{Y}_F = \{y_u^{T_u} \mid u \in \mathcal{V}\}$, the set of ground-truth labels at the final timestamps, and $\mathcal{Y}_E = \{y_u^t \mid u \in \mathcal{V}, t \in \mathcal{T}_u \setminus \{T_u\}\}$, the set of labels at non-last timestamps

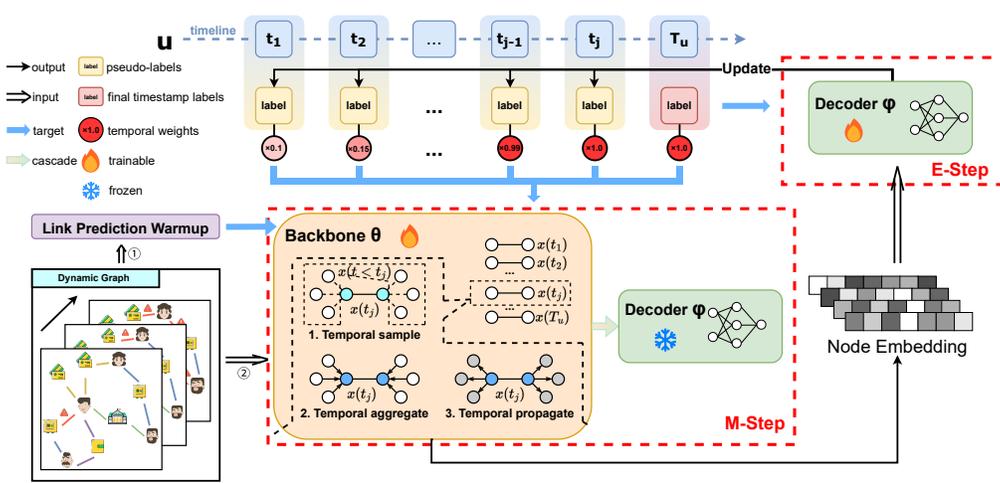


Figure 2: Overview of our proposed method. PTCL consists of a Variational EM process with a dynamic graph backbone and a decoder. During the warmup phase, the dynamic graph backbone is trained on a link prediction task, where the dynamic graph structure serves as the target. After warmup, in each M-step, the backbone receives final timestamp labels, pseudo-labels, and the dynamic graph structure as input, while the decoder is trained in the E-step to refine pseudo-labels. Additionally, the Temporal Curriculum Learning strategy prioritizes pseudo-labels based on their temporal proximity to the final timestamp labels, ensuring higher-quality training.

of every node. In most cases, $|\mathcal{Y}_E| \gg |\mathcal{Y}_F|$. In our research scenario, \mathcal{Y}_F are known, whereas \mathcal{Y}_E are considered unknown due to data collection constraints. Following the training-evaluation paradigm, we determine a boundary time T_B to separate the training and evaluation datasets. The final timestamp label set \mathcal{Y}_F is then divided into two subsets: $\mathcal{Y}_{F,B} = \{y_u^{T_u} \mid u \in \mathcal{V}, T_u \leq T_B\}$ consisting of labels for nodes whose final timestamps are before T_B , and $\mathcal{Y}_{F,A} = \{y_u^{T_u} \mid u \in \mathcal{V}, T_u > T_B\}$ containing labels for nodes whose final timestamps is after T_B . Similarly, \mathcal{Y}_E is partitioned into $\mathcal{Y}_{E,B} = \{y_u^t \mid u \in \mathcal{V}, t \in \mathcal{T}_u \setminus \{T_u\}, t \leq T_B\}$ and $\mathcal{Y}_{E,A} = \{y_u^t \mid u \in \mathcal{V}, t \in \mathcal{T}_u \setminus \{T_u\}, t > T_B\}$, representing labels whose corresponding timestamps are before and after T_B , respectively.

The dynamic graph backbone generates node embeddings \mathbf{h}_u^t for each node u at each timestamp $t \in \mathcal{T}_u$. The backbone takes the node features as input $\mathbf{n}_u \in \mathbb{R}^{d_N}$ and edge features $\mathbf{e}_{u,v}^t \in \mathbb{R}^{d_E}$. If the graph is non-attributed, we assume $\mathbf{n}_u = \mathbf{0}$ and $\mathbf{e}_{u,v}^t = \mathbf{0}$ for all nodes and edges, respectively.

Given a dynamic graph \mathcal{G} with dynamic labels and $|\mathcal{Y}_E| \gg |\mathcal{Y}_F|$, our task **label-limited dynamic node classification** aims to maximize $\log p(\mathcal{Y}_{F,B} | \mathcal{G})$. Specifically, our goal is to learn a model that can finally accurately predict \mathcal{Y}_F .

3 METHODOLOGY

In this section, we present PTCL for label-limited dynamic label learning. As shown in Figure 2, PTCL combines the variational EM framework with a Temporal Curriculum Learning strategy to effectively leverage both final timestamp labels and pseudo-labels in a dynamic graph setting.

3.1 VARIATIONAL EM FRAMEWORK

Following previous work (Zhao et al., 2022; Qu et al., 2019), we adopt the variational EM framework (Dempster et al., 1977; Neal & Hinton, 1998) to maximize $\log p(\mathcal{Y}_{F,B} | \mathcal{G})$.

3.1.1 EVIDENCE LOWER BOUND (ELBO)

To handle the unknown labels $\mathcal{Y}_{E,B}$, instead of directly optimizing $\log p_\theta(\mathcal{Y}_{F,B}|\mathcal{G})$, we maximize the evidence lower bound (ELBO) of the log-likelihood function:

$$\log p(\mathcal{Y}_{F,B}|\mathcal{G}) \geq \mathbb{E}_{q_\phi(\mathcal{Y}_{E,B}|\mathcal{G})}[\log p_\theta(\mathcal{Y}_{F,B}, \mathcal{Y}_{E,B}|\mathcal{G}) - \log q_\phi(\mathcal{Y}_{E,B}|\mathcal{G})], \quad (1)$$

where $p_\theta(\mathcal{Y}_{F,B}, \mathcal{Y}_{E,B}|\mathcal{G})$ is the joint distribution of observed and unknown labels, modeled by the dynamic graph backbone with parameters θ . $q_\phi(\mathcal{Y}_{E,B}|\mathcal{G})$ is the variational distribution approximating the true posterior distribution $p_\theta(\mathcal{Y}_{E,B}|\mathcal{Y}_{F,B}, \mathcal{G})$, modeled by the decoder with parameters ϕ .

To facilitate optimization, we follow mean field assumption (Getoor et al., 2001), which yields the following factorization:

$$q_\phi(\mathcal{Y}_{E,B}|\mathcal{G}) = \prod_{u \in \mathcal{V}} \prod_{\substack{t \in \mathcal{T}_u \setminus \{T_u\} \\ t \leq T_B}} q_\phi(y_u^t | \mathbf{h}_u^t), \quad (2)$$

where $q_\phi(y_u^t | \mathbf{h}_u^t)$ is the label distribution predicted by the decoder.

The ELBO can be optimized by alternating between the E-step and the M-step.

3.1.2 E-STEP

In the E-step, we use the wake-sleep algorithm (Hinton et al., 1995), following (Zhao et al., 2022). We fix the dynamic graph backbone θ and optimize the decoder ϕ to minimize the KL divergence between the true posterior distribution $p_\theta(\mathcal{Y}_{E,B}|\mathcal{G}, \mathcal{Y}_{F,B})$ and the variational distribution $q_\phi(\mathcal{Y}_{E,B}|\mathcal{G})$. The objective function for the decoder is:

$$\hat{\mathcal{O}}_\phi = \sum_{u \in \mathcal{V}} \sum_{\substack{t \in \mathcal{T}_u \setminus \{T_u\} \\ t \leq T_B}} \mathbb{E}_{p_\theta(y_u^t | \mathcal{G}, \mathcal{Y}_{F,B})} [\log q_\phi(y_u^t | \mathbf{h}_u^t)], \quad (3)$$

Following (Zhao et al., 2022), we use the pseudo-labels $\hat{\mathcal{Y}}_{E,B}$ generated by the decoder to approximate the distribution $p_\theta(y_u^t | \mathcal{G}, \mathcal{Y}_{F,B})$:

$$p_\theta(y_u^t | \mathcal{G}, \mathcal{Y}_{F,B}) \approx p_\theta(y_u^t | \mathcal{G}, \mathcal{Y}_{F,B}, \hat{\mathcal{Y}}_{E,B} \setminus \{\hat{y}_u^t\}), \quad (4)$$

then the objective function of the decoder changes to:

$$\hat{\mathcal{O}}_\phi = \alpha \sum_{u \in \mathcal{V}} \sum_{\substack{t \in \mathcal{T}_u \setminus \{T_u\} \\ t \leq T_B}} \mathbb{E}_{p_\theta(y_u^t | \mathcal{G}, \mathcal{Y}_{F,B}, \hat{\mathcal{Y}}_{E,B} \setminus \{\hat{y}_u^t\})} [\log q_\phi(y_u^t | \mathbf{h}_u^t)] + (1 - \alpha) \sum_{\substack{u \in \mathcal{V} \\ T_u \leq T_B}} \log q_\phi(y_u^{T_u} | \mathbf{h}_u^{T_u}), \quad (5)$$

where α is a hyperparameter that balances the weight of pseudo-labels and final timestamp labels.

But in practice, as shown in Section 4.2, we find that setting α to 0 yields the best performance, which means we train the decoder only with final timestamp labels. Therefore, the final objective function for the decoder is:

$$\mathcal{O}_\phi = \sum_{\substack{u \in \mathcal{V} \\ T_u \leq T_B}} \log q_\phi(y_u^{T_u} | \mathbf{h}_u^{T_u}), \quad (6)$$

3.1.3 M-STEP

In the M-step, following the previous work (Zhao et al., 2022; Qu et al., 2019), we aim to maximize the following pseudo-likelihood (Besag, 1975). Specifically, we fix the decoder ϕ and optimize the dynamic graph backbone θ using both the final timestamp labels $\mathcal{Y}_{F,B}$ and the pseudo-labels $\hat{\mathcal{Y}}_{E,B}$ generated in the E-step. The objective is to maximize the pseudo-likelihood:

$$\begin{aligned} \hat{\mathcal{O}}_\theta = & \beta \sum_{u \in \mathcal{V}} \sum_{\substack{t \in \mathcal{T}_u \setminus \{T_u\} \\ t \leq T_B}} \log p_\theta(y_u^t | \mathcal{G}, \mathcal{Y}_{F,B}, \hat{\mathcal{Y}}_{E,B} \setminus \{\hat{y}_u^t\}) \\ & + (1 - \beta) \sum_{\substack{u \in \mathcal{V} \\ T_u \leq T_B}} \log p_\theta(y_u^{T_u} | \mathcal{G}, \mathcal{Y}_{F,B} \setminus \{y_u^{T_u}\}, \hat{\mathcal{Y}}_{E,B}), \end{aligned} \quad (7)$$

where β is a hyperparameter that balance the weight of pseudo-labels and final timestamp labels. Since the backbone can only generate embeddings, we cascade the backbone and the fixed decoder, using final timestamp labels and pseudo-labels generated by the decoder to train the backbone.

3.2 TEMPORAL CURRICULUM LEARNING

In the M-step, the backbone is trained on both final labels $\mathcal{Y}_{F,B}$ and pseudo-labels $\hat{\mathcal{Y}}_{E,B}$. To mitigate noise from unreliable pseudo-labels at earlier timestamps, we introduce a **Temporal Curriculum Learning** strategy that dynamically adjusts pseudo-label weights based on their temporal proximity to the final timestamp and the EM iteration τ .

Specifically, we design a weighting mechanism for pseudo-labels based on their temporal order relative to the final timestamp T_u . Specifically, for each node $u \in \mathcal{V}$, timestamp $t \in \mathcal{T}_u$ in τ -th iteration, we define a weight $w_u^{t,\tau}$ as follows:

$$w_u^{t,\tau} = f_{\text{TW}}(t, u, \tau, T_u, \gamma) = \begin{cases} 1, & \text{if } d_u^t \leq \tau, \\ \exp(-\gamma \cdot (d_u^t - \tau)), & \text{if } d_u^t > \tau, \end{cases} \quad (8)$$

$$d_u^t = |\{t' \in \mathcal{T}_u \mid t' > t\}|, \quad (9)$$

where d_u^t is the discrete temporal distance between of timestamp t and T_u in \mathcal{T}_u . $\gamma > 0$ is a hyperparameter that controls the rate of Temporal Curriculum Learning decay. $w_u^{t,\tau}$ dynamically adjusts the importance of pseudo-labels during training. If $d_u^t \leq \tau$, the timestamp t is considered close to the final timestamp T_u , and the pseudo-label is assigned a weight of 1, indicating high confidence in its quality. And if $d_u^t > \tau$, the timestamp t is considered far from T_u , and the pseudo-label weight decays exponentially with the distance $\tau - d_u^t$, reducing its influence on the training process.

