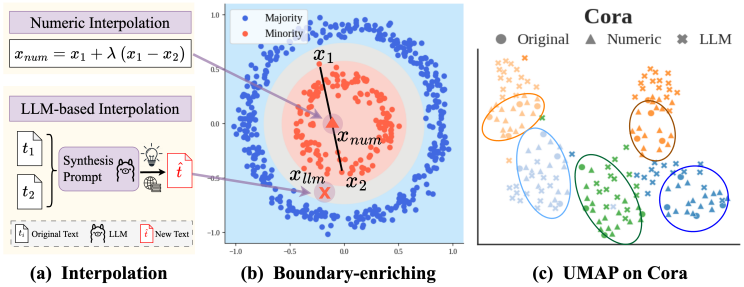
SAVE-TAG: LLM-based Interpolation for Long-Tailed Text-Attributed Graphs

Leyao Wang[1] **Yu Wang**[2] **Bo Ni**[3]
Yuying Zhao[3] **Hanyu Wang**[4] **Yao Ma**[5] **Tyler Derr**[3]
[1]Yale University [2]University of Oregon [3]Vanderbilt University
[4]Renmin University of China [5]Rensselaer Polytechnic Institute
leyao.wang.lw855@yale.edu yuwang@uoregon.edu
{bo.ni,yuying.zhao,tyler.derr}@vanderbilt.edu hy.wang@ruc.edu.cn may13@rpi.edu

Abstract

Real-world graph data often follows long-tailed distributions, making it difficult for Graph Neural Networks to generalize well across both head and tail classes. Recent advances in Vicinal Risk Minimization (VRM) have shown promise in mitigating class imbalance with numeric interpolation; however, existing approaches largely rely on embedding-space arithmetic, which fails to capture the rich semantics inherent in text-attributed graphs. In this work, we propose our method **SAVE-TAG** (**S**emantic-**a**ware **V**icinal Risk Minimization for Long-Tailed **T**ext-**A**ttributed **G**raphs), a novel VRM framework that leverages Large Language Models to perform text-level interpolation, generating on-manifold, boundary-enriching synthetic samples for minority classes. To mitigate the risk of noisy generation, we introduce a confidence-based edge assignment that uses graph topology as a natural filter to ensure structural consistency. We provide theoretical justification for our method and conduct extensive experiments on benchmarks, showing that our approach consistently outperforms both numeric interpolation and prior long-tailed node classification baselines. Our results highlight the importance of integrating semantic and structural signals for effective learning on text-attributed graphs.



(a) Interpolation (b) Boundary-enriching (c) UMAP on Cora

Figure 1: **Comparison of numeric and LLM-based interpolation.**(a) Illustration of the two interpolation strategies; (b) Advantages of LLM-based interpolation in producing boundary-enriching samples are highlighted (visualization uses synthetic data for illustration); (c) UMAP projection of the original and interpolated samples from the Cora dataset—the original data boundary is outlined, with numeric samples mostly staying within and LLM-generated samples extending beyond.

1 Introduction

Graph Neural Networks (GNNs) have proven effective for node classification by modeling structural dependencies in graph data. However, real-world graphs frequently exhibit long-tail distributions [1–3], making it difficult to learn from underrepresented classes.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

Recent approaches to data imbalance often adopt the Vicinal Risk Minimization (VRM) principle [4], which minimizes loss over a synthetic neighborhood of the training data—extended from Empirical Risk Minimization (ERM) [5] that focuses solely on the existing data points. Typically, VRM is instantiated via interpolation techniques like Mixup [6] and SMOTE [7], and has been adapted for long-tailed graph learning in methods including GraphSMOTE [8] and MixupForGraph [9], etc. However, these methods often apply mathematic operations over input embeddings or hidden features, failing to preserve the rich manifold structure of text-attributed graphs (TAGs).

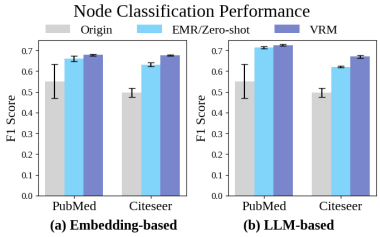


Figure 2: Effect of VRM.

Advances in large language models (LLMs) present new opportunities for data augmentation by offering external knowledge and creative generation. However, prior LLM-based augmentation approaches typically rely on zero-shot or few-shot prompting [10, 11], which usually fail to generate samples within the neighborhood of minority instances, as required by the VRM principle. Nevertheless, VRM has been demonstrated critical in improving long-tailed node classification. As shown in Figure 2a, VRM-guided numeric interpolation with nearest neighbors yields synthetic embeddings that outperform ERM-style embedding duplication.

Motivated by this insight, we hypothesize that prompting strategies explicitly designed to mimic interpolation will surpass standard zero/few-shot approaches, which is subsequently validated by the results in Figure 2b. Furthermore, integrating controlled prompting into LLM generation can alleviate common pitfalls of numeric interpolation, enabling manifold-aware, boundary-enriching augmentation that effectively exploits the knowledge encoded in LLMs (see Figure 1).