With the pseudo-label temporal weights $w_u^{t,\tau}$, the objective function for the M-step is modified as follows:

$$\begin{aligned} \mathcal{O}_\theta = & \beta \sum_{u \in \mathcal{V}} \sum_{\substack{t \in \mathcal{T}_u \setminus \{T_u\} \\ t \leq T_B}} w_u^{t,\tau} \log p_\theta(y_u^t | \mathcal{G}, \mathcal{Y}_{F,B}, \hat{\mathcal{Y}}_{E,B} \setminus \{\hat{y}_u^t\}) \\ & + (1 - \beta) \sum_{\substack{u \in \mathcal{V} \\ T_u \leq T_B}} \log p_\theta(y_u^{T_u} | \mathcal{G}, \mathcal{Y}_{F,B} \setminus \{y_u^{T_u}\}, \hat{\mathcal{Y}}_{E,B}). \end{aligned} \quad (10)$$

By incorporating $w_u^{t,\tau}$, the backbone is trained to prioritize high-quality pseudo-labels.

3.3 LEARNING AND OPTIMIZATION

Since the initial parameters of the EM algorithm are crucial for its performance (Dy & Brodley, 2004; Kwedlo, 2015), we first warm up the backbone by training it on a link prediction task. Then we proceed with the variational EM algorithm, which alternates between the E-step and the M-step. Finally, we use the decoder to predict $\mathcal{Y}_{F,A}$. The complete algorithm is summarized in Appendix.C.

4 EXPERIMENTS

Our experiments are designed to address the following key research questions (RQs):

RQ1: How does PTCL perform compared to other baselines when evaluated on the final timestamp labels? **RQ2:** Does pseudo-labels generated by PTCL improve performance by capturing the dynamic information of nodes? **RQ3:** Does the Temporal Curriculum Learning strategy improve performance? **RQ4:** Is PTCL stable and computationally efficient? More experiments can be found in Appendix.F.

4.1 EXPERIMENT SETTINGS

4.1.1 DATASETS

We evaluate PTCL on four datasets:

270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323

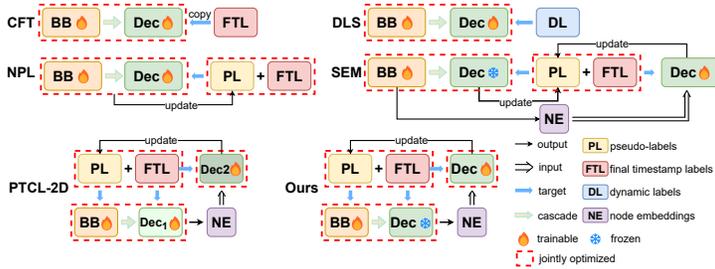


Figure 3: Architectures of baselines and PTCL. ‘BB’: backbone, ‘Dec’: decoder.

- **Wikipedia** (Kumar et al., 2019) and **Reddit** (Kumar et al., 2019): Bipartite interaction graphs capturing user activities over one month. These datasets feature binary classification tasks predicting banned users due to policy violations (Wikipedia) or community guideline violations (Reddit). Labels evolve dynamically as user behavior changes over time.
- **Dsub**: A subgraph of Dgraph (Huang et al., 2022) representing a financial transaction network. The binary classification task focuses on fraud detection—identifying users who ultimately fail to repay loans. Many nodes in this dataset represent background users that lack sufficient information for labeling but are retained to maintain graph connectivity.
- **CoOAG**: Our novel academic co-authorship graph derived from OAG (Open Academic Graph (Sinha et al.; Zhang et al., b;a; Tang et al.)). Nodes represent authors and edges denote co-authorship on AI conference papers. Node labels reflect evolving research interests across five fields: Computer Vision (CV), Natural Language Processing (NLP), Robotics (ROB), Data Mining/Web Search (DM/WS), and other AI/ML fields. Crucially, 43.2% of nodes change labels at least once, making it particularly valuable for studying label dynamics. Construction procedure of CoOAG is detailed in Appendix D.2.2, while comprehensive details for all datasets are provided in Appendix D.

For the binary classification tasks (Wikipedia, Reddit, Dsub), we report AUC (excluding background nodes in Dsub); for the multi-class CoOAG, we use ACC. To reflect real-world scenarios where only final labels are available, we adopt a timestamp-based split and evaluate solely on final labels ($\mathcal{Y}_{F,A}$). Nodes are split into train/val/test sets at a 7:1.5:1.5 ratio based on label distributions. All results are averaged over five random seeds. Implementation and hyperparameters are in Appendix J and K.

4.1.2 BASELINES

We compare PTCL with several different methods that can be adapted to our task. And specifically, we use five different dynamic backbones as backbones: TGAT (Xu et al., 2020), GraphMixer (Cong et al., 2023), TCL (Wang et al., 2021a), TGN (Rossi et al., 2020), and DyGFormer (Yu et al., 2023), and apply a simple 3-layer MLP as the decoder. More backbone details are in Appendix.E. As shown in Figure 3, the baselines are designed to cover a range of approaches:

- **CFT (Copy-Final Timestamp labels)**: A naive baseline that simply copies the final timestamp labels ($\mathcal{Y}_{F,B}$) to earlier timestamps as approximations of dynamic labels ($\mathcal{Y}_{E,B}$) for training.
- **DLS (Dynamic Label Supervision)**: A baseline that performs supervised training directly using the dynamic labels provided by the dataset (e.g., Wikipedia, Reddit), where available (Yu et al., 2023; Rossi et al., 2020).
- **NPL (Naive Pseudo-Labels)**: A variant of PTCL that uses pseudo-labels but jointly optimizes the backbone and decoder without EM optimization.
- **PTCL-2D (PTCL with 2 Decoders)**: A variant of PTCL that uses two decoders: one decoder is trained exclusively on the final timestamp labels (E-step), generating pseudo-labels, while the other decoder is jointly optimized with the backbone on weighted pseudo-labels and final timestamp labels (M-step). The final embeddings are provided by the backbone for the E-step training.
- **SEM (Standard EM)**: A variant of PTCL where both the E-step and M-step are trained on the weighted loss of pseudo-labels and final timestamp labels (Zhao et al., 2022), while other components remain the same as PTCL. In this way, E-step uses Eq. (5) as the objective function instead of Eq. (6).

Table 1: Performance comparison across datasets (Wikipedia, Reddit, Dsub, CoOAG). We run all experiments with five random seeds to ensure a consistent evaluation and report the average performance as well as standard deviation in parentheses. **Bold** indicates the best performance, underline the second best. Dsub and CoOAG datasets can not apply the DLS method due to a lack of dynamic labels. TGN runs out of memory on Dsub due to its high space cost.

Backbone	Method	Wikipedia	Reddit	Dsub	CoOAG
		AUC	AUC	AUC	ACC
TGAT	CFT	77.43 (\pm 3.01)	82.68 (\pm 0.06)	62.32 (\pm 1.27)	86.28 (\pm 0.18)
	DLS	79.56 (\pm 2.55)	78.66 (\pm 0.04)	-	-
	NPL	78.52 (\pm 2.28)	80.72 (\pm 2.65)	61.71 (\pm 2.21)	87.67 (\pm 0.61)
	PTCL-2D	78.20 (\pm 6.83)	85.84 (\pm 6.09)	60.76 (\pm 2.91)	88.37 (\pm 0.16)
	SEM	81.09 (\pm 3.62)	86.27 (\pm 5.99)	64.34 (\pm 0.99)	88.38 (\pm 0.38)
	Ours	85.52 (\pm 3.29)	87.31 (\pm 6.50)	65.07 (\pm 1.57)	89.05 (\pm 0.63)
TCL	CFT	76.27 (\pm 4.68)	84.48 (\pm 5.53)	62.60 (\pm 1.26)	86.12 (\pm 1.11)
	DLS	80.55 (\pm 1.93)	82.85 (\pm 0.38)	-	-
	NPL	77.71 (\pm 5.66)	84.20 (\pm 3.86)	63.59 (\pm 3.09)	87.59 (\pm 0.26)
	PTCL-2D	76.68 (\pm 2.86)	86.98 (\pm 3.34)	60.64 (\pm 1.41)	<u>87.94</u> (\pm 0.30)
	SEM	81.02 (\pm 2.82)	87.56 (\pm 1.56)	<u>65.11</u> (\pm 1.26)	87.90 (\pm 0.18)
	Ours	82.27 (\pm 4.62)	89.41 (\pm 3.32)	66.80 (\pm 2.45)	88.24 (\pm 0.26)
TGN	CFT	80.68 (\pm 2.02)	89.69 (\pm 2.07)	OOM	83.65 (\pm 0.63)
	DLS	78.48 (\pm 1.60)	80.92 (\pm 4.99)	-	-
	NPL	<u>87.58</u> (\pm 2.14)	84.22 (\pm 2.48)	OOM	86.13 (\pm 0.31)
	PTCL-2D	86.59 (\pm 3.01)	<u>86.76</u> (\pm 3.89)	OOM	<u>86.23</u> (\pm 0.66)
	SEM	86.34 (\pm 2.66)	82.61 (\pm 3.07)	OOM	86.07 (\pm 0.66)
	Ours	87.97 (\pm 2.90)	84.32 (\pm 2.07)	OOM	86.71 (\pm 0.66)
GraphMixer	CFT	76.60 (\pm 2.00)	66.11 (\pm 6.04)	62.78 (\pm 1.90)	85.63 (\pm 0.14)
	DLS	80.70 (\pm 4.00)	61.97 (\pm 7.36)	-	-
	NPL	80.86 (\pm 1.62)	<u>71.72</u> (\pm 6.48)	67.14 (\pm 1.68)	86.98 (\pm 0.61)
	PTCL-2D	81.41 (\pm 4.25)	66.86 (\pm 11.14)	62.33 (\pm 1.35)	87.51 (\pm 0.51)
	SEM	<u>83.33</u> (\pm 1.45)	68.65 (\pm 3.70)	<u>69.23</u> (\pm 1.92)	<u>88.07</u> (\pm 0.30)
	Ours	84.09 (\pm 0.95)	71.93 (\pm 7.94)	69.76 (\pm 1.54)	88.26 (\pm 0.38)
DyGFormer	CFT	64.76 (\pm 9.21)	67.14 (\pm 8.04)	68.48 (\pm 1.46)	85.27 (\pm 0.83)
	DLS	71.95 (\pm 2.29)	64.63 (\pm 4.90)	-	-
	NPL	<u>73.85</u> (\pm 5.44)	67.44 (\pm 3.47)	<u>70.31</u> (\pm 1.11)	<u>86.16</u> (\pm 0.38)
	PTCL-2D	66.48 (\pm 6.76)	71.14 (\pm 6.27)	69.11 (\pm 2.96)	86.04 (\pm 0.30)
	SEM	70.91 (\pm 8.80)	<u>71.59</u> (\pm 4.51)	69.75 (\pm 2.47)	86.07 (\pm 0.66)
	Ours	74.85 (\pm 3.07)	75.86 (\pm 8.04)	72.39 (\pm 1.91)	86.26 (\pm 0.27)

4.1.3 FLiD: A NOVEL FRAMEWORK

We introduce **FLiD**, a new code framework tailored for scenarios with only final timestamp labels. Existing frameworks such as DyGLib (Yu et al., 2023) and TGL (Zhou et al., 2022) have made important progress in dynamic graph learning: DyGLib unifies models for fully supervised link prediction and node classification, while TGL focuses on scalable training for large dynamic graphs. However, neither framework directly addresses the label-limited dynamic node classification setting considered in this work.

All experiments in this paper are conducted in FLiD, featuring:

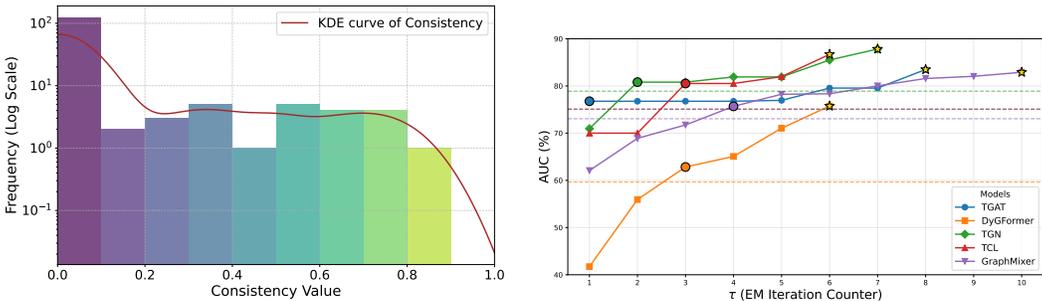
- Support for multiple training paradigms, including CFT, DLS, NPL, SEM, PTCL, PTCL-2D;
- Pseudo-labeling enhancements such as Confidence Score Threshold (CST), Entropy of Softmax Trajectory (EST), and Temporal Curriculum Learning;
- A variety of dynamic graph backbones, including TGAT, TGN, GraphMixer, TCL, and DyGFormer;
- A custom data preprocessing and splitting strategy designed for the label-limited setting.

4.2 RQ1: MAIN RESULTS

To evaluate the effectiveness of PTCL, we conduct experiments using five different backbones and compare against several baselines. As shown in Table 1, PTCL consistently improves performance across all datasets and backbones.

Table 2: AUC comparison of different backbones using Dynamic Label Supervised Learning (DLS) and Pseudo-Label Supervised Learning (PLS) on Wikipedia Dataset.

	TGAT	TCL	TGN	GraphMixer	DyGFormer
DLS	79.56 (± 2.55)	80.55 (± 1.93)	78.48 (± 1.60)	80.70 (± 4.00)	71.95 (± 2.29)
PLS	81.11 (± 5.12)	82.19 (± 0.64)	79.33 (± 2.58)	81.02 (± 2.94)	72.62 (± 2.07)



(a) Histogram of pseudo-labels consistency. (b) Convergence curves for 5 backbones. Star markers (*) denote peak performance; circled points (●) indicate surpassing baselines. Dashed lines show baseline AUC.

Figure 4: (a) Pseudo-label consistency; (b) Convergence of backbones.

- Effectiveness of Pseudo-Labels.** PTCL significantly outperforms both CFT and DLS (up to +11.23% in AUC/ACC), showing that learned pseudo-labels better capture node dynamics than copied or even original dynamic labels (details in Section 4.3).
- Importance of Our Optimization Strategy.** Compared to NPL, PTCL achieves +2.74% improvement on average. Additionally, PTCL outperforms SEM, demonstrating that training the decoder exclusively on final timestamp labels ($\alpha = 0$, as described in Section 3.1.2) leads to better results. Although different with the original variation EM framework, we regard it as a trade-off between strict mathematical adherence to the standard variational EM M-step and engineering effectiveness in dealing with pseudo-labels. These results highlight the benefit of optimizing the decoder solely with final labels, which ensures alignment with ground truth and prevents error propagation from noisy pseudo-labels.
- Efficiency over PTCL-2D.** PTCL achieves better performance with lower compute cost than PTCL-2D variant, which indicates that a single decoder shared by E-step and M-step promotes more stable and consistent training.

4.3 RQ2: PSEUDO-LABEL ANALYSIS

In this section, we evaluate the effectiveness of our pseudo-labels and their ability to capture dynamic patterns through two experiments.

4.3.1 PSEUDO-LABEL SUPERVISION STUDY

To analyze the effectiveness of our pseudo-labels, we conduct the following experiment: We train models from scratch using pseudo-labels generated by our trained model as full supervision labels and compare the results with DLS. As shown in Table 2, models trained with our pseudo-labels consistently outperform those trained with original dynamic labels with an average improvement of 0.96% in AUC.

4.3.2 LABEL CONSISTENCY ANALYSIS

To further investigate the temporal changes of labels and to explain the counter-intuitive result where the Dynamic Label Supervision (DLS) baseline underperforms simple pseudo-labeling methods (Table 1), we analyze the labels’ consistency on Wikipedia’s positive samples. Consistency is

Table 3: AUC comparison of our Temporal Curriculum Learning with other strategies. CST = Confidence Score Threshold, EST = Entropy of Softmax Trajectory.

Backbone	Dataset	Naive	CST	EST	Ours
TGAT	Wikipedia	<u>82.38</u> (± 4.16)	79.48 (± 2.84)	81.89 (± 4.83)	85.52 (± 3.29)
	Dsub	<u>64.12</u> (± 1.87)	62.78 (± 2.27)	63.80 (± 1.45)	65.07 (± 1.57)
TCL	Wikipedia	<u>81.77</u> (± 1.45)	78.06 (± 2.59)	80.56 (± 5.75)	82.27 (± 4.62)
	Dsub	<u>65.29</u> (± 1.56)	63.68 (± 1.36)	64.57 (± 1.53)	66.80 (± 2.45)
TGN	Wikipedia	86.33 (± 3.51)	83.63 (± 1.33)	<u>86.88</u> (± 1.52)	87.97 (± 2.90)
	Dsub	OOM	OOM	OOM	OOM
GraphMixer	Wikipedia	<u>83.34</u> (± 2.22)	81.59 (± 2.54)	81.13 (± 1.44)	84.09 (± 0.95)
	Dsub	68.08 (± 1.65)	<u>68.79</u> (± 1.37)	67.55 (± 3.05)	69.76 (± 1.54)
DyGFormer	Wikipedia	<u>69.42</u> (± 7.60)	67.15 (± 4.81)	68.39 (± 8.78)	74.85 (± 3.07)
	Dsub	<u>72.30</u> (± 2.02)	70.86 (± 2.63)	69.21 (± 2.08)	72.39 (± 1.91)

quantified as follows:

$$\hat{N}_{u'} = \max \left\{ k \in \mathbb{N}^+ \mid y_{u'}^{t_i} = y_{u'}^{T_{u'}}, \forall i \in \{|\mathcal{T}_{u'}| - k, \dots, |\mathcal{T}_{u'}| - 1\} \right\}, \quad (11)$$

$$C_{u'} = \frac{\hat{N}_{u'}}{|\mathcal{T}_{u'}| - 1}. \quad (12)$$

where $u' \in \mathcal{V}_{\text{neg}}$, $\mathcal{V}_{\text{neg}} = \{u' \mid u' \in \mathcal{V}, y_{u'}^{T_{u'}} = 1\}$.

Our analysis reveals that the low performance of the DLS baseline stems from the extreme inconsistency of the original dynamic labels. In Wikipedia, dynamic labels for negative samples change abruptly, leading to a consistency measure of $C_{u'} \equiv 0$ for many samples. This sudden shift represents potentially noisy or instantaneous state changes that are detrimental to the model’s temporal learning process. Conversely, simple pseudo-labeling like CFT enforces overly rigid continuity ($C_{u'} \equiv 1$), which causes feature misalignment by treating all historical steps as the final state.

In contrast, PTCL generates pseudo-labels with optimized and varying temporal consistency (Figure 4a, typically $0 < C_{u'} < 1$). This targeted intermediate consistency is crucial: it facilitates smooth and meaningful temporal transitions in node representations and creates a stabilized supervision signal that is qualitatively superior to the original, noisy dynamic labels. The necessity of our variational EM strategy, therefore, lies not in mere label recovery, but in the generation of a cleaner, temporally adaptive supervision signal for robust dynamic learning.

4.4 RQ3: TEMPORAL CURRICULUM LEARNING ANALYSIS

To comprehensively evaluate our Temporal Curriculum Learning design, we conduct a comparison experiment against the naive solution which uses all the generated pseudo-labels, and two commonly used baseline strategies to choose more reliable pseudo-labels in Curriculum Learning, as introduced in Section G.1:

- *Confidence Score Threshold (CST)* (Sun et al., 2019; He et al., 2022; Cascante-Bonilla et al., 2020): This method filters pseudo-labels based on their confidence scores, improving the overall quality of the labels.
- *Entropy of Softmax Trajectory (EST)* (Song et al., 2019; Pei et al., 2024): This method filters pseudo-labels using the entropy of the softmax trajectory, which is an accumulated distribution that summarizes the model’s disagreement across training rounds.

As shown in Table 3, our Temporal Curriculum Learning consistently achieves the best AUC across all backbones and datasets, confirming its effectiveness. The Naive strategy underperforms PTCL by an average of 1.74 %, reflecting the noise in unfiltered pseudo-labels. CST and EST perform even worse, as their static filters fail to capture the temporal reliability of pseudo-labels. These results highlight the advantage of leveraging temporal dynamics for curriculum learning.

486 4.5 RQ4: CONVERGENCE AND EFFICIENCY ANALYSIS

487
488 We assess the convergence and efficiency of `PTCL` on Wikipedia using 5 backbones, with CFT as
489 the baseline. As shown in Figure 4b, `PTCL` converges rapidly across all models: TGAT surpasses
490 its baseline at the first iteration, others within 2–4. All reach peak AUC in 6–10 iterations, with
491 DyGFormer showing the largest gain (+16.1%). Each EM iteration adds only $0.8\times-1.2\times$ training
492 time, demonstrating both fast convergence and practical overhead.

493 494 5 RELATED WORK

495 496 5.1 DYNAMIC NODE CLASSIFICATION

497
498 Dynamic graph learning spans discrete-time (snapshot-based) and continuous-time paradigms. A
499 large body of work develops powerful dynamic graph encoders that process temporal interactions
500 for downstream prediction tasks. Representative continuous-time models include Temporal Graph
501 Networks (TGN) (Rossi et al., 2020), which maintain learnable memory states to aggregate historical
502 information, and TGAT (Xu et al., 2020), which introduces functional time encodings and temporal
503 self-attention. Other approaches, such as DyRep (Trivedi et al., 2019), DBGNN (Qarkaxhija et al.,
504 2022), and TGBase (Poursafaei et al.), further explore temporal point processes, higher-order temporal
505 message passing, or feature-based temporal representations to model evolving node states.

506 Beyond representation learning, some works perform dynamic node classification under various
507 supervision settings. Early models such as JODIE (Kumar et al., 2019) and DynPPE (Guo et al., 2021)
508 use temporal embeddings to classify user or node states. More recent methods—e.g., TADGNN
509 (Sun et al., 2022) and OTGNet (Feng et al., 2023)—explicitly propagate information over time-
510 augmented structures or disentangle temporal class-dependent signals. HYDG (Ma et al., 2024)
511 further incorporates hypergraph structures to capture high-order temporal dependencies among
512 evolving node–hyperedge relations. DyGPrompt (Yu et al., 2025) introduces a dual-prompt framework
513 that aligns temporal pre-training objectives with downstream dynamic classification tasks via evolving
514 node–time patterns. PRES (Su et al., 2024) further improves the scalability of memory-based dynamic
515 GNNs via a predict-to-smooth framework that supports substantially larger temporal batches. Notably,
516 several studies do not require full supervision at all timestamps. For example, SAD (Tian et al.,
517 2023)—a semi-supervised anomaly detection framework on dynamic graphs—leverages temporal
518 GNNs with pseudo-label contrastive learning to exploit limited labeled nodes, demonstrating that
519 temporal supervision can be significantly relaxed.

520 The above literature provides strong evidence that dynamic node prediction has been studied across
521 representation learning, timestamp-wise classification, and semi-supervised settings. However,
522 most existing approaches either assume labels distributed across multiple timestamps or rely on
523 abundant anomaly labels. In contrast, our work focuses on the under-explored setting where only
524 final-timestamp labels are available, a setting that naturally arises in real-world applications with high
525 annotation costs and label uncertainty. Building on continuous-time graph encoders, we introduce
526 a curriculum-guided pseudo-labeling mechanism tailored to label-scarce dynamic classification.
527 Additional related work is provided in Appendix G.

528 529 530 6 CONCLUSION

531
532 In this work, we address dynamic node classification under limited labels by proposing `PTCL`, an
533 extensible method based on temporally-weighted pseudo-labels and a variational EM framework.
534 `PTCL` achieves up to 11.23% AUC/ACC gain across diverse real-world scenarios, validating the
535 effectiveness of modeling temporal dynamics and our proposed Temporal Curriculum Learning. We
536 also introduce the CoOAG dataset and FLiD framework to support practical evaluation. Beyond
537 classification, `PTCL` is adaptable to other dynamic graph tasks, offering a solid foundation for future
538 research on learning node evolution under realistic supervision constraints.

540 ETHICS STATEMENT

541

542

543

544

545

546

547

548

549

550

551

552

553

554

All authors of this paper have read and agree to adhere to the ICLR Code of Ethics. Our work involves the construction and release of a new academic collaboration dataset (CoOAG), derived from publicly available data in the Open Academic Graph (OAG). All author labels in CoOAG are inferred using a large language model (Qwen-Plus (Yang et al., 2024)) based on anonymized publication metadata (e.g., Fields of Study and abstracts), without accessing personally identifiable information. The labeling process was validated on a manually annotated subset (120 samples) and does not involve human subjects in experimental settings, thus no Institutional Review Board (IRB) approval was required. We do not foresee harmful applications of our method; on the contrary, PTCL is designed to reduce annotation costs in dynamic graph settings such as fraud detection or academic trend analysis, potentially benefiting resource-constrained institutions. The proposed framework FLiD is open-sourced to promote transparency and equitable access. No conflicts of interest exist among the authors, and this research received no external sponsorship.

555 REPRODUCIBILITY STATEMENT

556

557

558

559

560

561

562

563

564

565

We have taken extensive measures to ensure the reproducibility of our results. The complete implementation of our method PTCL, the FLiD framework, data preprocessing pipelines, and all experimental configurations are included in the supplementary materials. Hyperparameters for each backbone, training protocols, optimizer settings, and early stopping criteria are detailed in Appendix K (Table 10), and model architectures are described in Appendix E. The CoOAG dataset construction procedure, including prompt templates, feature extraction, and temporal splitting, is fully documented in Appendix D.2.2. All experiments were repeated over five random seeds, with mean and standard deviation reported in Table 1. Code and data will be made publicly available upon publication.

566 REFERENCES

567

568

569

570

571

572

573

574

575

576

577

578

579

580

581

582

583

584

585

586

587

588

589

590

591

592

593

- Qwen2 technical report. *arXiv preprint arXiv:2407.10671*, 2024.
- Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *International Conference on Machine Learning*, 2009. URL <https://api.semanticscholar.org/CorpusID:873046>.
- Julian Besag. Statistical analysis of non-lattice data. *The Statistician*, 24:179–195, 1975. URL <https://api.semanticscholar.org/CorpusID:116757950>.
- Smriti Bhagat, Graham Cormode, and S Muthukrishnan. Node classification in social networks. *Social network data analytics*, pp. 115–148, 2011.
- Michael M Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42, 2017.
- Paola Cascante-Bonilla, Fuwen Tan, Yanjun Qi, and Vicente Ordonez. Curriculum labeling: Revisiting pseudo-labeling for semi-supervised learning. In *AAAI Conference on Artificial Intelligence*, 2020. URL <https://api.semanticscholar.org/CorpusID:228096598>.
- Olivier Chapelle and Alexander Zien. Semi-supervised classification by low density separation. In *International Conference on Artificial Intelligence and Statistics*, 2005. URL <https://api.semanticscholar.org/CorpusID:14283441>.
- Olivier Chapelle, Bernhard Schölkopf, and Alexander Zien. Introduction to semi-supervised learning. 2006.
- Weilin Cong, Si Zhang, Jian Kang, Baichuan Yuan, Hao Wu, Xin Zhou, Hanghang Tong, and Mehrdad Mahdavi. Do we really need complicated model architectures for temporal networks? *arXiv preprint arXiv:2302.11636*, 2023.

- 594 Arthur P. Dempster, Nan M. Laird, and Donald B. Rubin. Maximum likelihood from incom-
595 plete data via the em - algorithm plus discussions on the paper. 1977. URL <https://api.semanticscholar.org/CorpusID:4193919>.
596
597
- 598 Jennifer G Dy and Carla E Brodley. Feature selection for unsupervised learning. *Journal of machine*
599 *learning research*, 5(Aug):845–889, 2004.
- 600 Kaituo Feng, Changsheng Li, Xiaolu Zhang, and Jun Zhou. Towards open temporal graph neural
601 networks. *arXiv preprint arXiv:2303.15015*, 2023.
602
- 603 Shangbin Feng, Zhaoxuan Tan, Herun Wan, Ningnan Wang, Zilong Chen, Binchi Zhang, Qinghua
604 Zheng, Wenqian Zhang, Zhenyu Lei, Shujie Yang, et al. Twibot-22: Towards graph-based twitter
605 bot detection. *Advances in Neural Information Processing Systems*, 35:35254–35269, 2022.
- 606 Matthias Fey and Jan E. Lenssen. Fast graph representation learning with PyTorch Geometric. In
607 *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019.
608
- 609 Lise Getoor, Nir Friedman, Daphne Koller, and Ben Taskar. Learning probabilistic models of
610 relational structure. In *International Conference on Machine Learning*, 2001. URL <https://api.semanticscholar.org/CorpusID:10551607>.
611
- 612 Yves Grandvalet and Yoshua Bengio. Semi-supervised learning by entropy minimization.
613 In *Conférence francophone sur l'apprentissage automatique*, 2004. URL <https://api.semanticscholar.org/CorpusID:7890982>.
614
- 615 Xingzhi Guo, Baojian Zhou, and Steven Skiena. Subset node representation learning over large
616 dynamic graphs. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery*
617 *& Data Mining*, pp. 516–526, 2021.
618
- 619 Haiyun He, Gholamali Aminian, Yuheng Bu, Miguel L. Rodrigues, and Vincent Y. F. Tan. How
620 does pseudo-labeling affect the generalization error of the semi-supervised gibbs algorithm? In
621 *International Conference on Artificial Intelligence and Statistics*, 2022. URL <https://api.semanticscholar.org/CorpusID:252918807>.
622
- 623 Geoffrey E. Hinton, Peter Dayan, Brendan J. Frey, and R M Neal. The "wake-sleep" algorithm
624 for unsupervised neural networks. *Science*, 268 5214:1158–61, 1995. URL <https://api.semanticscholar.org/CorpusID:871473>.
625
- 626 Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network.
627 *ArXiv*, abs/1503.02531, 2015. URL <https://api.semanticscholar.org/CorpusID:7200347>.
628
629
- 630 Petter Holme and Jari Saramäki. Temporal networks. *Physics reports*, 519(3):97–125, 2012.
631
- 632 Weihua Hu, Matthias Fey, Hongyu Ren, Maho Nakata, Yuxiao Dong, and Jure Leskovec. Ogb-lsc: A
633 large-scale challenge for machine learning on graphs. *arXiv preprint arXiv:2103.09430*, 2021.
- 634 Xuanwen Huang, Yang Yang, Yang Wang, Chunping Wang, Zhisheng Zhang, Jiarong Xu, Lei Chen,
635 and Michalis Vazirgiannis. Dgraph: A large-scale financial dataset for graph anomaly detection.
636 *Advances in Neural Information Processing Systems*, 35:22765–22777, 2022.
637
- 638 Tao Jia, Dashun Wang, and Boleslaw K Szymanski. Quantifying patterns of research-interest
639 evolution. *Nature Human Behaviour*, 1(4):0078, 2017.
- 640 Patrick Kage, Jay C Rothenberger, Pavlos Andreadis, and Dimitrios I Diochnos. A review of
641 pseudo-labeling for computer vision. *arXiv preprint arXiv:2408.07221*, 2024.
642
- 643 Barakeel Fansu Kamhoua, Lin Zhang, Yongqiang Chen, Tongliang Liu, Huamin Qu, and Bo Han.
644 Sparse labels node classification: Unsupervised learning for mentoring supervised learning in
645 sparse label settings.
- 646 Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization.
647 *CoRR*, abs/1412.6980, 2014. URL <https://api.semanticscholar.org/CorpusID:6628106>.

- 648 Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks.
649 *arXiv preprint arXiv:1609.02907*, 2016.
- 650
- 651 Srijan Kumar, Xikun Zhang, and Jure Leskovec. Predicting dynamic embedding trajectory in
652 temporal interaction networks. In *Proceedings of the 25th ACM SIGKDD international conference*
653 *on knowledge discovery & data mining*, pp. 1269–1278, 2019.
- 654
- 655 Wojciech Kwedlo. A new random approach for initialization of the multiple restart em algorithm for
656 gaussian model-based clustering. *Pattern Analysis and Applications*, 18:757–770, 2015.
- 657
- 658 Semi-Supervised Learning. Semi-supervised learning. *CSZ2006. html*, 5:2, 2006.
- 659
- 660 Dong-Hyun Lee et al. Pseudo-label: The simple and efficient semi-supervised learning method for
661 deep neural networks. In *Workshop on challenges in representation learning, ICML*, volume 3, pp.
662 896. Atlanta, 2013.
- 663
- 664 Jianfeng Li and Dexiang Yang. Research on financial fraud detection models integrating multiple
665 relational graphs. *Systems*, 11(11):539, 2023.
- 666
- 667 Michelle M Li, Kexin Huang, and Marinka Zitnik. Graph representation learning in biomedicine and
668 healthcare. *Nature Biomedical Engineering*, 6(12):1353–1369, 2022a.
- 669
- 670 Qimai Li, Zhichao Han, and Xiao-Ming Wu. Deeper insights into graph convolutional networks for
671 semi-supervised learning. In *AAAI Conference on Artificial Intelligence*, 2018a. URL <https://api.semanticscholar.org/CorpusID:11118105>.
- 672
- 673 Yayong Li, Jie Yin, and Ling Chen. Informative pseudo-labeling for graph neural networks with
674 few labels. *Data Mining and Knowledge Discovery*, 37:228–254, 2022b. URL <https://api.semanticscholar.org/CorpusID:246063495>.
- 675
- 676 Zhun Li, ByungSoo Ko, and Ho-Jin Choi. Naive semi-supervised deep learning using pseudo-
677 label. *Peer-to-Peer Networking and Applications*, 12:1358 – 1368, 2018b. URL <https://api.semanticscholar.org/CorpusID:69529604>.
- 678
- 679 Xiaoxu Ma, Chen Zhao, Minglai Shao, and Yujie Lin. Hypergraph-based dynamic graph node
680 classification, 2024. URL <https://arxiv.org/abs/2412.20321>.
- 681
- 682 Takeru Miyato, Shin ichi Maeda, Masanori Koyama, and Shin Ishii. Virtual adversarial train-
683 ing: A regularization method for supervised and semi-supervised learning. *IEEE Transac-*
684 *tions on Pattern Analysis and Machine Intelligence*, 41:1979–1993, 2017. URL <https://api.semanticscholar.org/CorpusID:17504174>.
- 685
- 686 Radford M. Neal and Geoffrey E. Hinton. A view of the em algorithm that justifies incremental,
687 sparse, and other variants. In *Learning in Graphical Models*, 1998. URL <https://api.semanticscholar.org/CorpusID:17947141>.
- 688
- 689 Mark EJ Newman, Duncan J Watts, and Steven H Strogatz. Random graph models of social networks.
690 *Proceedings of the national academy of sciences*, 99(suppl_1):2566–2572, 2002.
- 691
- 692 Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor
693 Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style,
694 high-performance deep learning library. *Advances in neural information processing systems*, 32,
695 2019.
- 696
- 697 Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier
698 Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn:
699 Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011.
- 700
- 701 Hongbin Pei, Yuheng Xiong, Pinghui Wang, Jing Tao, Jialun Liu, Huiqi Deng, Jie Ma, and Xiaohong
Guan. Memory disagreement: A pseudo-labeling measure from training dynamics for semi-
supervised graph learning. *Proceedings of the ACM on Web Conference 2024*, 2024. URL
<https://api.semanticscholar.org/CorpusID:269708945>.

- 702 Farimah Poursafaei, Zeljko Zilic, and Reihaneh Rabbany. *A Strong Node Classification Baseline for*
703 *Temporal Graphs*, pp. 648–656. doi: 10.1137/1.9781611977172.73. URL <https://epubs.siam.org/doi/abs/10.1137/1.9781611977172.73>.
704
705
- 706 Lisi Qarkaxhija, Vincenzo Perri, and Ingo Scholtes. De bruijn goes neural: Causality-aware graph
707 neural networks for time series data on dynamic graphs. In Bastian Rieck and Razvan Pascanu
708 (eds.), *Proceedings of the First Learning on Graphs Conference*, volume 198 of *Proceedings*
709 *of Machine Learning Research*, pp. 51:1–51:21. PMLR, 09–12 Dec 2022. URL <https://proceedings.mlr.press/v198/qarkaxhija22a.html>.
710
- 711 Meng Qu. *Towards combining deep learning and statistical relational learning for reasoning*
712 *on graphs*. Ph.d. thesis, Université de Montréal, Montreal, Canada, Jan 2024. URL <http://hdl.handle.net/1866/32584>.
713
- 714 Meng Qu, Yoshua Bengio, and Jian Tang. Gmnn: Graph markov neural networks.
715 *ArXiv*, abs/1905.06214, 2019. URL <https://api.semanticscholar.org/CorpusID:155093091>.
716
- 717 Yu Rong, Wenbing Huang, Tingyang Xu, and Junzhou Huang. Dropedge: Towards deep graph
718 convolutional networks on node classification. *arXiv preprint arXiv:1907.10903*, 2019.
719
- 720 Emanuele Rossi, Ben Chamberlain, Fabrizio Frasca, Davide Eynard, Federico Monti, and Michael
721 Bronstein. Temporal graph networks for deep learning on dynamic graphs. *arXiv preprint*
722 *arXiv:2006.10637*, 2020.
- 723 Oleksandr Shchur, Maximilian Mumme, Aleksandar Bojchevski, and Stephan Günnemann. Pitfalls
724 of Graph Neural Network Evaluation. URL <http://arxiv.org/abs/1811.05868>.
725
- 726 Arnab Sinha, Zhihong Shen, Yang Song, Hao Ma, Darrin Eide, Bo-June (Paul) Hsu, and Kuansan
727 Wang. An Overview of Microsoft Academic Service (MAS) and Applications. In *Proceedings*
728 *of the 24th International Conference on World Wide Web, WWW ’15 Companion*, pp. 243–246.
729 Association for Computing Machinery. ISBN 978-1-4503-3473-0. doi: 10.1145/2740908.2742839.
730 URL <https://doi.org/10.1145/2740908.2742839>.
731
- 732 Hwanjun Song, Minseok Kim, and Jae-Gil Lee. Selfie: Refurbishing unclean samples for robust
733 deep learning. In *International Conference on Machine Learning*, 2019. URL <https://api.semanticscholar.org/CorpusID:174800904>.
734
- 735 Petru Soviany, Radu Tudor Ionescu, Paolo Rota, and N. Sebe. Curriculum learning: A sur-
736 vey. *International Journal of Computer Vision*, 130:1526 – 1565, 2021. URL <https://api.semanticscholar.org/CorpusID:231709290>.
737
- 738 Junwei Su, Difan Zou, and Chuan Wu. Pres: Toward scalable memory-based dynamic graph neural
739 networks, 2024. URL <https://arxiv.org/abs/2402.04284>.
740
- 741 Jiarui Sun, Mengting Gu, Chin-Chia Michael Yeh, Yujie Fan, Girish Chowdhary, and Wei Zhang.
742 Dynamic graph node classification via time augmentation, 2022. URL <https://arxiv.org/abs/2212.03449>.
743
- 744 Ke Sun, Zhanxing Zhu, and Zhouchen Lin. Multi-stage self-supervised learning for graph convo-
745 lutional networks. *ArXiv*, abs/1902.11038, 2019. URL <https://api.semanticscholar.org/CorpusID:67855919>.
746
- 747 Jie Tang, Jing Zhang, Limin Yao, Juanzi Li, Li Zhang, and Zhong Su. ArnetMiner: Extraction and
748 mining of academic social networks. In *Proceedings of the 14th ACM SIGKDD International*
749 *Conference on Knowledge Discovery and Data Mining, KDD ’08*, pp. 990–998. Association for
750 Computing Machinery. doi: 10.1145/1401890.1402008. URL <https://doi.org/10.1145/1401890.1402008>.
751
- 752 Sheng Tian, Jihai Dong, Jintang Li, Wenlong Zhao, Xiaolong Xu, Baokun Wang, Bowen Song,
753 Changhua Meng, Tianyi Zhang, and Liang Chen. Sad: Semi-supervised anomaly detection on
754 dynamic graphs. In Edith Elkind (ed.), *Proceedings of the Thirty-Second International Joint*
755 *Conference on Artificial Intelligence, IJCAI-23*, pp. 2306–2314. International Joint Conferences
on Artificial Intelligence Organization, 8 2023. doi: 10.24963/ijcai.2023/256. URL <https://doi.org/10.24963/ijcai.2023/256>. Main Track.

- 756 Rakshit Trivedi, Mehrdad Farajtabar, Prasenjeet Biswal, and Hongyuan Zha. Dyrep: Learning
757 representations over dynamic graphs. In *International Conference on Learning Representations*
758 (*ICLR*), 2019.
- 759 Jesper E Van Engelen and Holger H Hoos. A survey on semi-supervised learning. *Machine learning*,
760 109(2):373–440, 2020.
- 761 Lu Wang, Xiaofu Chang, Shuang Li, Yunfei Chu, Hui Li, Wei Zhang, Xiaofeng He, Le Song, Jingren
762 Zhou, and Hongxia Yang. Tcl: Transformer-based dynamic graph modelling via contrastive
763 learning. *ArXiv*, abs/2105.07944, 2021a. URL [https://api.semanticscholar.org/
764 CorpusID:234741805](https://api.semanticscholar.org/CorpusID:234741805).
- 765 Xin Wang, Yudong Chen, and Wenwu Zhu. A survey on curriculum learning. *IEEE Transactions*
766 *on Pattern Analysis and Machine Intelligence*, 44:4555–4576, 2021b. URL [https://api.
767 semanticscholar.org/CorpusID:232362223](https://api.semanticscholar.org/CorpusID:232362223).
- 768 Shunxin Xiao, Shiping Wang, Yuanfei Dai, and Wenzhong Guo. Graph neural networks in node
769 classification: survey and evaluation. *Machine Vision and Applications*, 33(1):4, 2022.
- 770 Qizhe Xie, Zihang Dai, Eduard H. Hovy, Minh-Thang Luong, and Quoc V. Le. Unsupervised
771 data augmentation. *ArXiv*, abs/1904.12848, 2019. URL [https://api.semanticscholar.
772 org/CorpusID:139102880](https://api.semanticscholar.org/CorpusID:139102880).
- 773 Da Xu, Chuanwei Ruan, Evren Korpeoglu, Sushant Kumar, and Kannan Achan. Inductive representa-
774 tion learning on temporal graphs. *arXiv preprint arXiv:2002.07962*, 2020.
- 775 Mengde Xu, Zheng Zhang, Han Hu, Jianfeng Wang, Lijuan Wang, Fangyun Wei, Xiang Bai, and
776 Zicheng Liu. End-to-end semi-supervised object detection with soft teacher. In *Proceedings of the*
777 *IEEE/CVF international conference on computer vision*, pp. 3060–3069, 2021.
- 778 An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li,
779 Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin
780 Yang, Jiayi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang,
781 Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tingyu Xia,
782 Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu
783 Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*,
784 2024.
- 785 Xiangli Yang, Zixing Song, Irwin King, and Zenglin Xu. A survey on deep semi-supervised learning.
786 *IEEE Transactions on Knowledge and Data Engineering*, 35(9):8934–8954, 2022.
- 787 Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec.
788 Graph convolutional neural networks for web-scale recommender systems. In *Proceedings of the*
789 *24th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 974–983,
790 2018.
- 791 Le Yu, Leilei Sun, Bowen Du, and Weifeng Lv. Towards better dynamic graph learning: New
792 architecture and unified library. *Advances in Neural Information Processing Systems*, 36:67686–
793 67700, 2023.
- 794 Xingtong Yu, Zhenghao Liu, Xinming Zhang, and Yuan Fang. Node-time conditional prompt learning
795 in dynamic graphs. In *The Thirteenth International Conference on Learning Representations*,
796 2025.
- 797 Fanjin Zhang, Xiao Liu, Jie Tang, Yuxiao Dong, Peiran Yao, Jie Zhang, Xiaotao Gu, Yan Wang,
798 Evgeny Kharlamov, Bin Shao, Rui Li, and Kuansan Wang. OAG: Linking Entities Across Large-
799 Scale Heterogeneous Knowledge Graphs. 35(9):9225–9239, a. ISSN 1558-2191. doi: 10.1109/
800 TKDE.2022.3222168. URL <https://ieeexplore.ieee.org/document/9950622>.
- 801 Fanjin Zhang, Xiao Liu, Jie Tang, Yuxiao Dong, Peiran Yao, Jie Zhang, Xiaotao Gu, Yan Wang,
802 Bin Shao, Rui Li, and Kuansan Wang. OAG: Toward Linking Large-scale Heterogeneous Entity
803 Graphs. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge*
804 *Discovery & Data Mining*, KDD '19, pp. 2585–2595. Association for Computing Machinery,
805 b. ISBN 978-1-4503-6201-6. doi: 10.1145/3292500.3330785. URL [https://doi.org/10.
806 1145/3292500.3330785](https://doi.org/10.1145/3292500.3330785).

810 Jianan Zhao, Meng Qu, Chaozhuo Li, Hao Yan, Qian Liu, Rui Li, Xing Xie, and Jian Tang. Learning
811 on large-scale text-attributed graphs via variational inference. *ArXiv*, abs/2210.14709, 2022. URL
812 <https://api.semanticscholar.org/CorpusID:253117079>.
813

814 Hongkuan Zhou, Da Zheng, Israt Nisa, Vasileios Ioannidis, Xiang Song, and George Karypis.
815 Tgl: A general framework for temporal gnn training on billion-scale graphs. *arXiv preprint*
816 *arXiv:2203.14883*, 2022.

817 Tianyi Zhou, Shengjie Wang, and Jeff Bilmes. Time-consistent self-supervision for semi-supervised
818 learning. In *International conference on machine learning*, pp. 11523–11533. PMLR, 2020.
819

820 Xiaojin Jerry Zhu. Semi-supervised learning literature survey. 2005.

821 Marinka Zitnik, Monica Agrawal, and Jure Leskovec. Modeling polypharmacy side effects with
822 graph convolutional networks. *Bioinformatics*, 34(13):i457–i466, 2018.
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863

864	CONTENTS	
865		
866	1 Introduction	1
867		
868	2 Problem Formulation	2
869		
870		
871	3 Methodology	3
872		
873	3.1 Variational EM Framework	3
874	3.1.1 Evidence Lower Bound (ELBO)	4
875	3.1.2 E-step	4
876	3.1.3 M-step	4
877		
878	3.2 Temporal Curriculum Learning	5
879		
880	3.3 Learning and Optimization	5
881		
882	4 Experiments	5
883		
884	4.1 Experiment Settings	5
885	4.1.1 Datasets	5
886	4.1.2 Baselines	6
887	4.1.3 FLiD: A novel framework	7
888		
889	4.2 RQ1: Main Results	7
890		
891	4.3 RQ2: Pseudo-Label Analysis	8
892	4.3.1 Pseudo-label Supervision Study	8
893	4.3.2 Label Consistency Analysis	8
894		
895	4.4 RQ3: Temporal Curriculum Learning Analysis	9
896		
897	4.5 RQ4: Convergence and Efficiency Analysis	10
898		
899	5 Related Work	10
900	5.1 Dynamic Node Classification	10
901		
902	6 Conclusion	10
903		
904		
905	A Notation Table	19
906		
907	B Derivations	19
908		
909	B.1 Derivation of Eq. (1)	19
910	B.2 Derivation of Eq. (3)	20
911	B.3 Derivation of Eq. (7)	20
912		
913		
914	C Algorithm for Training PTCL	21
915		
916	D Dataset Details	21
917	D.1 Previous datasets	22

918	D.1.1 Dsub	22
919	D.1.2 Wikipedia	22
920	D.1.3 Reddit	23
921	D.2 CoOAG	23
922	D.2.1 Description	23
923	D.2.2 Preprocessing	23
924	D.2.3 Examples	24
925		
926		
927		
928		
929	E Backbone Details	24
930		
931	F More Experiments	25
932	F.1 RQ5: Decoder Comparison	25
933	F.2 RQ6: Hyperparameters Sensitivity	25
934		
935		
936	G More Related Work	26
937	G.1 Pseudo-Labeling	26
938	G.2 Variational EM Framework	27
939		
940		
941	H Relationship with other tasks	27
942		
943		
944	I Discussion	27
945		
946	J Implementation Details	27
947		
948	K Hyperparameters	27
949	K.1 Model Configurations	28
950	K.2 PTCL Hyperparameters	28
951		
952		
953	L The Use of Large Language Models (LLMs)	28
954		
955		
956		
957		
958		
959		
960		
961		
962		
963		
964		
965		
966		
967		
968		
969		
970		
971		

A NOTATION TABLE

In this section, we summarize the notations used throughout the paper in Table 4.

Symbol	Description
\mathcal{G}	Dynamic graph consisting of temporally ordered events
$x(t_i) = (u_i, v_i, t_i)$	Interaction event at timestamp t_i
\mathcal{V}	Set of nodes in the dynamic graph
\mathcal{Y}	Set of possible class labels
y_u^t	Label of node u at timestamp t
\mathcal{T}_u	Set of timestamps at which node u appears
$T_u = \max \mathcal{T}_u$	Final timestamp of node u
\mathcal{Y}_F	Set of final timestamp labels for all nodes
\mathcal{Y}_E	Set of labels at non-final timestamps
T_B	Boundary timestamp separating train/eval partitions
$\mathcal{Y}_{F,B}$	Final labels whose timestamps are before T_B
$\mathcal{Y}_{F,A}$	Final labels whose timestamps are after T_B
$\mathcal{Y}_{E,B}$	Non-final labels before T_B (unknown in our setting)
$n_u \in \mathbb{R}^{d_N}$	Node feature vector
$e_{u,v}^t \in \mathbb{R}^{d_E}$	Edge feature at timestamp t
h_t^u	Backbone-generated embedding of node u at timestamp t
θ	Parameters of the dynamic graph backbone
ϕ	Parameters of the decoder used for label prediction
$\hat{\mathcal{Y}}_{E,B}$	Pseudo-labels generated by the decoder
α, β	Hyperparameters controlling label weighting
τ	EM iteration index
d_u^t	Temporal distance from timestamp t to T_u
$w_u^{t,\tau}$	Temporal curriculum weight for node u at timestamp t in the τ -th EM iteration
γ	Decay coefficient for temporal curriculum learning

Table 4: Summary of notations used in the paper.

B DERIVATIONS

B.1 DERIVATION OF EQ. (1)

The objective of our Variational EM framework is to maximize the marginal log-likelihood of the observed data, $\log p_\theta(\mathcal{Y}_{F,B}|\mathcal{G})$. Since $\mathcal{Y}_{E,B}$ are the latent variables, we can express the marginal log-likelihood by introducing an arbitrary variational distribution $q_\phi(\mathcal{Y}_{E,B}|\mathcal{G})$:

$$\begin{aligned}
\log p_\theta(\mathcal{Y}_{F,B}|\mathcal{G}) &= \mathbb{E}_{q_\phi(\mathcal{Y}_{E,B}|\mathcal{G})} [\log p_\theta(\mathcal{Y}_{F,B}|\mathcal{G})] \\
&= \mathbb{E}_{q_\phi(\mathcal{Y}_{E,B}|\mathcal{G})} \left[\log \left(\frac{p_\theta(\mathcal{Y}_{F,B}, \mathcal{Y}_{E,B}|\mathcal{G})}{p_\theta(\mathcal{Y}_{E,B}|\mathcal{Y}_{F,B}, \mathcal{G})} \right) \right] \\
&= \mathbb{E}_{q_\phi(\mathcal{Y}_{E,B}|\mathcal{G})} \left[\log \left(\frac{p_\theta(\mathcal{Y}_{F,B}, \mathcal{Y}_{E,B}|\mathcal{G})}{q_\phi(\mathcal{Y}_{E,B}|\mathcal{G})} \cdot \frac{q_\phi(\mathcal{Y}_{E,B}|\mathcal{G})}{p_\theta(\mathcal{Y}_{E,B}|\mathcal{Y}_{F,B}, \mathcal{G})} \right) \right] \\
&= \mathbb{E}_{q_\phi(\mathcal{Y}_{E,B}|\mathcal{G})} \left[\log \left(\frac{p_\theta(\mathcal{Y}_{F,B}, \mathcal{Y}_{E,B}|\mathcal{G})}{q_\phi(\mathcal{Y}_{E,B}|\mathcal{G})} \right) \right] + \mathbb{E}_{q_\phi(\mathcal{Y}_{E,B}|\mathcal{G})} \left[\log \left(\frac{q_\phi(\mathcal{Y}_{E,B}|\mathcal{G})}{p_\theta(\mathcal{Y}_{E,B}|\mathcal{Y}_{F,B}, \mathcal{G})} \right) \right] \\
&= \underbrace{\mathbb{E}_{q_\phi(\mathcal{Y}_{E,B}|\mathcal{G})} [\log p_\theta(\mathcal{Y}_{F,B}, \mathcal{Y}_{E,B}|\mathcal{G}) - \log q_\phi(\mathcal{Y}_{E,B}|\mathcal{G})]}_{\text{ELBO, } \mathcal{L}(\theta, \phi)} \\
&\quad + \underbrace{D_{KL} \left(q_\phi(\mathcal{Y}_{E,B}|\mathcal{G}) \parallel p_\theta(\mathcal{Y}_{E,B}|\mathcal{Y}_{F,B}, \mathcal{G}) \right)}_{\geq 0} \\
&\geq \mathbb{E}_{q_\phi(\mathcal{Y}_{E,B}|\mathcal{G})} [\log p_\theta(\mathcal{Y}_{F,B}, \mathcal{Y}_{E,B}|\mathcal{G}) - \log q_\phi(\mathcal{Y}_{E,B}|\mathcal{G})]
\end{aligned}$$

Since the KL divergence $D_{KL}(q_\phi \| p_\theta) \geq 0$, the ELBO $\mathcal{L}(\theta, \phi)$ is indeed a lower bound on the marginal log-likelihood. The Variational EM algorithm iteratively maximizes this lower bound.

B.2 DERIVATION OF EQ. (3)

The goal of the E-step in the Variational EM framework is to find the optimal variational distribution $q_\phi(\mathcal{Y}_{E,B}|\mathcal{G})$ that best approximates the true posterior distribution $p_\theta(\mathcal{Y}_{E,B}|\mathcal{Y}_{F,B}, \mathcal{G})$ by minimizing the Kullback-Leibler (KL) divergence $D_{KL}(q_\phi(\mathcal{Y}_{E,B}|\mathcal{G}) \| p_\theta(\mathcal{Y}_{E,B}|\mathcal{Y}_{F,B}, \mathcal{G}))$.

Directly optimizing this standard KL divergence is challenging because the entropy of the variational distribution $q_\phi(\mathcal{Y}_{E,B}|\mathcal{G})$ is generally computationally intractable to calculate.

Following prior work (Zhao et al., 2022; Qu et al., 2019), we adopt the wake-sleep algorithm (Hinton et al., 1995) for the E-step. This approach shifts the optimization focus to the reverse KL-divergence, $D_{KL}(p_\theta(\mathcal{Y}_{E,B}|\mathcal{Y}_{F,B}, \mathcal{G}) \| q_\phi(\mathcal{Y}_{E,B}|\mathcal{G}))$, which is equivalent to maximizing $\mathbb{E}_{p_\theta(\mathcal{Y}_{E,B}|\mathcal{Y}_{F,B}, \mathcal{G})} [\log q_\phi(\mathcal{Y}_{E,B}|\mathcal{G})]$:

$$\begin{aligned} & - D_{KL}(p_\theta(\mathcal{Y}_{E,B}|\mathcal{Y}_{F,B}, \mathcal{G}) \| q_\phi(\mathcal{Y}_{E,B}|\mathcal{G})) \\ &= \mathbb{E}_{p_\theta(\mathcal{Y}_{E,B}|\mathcal{Y}_{F,B}, \mathcal{G})} [\log q_\phi(\mathcal{Y}_{E,B}|\mathcal{G}) - \log p_\theta(\mathcal{Y}_{E,B}|\mathcal{Y}_{F,B}, \mathcal{G})] \\ &= \mathbb{E}_{p_\theta(\mathcal{Y}_{E,B}|\mathcal{Y}_{F,B}, \mathcal{G})} [\log q_\phi(\mathcal{Y}_{E,B}|\mathcal{G})] + \text{const}, \end{aligned}$$

To further simplify the optimization, we apply the mean-field approximation to the variational distribution $q_\phi(\mathcal{Y}_{E,B}|\mathcal{G})$. Since the dynamic graph backbone θ is fixed during E-step, the node feature \mathbf{h}_u^t is fixed too. Then we can get (Eq. (2) in the main paper):

$$q_\phi(\mathcal{Y}_{E,B}|\mathcal{G}) = \prod_{u \in \mathcal{V}} \prod_{\substack{t \in \mathcal{T}_u \setminus \{T_u\} \\ t \leq T_B}} q_\phi(y_u^t | \mathbf{h}_u^t), \quad (13)$$

Substituting this factorization into our objective gives the objective function $\hat{\mathcal{O}}_\phi$ (Eq. (3) in the main paper):

$$\begin{aligned} \hat{\mathcal{O}}_\phi &= \max_{\phi} \mathbb{E}_{p_\theta(\mathcal{Y}_{E,B}|\mathcal{Y}_{F,B}, \mathcal{G})} [\log q_\phi(\mathcal{Y}_{E,B}|\mathcal{G})] \\ &= \max_{\phi} \mathbb{E}_{p_\theta(\mathcal{Y}_{E,B}|\mathcal{Y}_{F,B}, \mathcal{G})} \left[\log \left(\prod_{u \in \mathcal{V}} \prod_{\substack{t \in \mathcal{T}_u \setminus \{T_u\} \\ t \leq T_B}} q_\phi(y_u^t | \mathbf{h}_u^t) \right) \right] \\ &= \max_{\phi} \mathbb{E}_{p_\theta(\mathcal{Y}_{E,B}|\mathcal{Y}_{F,B}, \mathcal{G})} \left[\sum_{u \in \mathcal{V}} \sum_{\substack{t \in \mathcal{T}_u \setminus \{T_u\} \\ t \leq T_B}} \log q_\phi(y_u^t | \mathbf{h}_u^t) \right] \\ &= \max_{\phi} \sum_{u \in \mathcal{V}} \sum_{\substack{t \in \mathcal{T}_u \setminus \{T_u\} \\ t \leq T_B}} \mathbb{E}_{p_\theta(y_u^t | \mathcal{Y}_{F,B}, \mathcal{G})} [\log q_\phi(y_u^t | \mathbf{h}_u^t)], \end{aligned}$$

B.3 DERIVATION OF EQ. (7)

The M-step of the Variational EM framework aims to update the dynamic graph backbone parameters θ by maximizing the expected log-likelihood while holding the decoder parameters ϕ fixed, that is:

$$\hat{\mathcal{O}}_\theta = \mathbb{E}_{q_\phi(\mathcal{Y}_{E,B}|\mathcal{G})} [\log p_\theta(\mathcal{Y}_{F,B}, \mathcal{Y}_{E,B}|\mathcal{G})],$$

However, this expectation over the latent variables $\mathcal{Y}_{E,B}$ is analytically intractable. To make optimization feasible, we introduce two approximations.

Pseudo-likelihood approximation. Following (Besag, 1975), we approximate the joint by a product of conditional likelihoods:

$$\begin{aligned}\hat{O}_\theta &= \mathbb{E}_{q_\phi(\mathcal{Y}_{E,B}|\mathcal{G})} [\log p_\theta(\mathcal{Y}_{F,B}, \mathcal{Y}_{E,B} | \mathcal{G})] \\ &\approx \mathbb{E}_{q_\phi(\mathcal{Y}_{E,B}|\mathcal{G})} \left[\sum_{u \in \mathcal{V}} \sum_{t \in T_u} \log p_\theta(y_u^t | \mathcal{G}, \mathcal{Y}_{F,B} \cup \mathcal{Y}_{E,B} \setminus \{y_u^t\}) \right],\end{aligned}$$

The pseudo-likelihood approximation replaces an intractable joint with a sum of conditionals, which makes per-label losses available for gradient-based optimization. Then we split the sum into (i) intermediate timestamps $t \in T_u \setminus \{T_u\}$ (pseudo-labels) and (ii) final timestamps $t = T_u$ (observed final labels). To control the relative contribution of pseudo-labels and final labels, we introduce a balancing hyperparameter $\beta \in [0, 1]$. Thus we approximate the expected log-likelihood by:

$$\begin{aligned}\hat{O}_\theta &\approx \mathbb{E}_{q_\phi(\mathcal{Y}_{E,B}|\mathcal{G})} \left[\beta \sum_{u \in \mathcal{V}} \sum_{\substack{t \in T_u \setminus \{T_u\} \\ t \leq T_B}} \log p_\theta(y_u^t | \mathcal{G}, \mathcal{Y}_{F,B}, \mathcal{Y}_{E,B} \setminus \{y_u^t\}) \right. \\ &\quad \left. + (1 - \beta) \sum_{\substack{u \in \mathcal{V} \\ T_u \leq T_B}} \log p_\theta(y_u^{T_u} | \mathcal{G}, \mathcal{Y}_{F,B} \setminus \{y_u^{T_u}\}, \mathcal{Y}_{E,B}) \right].\end{aligned}$$

Here we have explicitly conditioned each factor on the available final labels $\mathcal{Y}_{F,B}$ and on the unknown intermediate labels $\mathcal{Y}_{E,B}$.

Monte Carlo approximation. The expectation over $q_\phi(\mathcal{Y}_{E,B}|\mathcal{G})$ is still expensive. Using a Monte Carlo approximation with a single sample, which is common in practice, and denoting the sampled pseudo-label set by $\hat{\mathcal{Y}}_{E,B}$ as in the main text, we can get:

$$\begin{aligned}\hat{O}_\theta &= \beta \sum_{u \in \mathcal{V}} \sum_{\substack{t \in T_u \setminus \{T_u\} \\ t \leq T_B}} \log p_\theta(y_u^t | \mathcal{G}, \mathcal{Y}_{F,B}, \hat{\mathcal{Y}}_{E,B} \setminus \{y_u^t\}) \\ &\quad + (1 - \beta) \sum_{\substack{u \in \mathcal{V} \\ T_u \leq T_B}} \log p_\theta(y_u^{T_u} | \mathcal{G}, \mathcal{Y}_{F,B} \setminus \{y_u^{T_u}\}, \hat{\mathcal{Y}}_{E,B}),\end{aligned}\tag{14}$$

which is exactly Eq. (7) in the main paper. The Monte Carlo single-sample step removes the expectation over $q_\phi(\mathcal{Y}_{E,B} | \mathcal{G})$ and yields a deterministic training objective in each EM iteration, while a single sample is an efficient and effective choice. It can be seen as a stochastic unbiased estimator of the expectation.

C ALGORITHM FOR TRAINING PTCL

First, we define the link prediction task. Given a dynamic graph $\mathcal{G} = \{(u_i, v_i, t_i)\}$, the link prediction loss is defined as:

$$\mathcal{L}_{lp} = - \sum_{(u_i, v_i, t_i) \in \mathcal{G}} \log \sigma(\text{MLP}(\mathbf{h}_{u_i}^{t_i} \parallel \mathbf{h}_{v_i}^{t_i})) - \sum_{(u_j, v_j, t_j) \notin \mathcal{G}} \log \left(1 - \sigma(\text{MLP}(\mathbf{h}_{u_j}^{t_j} \parallel \mathbf{h}_{v_j}^{t_j})) \right).\tag{15}$$

where $\sigma(\cdot)$ is the sigmoid function, \parallel means concatenation. The first term encourages the model to predict existing edges correctly, while the second term penalizes the model for predicting non-existent edges.

Then the algorithm for training PTCL can be summarized in Alg. 1.

D DATASET DETAILS

Due to the lack of widely studied datasets for the **label-limited dynamic node classification**, we utilize three existing datasets that align closely with our task and propose a new dataset *CoOAG* specifically designed for this problem. The statistics of four datasets are introduced in Table 5.

Algorithm 1 Optimization Algorithm

```

1134 1: Input: A dynamic graph  $\mathcal{G}$ , final timestamp labels  $\mathcal{Y}_{F,B}$ , and hyperparameter  $\beta$ 
1135
1136 2: Output: Predicted  $\hat{\mathcal{Y}}_{F,A}$ .
1137
1138 3:  $\theta \leftarrow \arg \min_{\theta} \mathcal{L}_{lp}$  ▷ Warm up the backbone
1139 4:  $\tau \leftarrow 1$  ▷ Initialize iteration counter
1140 5: repeat
1141 6:   E-Step: Decoder Optimization
1142 7:    $\phi \leftarrow \arg \max_{\phi} \mathcal{O}_{\phi}$  ▷ Update the decoder  $q_{\phi}$ 
1143 8:    $\hat{\mathcal{Y}}_{E,B} \leftarrow \arg \max q_{\phi}(\mathcal{Y}_{E,B}|\mathcal{G})$  ▷ Generate pseudo-labels
1144 9:   M-Step: Backbone Optimization
1145 10:   $w_u^{t,\tau} \leftarrow f_{TW}(t, u, \tau, T_u, \gamma)$  ▷ Compute temporal weights
1146 11:   $\theta \leftarrow \arg \max_{\theta} \mathcal{O}_{\theta}$  ▷ Update backbone  $p_{\theta}$ 
1147 12:   $\tau \leftarrow \tau + 1$  ▷ Update iteration counter
1148 13: until Converged
1149 14:  $\hat{\mathcal{Y}}_{F,A} \leftarrow \arg \max q_{\phi}(\mathcal{Y}_{F,A}|\mathcal{G})$  ▷ Final prediction
1150 15: return  $\hat{\mathcal{Y}}_{F,A}$ 

```

Table 5: Dataset Statistics.

	Wikipedia	Reddit	Dsub	CoOAG
Nodes	9, 227	10, 984	150, 000	9, 559
Edges	15, 7474	67, 2447	16, 8154	11, 4337
Duration	1 month	1 month	1 year	22 years
Total classes	2	2	2	5
Bipartite	✓	✓	✗	✗
Node Feat Dim	–	–	34	384
Edge Feat Dim	172	172	1	384

D.1 PREVIOUS DATASETS

We use three publicly available datasets and do the preprocessing to adapt them to our task:

D.1.1 DSUB

Description: Dsub is a subgraph of the Dgraph (Huang et al., 2022) dataset, which is a financial fraud detection dataset where nodes represent users, and edges represent emergency contact relationships between users. Node labels indicate whether a user is ultimately identified as fraudulent (failing to repay loans over an extended period). Node features are derived from user metadata. In addition to confirmed fraudulent and non-fraudulent labels, the dataset includes background nodes that lack sufficient information for labeling but are retained to maintain graph connectivity.

Preprocessing: To facilitate efficient training, we extract a subgraph called Dsub using Breadth-First Search (BFS), ensuring that the subgraph remains connected and preserves the original label distribution.

D.1.2 WIKIPEDIA

Description: Wikipedia (Kumar et al., 2019) is a bipartite interaction graph that records edits on Wikipedia pages over one month. Nodes represent users and pages, and edges denote editing behaviors with timestamps. Each edge is associated with a 172-dimensional Linguistic Inquiry and Word Count (LIWC) feature. The dataset includes dynamic labels indicating whether users are temporarily banned from editing (Yu et al., 2023).

Task Adaptation: To simulate real-world scenarios where only the final labels are available, we split the dynamic labels into $\mathcal{Y}_{F,B}$ (final labels) and $\mathcal{Y}_{E,B}$ (unobserved labels). During training, only $\mathcal{Y}_{F,B}$ is used.

1188 D.1.3 REDDIT
1189

1190 **Description:** Reddit (Kumar et al., 2019) is a bipartite graph that records user posts under subreddits
1191 over one month. Nodes represent users and subreddits, and edges represent timestamped posting
1192 requests. Each edge is associated with a 172-dimensional LIWC feature. The dataset includes
1193 dynamic labels indicating whether users are banned from posting (Yu et al., 2023).

1194 **Task Adaptation:** Similar to Wikipedia, we split the dynamic labels into $\mathcal{Y}_{F,B}$ and $\mathcal{Y}_{E,B}$, using only
1195 $\mathcal{Y}_{F,B}$ for training to simulate real-world conditions.
1196

1197 D.2 COOAG
11981199 D.2.1 DESCRIPTION
1200

1201 To advance research in this domain, we introduce CoOAG, a novel dataset derived from the academic
1202 sphere, inspired by the Coauthor CS and Coauthor Physics networks (Shchur et al.). This dataset has
1203 undergone stringent quality control and temporal consistency checks. Label distributions are detailed
1204 in Table 6.

1205 The CoOAG dataset is constructed using the Microsoft Academic Graph (MAG) portion from Open
1206 Academic Graph 2.1 (Sinha et al.; Zhang et al., b;a; Tang et al.), with a focus on publications from
1207 leading AI conferences. The node labels in CoOAG denote authors’ research interests, classified into
1208 the following categories:
1209

- 1210 • CV (Computer Vision)
- 1211 • NLP (Natural Language Processing)
- 1212 • ROB (Robotics)
- 1213 • DM/WS (Data Mining/Web Search)
- 1214 • AI/ML (Other AI Fields)

1215 Table 6: Label Distributions of CoOAG.
1216

Field	Label Distribution
ROB	2,845 (29.64%)
CV	1,700 (17.71%)
NLP	1,652 (17.21%)
AI/ML	1,971 (20.53%)
DM/WS	1,431 (14.91%)

1224
1225
1226 D.2.2 PREPROCESSING
1227

1228 Although the Open Academic Graph (OAG) provides research interest tags, these annotations are
1229 static and fail to capture the temporal evolution of scholars’ research trajectories. To address this
1230 limitation, we leverage structured prompts with the Qwen-Plus API (qwe, 2024; Yang et al., 2024)
1231 to dynamically classify authors’ research interests based on their most recent ten publications,
1232 incorporating both abstracts and their associated Fields of Study (FoS). The prompt template, as
1233 illustrated in Listing D.2.2, encompasses:

- 1234 • Category definitions with canonical examples
- 1235 • Strict output format constraints
- 1236 • Weighted keyword matching logic
- 1237 • Interactive classification examples

1238 This approach achieves 98.3% ACC on 120 manually verified samples. Edge features are generated
1239 by concatenating paper metadata and abstracts, encoded using the all-MiniLM-L12-v2 model. Node
1240 features are computed as the average of all paper features for each author. Conference submission
1241 deadlines determine edge timestamps. The classification workflow maintains temporal consistency
by processing papers in chronological order.

1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295

```

Research field Classification Prompt Template

Classify the author's research field into one of the following

5 categories based on the given field keywords and weights:
- 0: CV (Computer Vision)
- 1: NLP (Natural Language Processing)
- 2: ROB (Robotics)
- 3: DM/WS (Data Mining/Web Search)
- 4: AI/ML (Other AI Fields)

Input: Multiple field keywords with weights
Output requirements:
- **(*@\textbf{Format}@*)**: Directly return classification
  result (0-4)
- **(*@\textbf{Constraint}@*)**: Answer must be a single digit
  without explanation

Examples:
- Input: "[computer vision (0.53377)] [image filter (0.5337)]"
- Output: 0

Input:
{fos_text}
    
```

D.2.3 EXAMPLES

To illustrate the temporal dynamics inherent in the CoOAG dataset, we present a concrete example of label evolution for a single author node. Consider **Node ID: 6816**, a researcher whose publication history spans 3,693 days (approximately 10 years). This node undergoes three label transitions across distinct research fields, reflecting meaningful shifts in academic focus:

Table 7: Label transitions of node id: 6816 over time.

Time interval	Number of papers	Research field (label)
2010.11.15 – 2012.09.10	5	Robotics (ROB)
2012.09.10 – 2014.12.15	7	Data Mining / Web Search (DM/WS)
2014.12.15 – 2016.12.04	6	Robotics (ROB)
2016.12.04 – 2020.12.25	24	Data Mining / Web Search (DM/WS)

This trajectory demonstrates repeated and substantial shifts between Robotics and Data Mining/Web Search, underscoring the non-stationary nature of research interests over time. Such patterns are not isolated: across the entire CoOAG dataset, 43.2% of labeled nodes experience at least one label transition during their publication lifetime, with an average of 1.19 label changes per node. These statistics confirm that label dynamics in CoOAG are both frequent and semantically meaningful, capturing real-world academic evolution.

E BACKBONE DETAILS

- **TGAT** (Xu et al., 2020) leverages a self-attention mechanism to simultaneously capture spatial and temporal dependencies. Initially, TGAT combines the raw node feature \mathbf{n}_u with a learnable time encoding $z(t)$, forming $\mathbf{n}_u(t) = [\mathbf{n}_u || z(t)]$, where $z(t) = \cos(tw + b)$. Subsequently, self-attention is applied to generate the representation of node u at time t_0 , denoted as $\mathbf{h}_u^{t_0} = \text{SAM}(\mathbf{n}_u(t_0), \{\mathbf{n}_v(m_v) \mid v \in N_{t_0}(u)\})$. Here, $N_{t_0}(u)$ represents the set of neighbors of node u

- at time t_0 , and m_v indicates the timestamp of the most recent interaction involving node v . Finally, predictions for any node pair (u, v) at time t_0 are obtained via $\text{MLP}([\mathbf{h}_u^{t_0} || \mathbf{h}_v^{t_0}])$.
- **TCL** (Wang et al., 2021a) adopts a contrastive learning framework. To construct interaction sequences for each node, TCL employs a breadth-first search algorithm on the temporal dependency subgraph. A graph transformer is then utilized to learn node representations by jointly considering graph topology and temporal dynamics. Additionally, TCL integrates a cross-attention mechanism to model the interdependencies between interacting nodes.
 - **TGN** (Rossi et al., 2020) combines RNN-based and self-attention-based techniques. TGN maintains a memory module to store and update the state $s_u(t)$ of each node u , which serves as a compact representation of u 's historical interactions. Given the memory updater as `mem`, when an edge $e_{uv}(t)$ connecting nodes u and v is observed, the memory state of node u is updated as $s_u(t) = \text{mem}(s_u(t^-), s_v(t^-) || \mathbf{e}_{u,v}^t)$, where $s_u(t^-)$ denotes the memory state of u just prior to time t , and $\mathbf{e}_{u,v}^t$ represents the edge feature. The memory updater `mem` is implemented using a recurrent neural network (RNN). Node embeddings \mathbf{h}_u^t are computed by aggregating information from the L -hop temporal neighborhood through self-attention.
 - **GraphMixer** (Cong et al., 2023) introduces a simple yet effective MLP-based architecture. Instead of relying on trainable time encodings, GraphMixer utilizes a fixed time encoding function, which is integrated into a link encoder based on MLP-Mixer to process temporal links. A node encoder with neighbor mean-pooling is employed to aggregate node features. Specifically, for each node u , GraphMixer computes its embedding \mathbf{h}_u^t by summarizing the features of its neighbors within the temporal context.
 - **DyGFormer** (Yu et al., 2023) employs a self-attention mechanism to model dynamic graphs. For a given node u , DyGFormer retrieves the features of its involved neighbors and edges to represent their encodings. It incorporates a neighbor co-occurrence encoding scheme, which captures the frequency of each neighbor's appearance in the interaction sequences of both the source and destination nodes, thereby explicitly exploring pairwise correlations. Rather than operating at the interaction level, DyGFormer divides the interaction sequences of each source or destination node into multiple patches, which are then processed by a transformer to compute node embeddings \mathbf{h}_u^t .

F MORE EXPERIMENTS

To further validate our framework, we additionally investigate the following research questions (RQs):

RQ5: How does decoder design impact PTCL's performance? **RQ6:** Is PTCL robust to the choice of temporal decay rate γ ?

F.1 RQ5: DECODER COMPARISON

In our framework, we follow the modular paradigm commonly adopted in temporal graph learning, where the encoder (backbone) captures dynamic structural-temporal features, and the decoder serves as a lightweight task-specific mapping function from node embeddings to labels (Rossi et al., 2020). While pseudo-labels play a critical role in training under the label-limited setting, their quality is primarily determined by the encoder's representation capacity rather than the complexity of the decoder.

To further examine the role of the decoder, we conducted additional experiments with more expressive designs, including deep 8-layer MLP and Transformer-based architectures. Interestingly, these complex decoders often resulted in performance degradation, which can be attributed to overfitting, increased variance, or optimization instability under the low-label regime. Figure 5 summarizes the performance comparison on the Wikipedia dataset, showing that our lightweight MLP decoder achieves competitive or superior results across backbones compared with deep MLP and Transformer decoders.

F.2 RQ6: HYPERPARAMETERS SENSITIVITY

To evaluate the sensitivity of our framework to the temporal decay rate γ , we conducted additional experiments on the Wikipedia dataset by varying γ from 0.1 to 0.9. We report results across five different backbones in Table 8. Overall, the performance remains relatively stable across settings, with moderate standard deviations: TGAT (83.15 ± 1.39), TGN (86.25 ± 0.76), GraphMixer (82.47

1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403

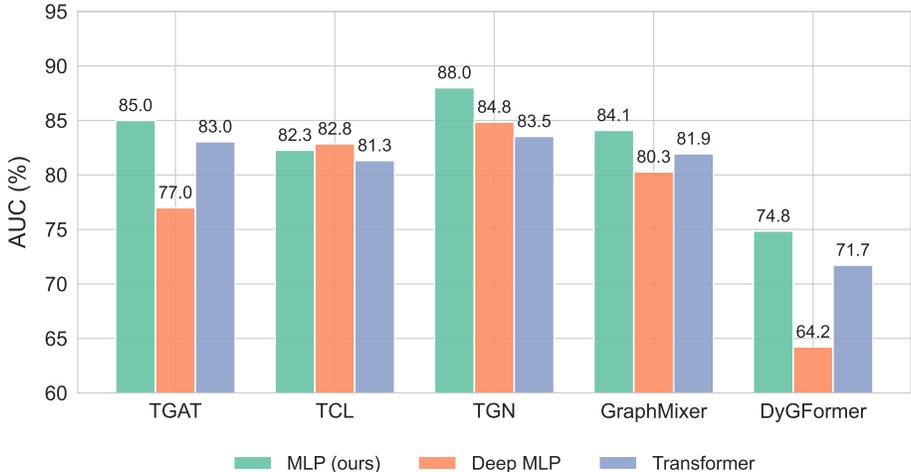


Figure 5: Decoder Comparison on Wikipedia.

± 0.57), TCL (80.30 ± 1.75), and DyGFormer (71.85 ± 2.24). These results suggest that our method is robust to the choice of γ .

Table 8: Sensitivity analysis of our framework to the temporal decay rate γ on the Wikipedia dataset.

Model	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	Avg \pm Std
TGAT	80.94	81.58	84.24	83.13	84.48	83.15	81.94	85.52	83.40	83.15 ± 1.39
TGN	85.58	87.03	85.27	87.01	86.57	87.17	85.03	86.09	86.52	86.25 ± 0.76
GraphMixer	82.37	81.23	83.44	82.27	82.92	82.36	82.86	82.50	82.30	82.47 ± 0.57
TCL	76.69	80.41	81.97	81.31	80.93	82.27	81.26	79.90	78.00	80.30 ± 1.75
DyGFormer	73.64	73.74	72.79	68.73	73.90	74.49	70.29	70.84	68.24	71.85 ± 2.24

G MORE RELATED WORK

G.1 PSEUDO-LABELING

Pseudo-labeling (Lee et al., 2013) is a widely used semi-supervised learning method that assigns labels to unlabeled data to reduce entropy and encourage low-density decision boundaries (Pei et al., 2024; Chapelle & Zien, 2005; Grandvalet & Bengio, 2004). It has proven effective in various fields including computer vision (Lee et al., 2013; Xu et al., 2021), graph learning (Li et al., 2022b; 2018b;a; Sun et al., 2019), knowledge distillation (Hinton et al., 2015), and adversarial training (Miyato et al., 2017; Xie et al., 2019). However, performance heavily depends on label quality, as noisy pseudo-labels may misguide training (Pei et al., 2024). Curriculum Learning (Bengio et al., 2009) mitigates this by ordering training samples from easy to hard, using heuristics like confidence or entropy (Cascante-Bonilla et al., 2020; He et al., 2022; Pei et al., 2024; Song et al., 2019). In contrast, our **Temporal Curriculum Learning** explicitly leverages temporal information to prioritize recent (easier) timestamps first, then gradually incorporates earlier (harder) ones, better aligning model training with evolving dynamics in time-dependent graphs.

A closely related work is ELI (Kamhoua et al.), which also addresses graph learning under limited supervision by introducing pseudo-labels and decoupling their generation from final label supervision. Despite this similarity, there are key differences. First, ELI is designed for static graphs with globally sparse labels, whereas our work considers dynamic graphs where labels are typically only available at the final timestamp, introducing temporally-driven label scarcity. Second, ELI infers pseudo-labels through an unsupervised label distribution estimation and constructs a pseudo-label graph, while our PTCL framework treats pseudo-labels as latent variables and refines them iteratively via a variational

EM procedure, with a decoder trained solely on ground-truth labels to ensure alignment with the true label space. Finally, ELI integrates labels through graph-based regularization using the Laplacian of the pseudo-label graph, whereas PTCL directly supervises the backbone with both pseudo- and ground-truth labels, augmented by a temporal curriculum weighting scheme that prioritizes recent timestamps. These distinctions highlight the novelty of our temporal perspective in pseudo-label integration for dynamic graphs.

G.2 VARIATIONAL EM FRAMEWORK

The variational EM framework (Dempster et al., 1977; Neal & Hinton, 1998) is a widely used framework for parameter estimation in probabilistic models with latent variables. In the classical EM algorithm, the goal is to maximize the likelihood of observed data by iteratively refining model parameters through alternating E-steps (expectation computation) and M-steps (parameter maximization). GMNN (Qu et al., 2019) applies EM for semi-supervised static graphs and GLEM (Zhao et al., 2022) combines GNNs with language models. Our contribution lies not in framework innovation but in adapting this established paradigm to address label-limited dynamic node classification.

H RELATIONSHIP WITH OTHER TASKS

In Section 2, we define the *label-limited dynamic node classification* task. Specifically, when all timestamps t_i are identical or omitted, the graph degenerates into a static graph (Kipf & Welling, 2016; Holme & Saramäki, 2012), where each node $u \in \mathcal{V}$ is associated with a single label y_u . Then the problem degrades to a *static node classification* task. Alternatively, if labels are available for all nodes at all timestamps, i.e., \mathcal{Y}_E is entirely known, the problem becomes a *fully supervised dynamic node classification* task, which has been extensively studied in prior research (Xu et al., 2020; Rossi et al., 2020; Cong et al., 2023; Yu et al., 2023).

I DISCUSSION

Limitation. Due to the open-access nature and the unique characteristics of our task setting, we were unable to evaluate our method on a broader range of datasets. This may limit the generalizability of our findings to other domains or data types. Additionally, we observed that performance can be influenced by the specific computational environment, including hardware and runtime configurations. As a result, reproduction under different resource settings may lead to variations in absolute performance.

Future Work. Although our Temporal Curriculum Learning outperforms alternative dynamic weighting strategies based on confidence or entropy (e.g., CST and EST), we acknowledge that our current design still relies on a manually set temporal decay parameter γ . A valuable future direction is to develop a confidence-aware weighting scheme that adaptively modulates pseudo-label contributions according to their reliability. In addition, we plan to explore the integration of large language models (LLMs) into our framework, leveraging their strong reasoning and representation capabilities to further enhance pseudo-label generation and temporal modeling in dynamic graphs.

J IMPLEMENTATION DETAILS

We use PyTorch (Paszke et al., 2019), scikit-learn (Pedregosa et al., 2011), PyTorch Geometric (Fey & Lenssen, 2019), DyGLib (Yu et al., 2023) library to implement our proposed framework FLiD. We conduct experiments on two clusters: (1) 4×Tesla V100 (32GB memory) using 16-core CPUs and 395GB RAM; (2) 8×2080Ti (11GB memory) using 12-core CPUs and 396GB RAM.

K HYPERPARAMETERS

We optimize all methods across all models using the Adam optimizer (Kingma & Ba, 2014), with cross-entropy loss as the objective function. Initially, we warm up all backbones through link prediction tasks (Kumar et al., 2019). Subsequently, the entire models are trained for 100 epochs, employing an early stopping strategy with a patience of 15. For consistency, we set the learning

rate to 0.0001 and the batch size to 200 across all methods and datasets. To ensure robustness and minimize deviations, we conduct five independent runs for each method with random seeds ranging from 0 to 4 and report the average performance (Yu et al., 2023).

K.1 MODEL CONFIGURATIONS

Here, we present the configurations for each model (Table 9): TGAT, TGN, TCL, GraphMixer, and DyGFormer, all of which remain consistent across datasets.

K.2 PTCL HYPERPARAMETERS

Here, we present the hyperparameters of PTCL (Table 10): β is a hyperparameter that balances the weight of pseudo-labels and final timestamp labels; γ is a hyper-parameter that controls the rate of Temporal Curriculum decay. Note that TGN runs out of memory on Dsub due to its high space cost.

Table 9: Model Configurations Comparison.

Hyperparameter	TGAT	TGN	TCL	GraphMixer	DyGFormer
Time encoding dim	100	100	100	100	100
Output dim	172	172	172	172	172
Attention heads	2	2	2	-	2
Graph conv layers	2	1	-	-	-
Transformer layers	-	-	2	-	2
MLP-Mixer layers	-	-	-	2	-
Node memory dim	-	172	-	-	-
Depth encoding dim	-	-	172	-	-
Co-occurrence dim	-	-	-	-	50
Aligned encoding dim	-	-	-	-	50
Memory updater	-	GRU	-	-	-
Time gap T	-	-	-	2000	-

Table 10: Hyperparameters of PTCL.

Model	Hyperparameters	Wikipedia	Reddit	Dsub	CoOAG
TGAT	β	0.9	0.9	0.7	0.2
	γ	0.8	0.1	0.2	0.9
TCL	β	0.1	0.9	0.1	0.8
	γ	0.6	0.9	0.5	0.6
TGN	β	0.9	0.9	-	0.9
	γ	0.05	0.01	-	0.1
GraphMixer	β	0.5	0.5	0.5	0.5
	γ	1.3	0.1	0.1	0.4
DyGFormer	β	0.7	0.5	0.1	0.1
	γ	0.01	0.01	0.5	0.1

L THE USE OF LARGE LANGUAGE MODELS (LLMs)

We employed large language models (LLMs) as auxiliary tools to enhance the clarity and readability of the text. Specifically, LLMs were used to assist in identifying and correcting grammar and spelling errors, as well as refining the overall expression of our writing. The role of LLMs was limited to language polishing, ensuring that the final manuscript meets academic writing standards.