While LLM-based interpolation shows great potential, it also carries the risk of generating noisy samples. To mitigate this, we leverage the inherent structure of the graph as a natural filter. Using a confidence-based edge assignment strategy, noisy samples tend to remain isolated, limiting their influence on downstream tasks. In contrast, synthesized nodes aligned with the original distribution will be connected with the original graph, allowing GNN layers to reinforce their alignment through neighborhood aggregation.

In summary, we introduce a novel interpolation framework, SAVE-TAG, which unifies textual semantics and graph topology. Our method employs LLM-based interpolation to expand the vicinal boundary and reduce vicinal risk, followed by a topology-aware filtering mechanism that preserves alignment within the underlying data distribution. Our key contributions are summarized as follows:

- To the best of our knowledge, we present the first semantic-aware VRM framework that employs LLM-based interpolation for node classification on long-tailed text-attributed graphs (TAGs).
- We provide theoretical insights showing that LLM interpolation can generate manifold-preserving, boundary-enriching samples that effectively minimize vicinal risk, while a topology-aware filter helps mitigate noisy generation in the synthesized data.
- We conduct extensive experiments showing our method outperforms embedding-based and long-tail baselines, highlighting the value of semantic and structural signals in imbalanced learning.

2 Problem Formulation

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{T})$ be an attributed graph with $|\mathcal{V}| = n$ nodes, feature matrix* $\mathcal{X} \in \mathbb{R}^{n \times d}$, and labels $\mathcal{Y} = \{y_v | v \in \mathcal{V}\}$ with $y_v \in \{1, \ldots, C\}$. Class imbalance induces a skewed empirical distribution $\mathcal{P}$ on $(\mathcal{X}, \mathcal{Y})$. Our goal is to learn a node-level classifier $f_\theta : \mathbb{R}^d \times \mathcal{V} \to \Delta^{C-1}$, e.g., a GNN, that maximizes classification performance.

3 SAVE-TAG

The SAVE-TAG pipeline is shown in Figure 3, with pseudocode in Algorithm 1 (Appendix B). It consists of: (1) **Vicinal Twin Identification:** find same-label twins in tail classes; (2) **LLM Synthesis:** generate class-consistent texts via a class-conditioned LLM; (3) **Confidence-aware Edge Assignment:** connect aligned samples and isolate noisy ones; (4) **GNN Training:** train a message-passing GNN on the augmented graph $\mathcal{G}' = (\mathcal{V}', \mathcal{E}', \mathcal{T}')$.

*For text-attributed graphs, each row of $\mathcal{X}$ stores a tokenized document.

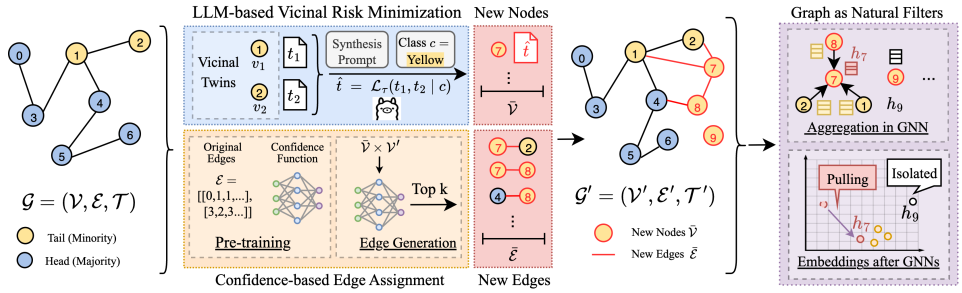


Figure 3: An overview of SAVE-TAG. Given an input graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{T})$, we perform *LLM-based interpolation* on identified *vicinal twins* to synthesize boundary-enriching samples $\hat{t}$ that minimize vicinal risk. A *confidence function* is then pre-trained on the original graph and later used to assign edges to the synthetic nodes via a top-k selection strategy. When the resulting augmented graph $\mathcal{G}' = (\mathcal{V}', \mathcal{E}', \mathcal{T}')$ is processed by a GNN, such edge assignment incorporates well-aligned nodes into their vicinity while isolating noisy samples.

Steps (1) and (2) are incorporated into *LLM-based Vicinal Risk Minimization* (§ 3.1), while steps (3) and (4) are unified under *Graph as Natural Filters* (§ 3.2), detailed below. Complete proofs of all theorems are provided in Appendix F.

3.1 LLM-based Vicinal Risk Minimization

Vicinal Twins for Tail Classes. Let $\mathcal{C}_t \subseteq \{1, \ldots, C\}$ denote the set of tail classes. For any text-attributed node v_1 with $y_{v_1} \in \mathcal{C}_t$, we define its vicinal twins as $\mathcal{N}_k^{(\text{tail})}(v_1) := \{ v_2 \in \mathcal{V} \mid y_{v_2} = y_{v_1},\, v_2 \in \mathcal{N}_k(v_1) \}$.

LLM-based interpolation. For any pair of vicinal twins (v_i, v_j) and their corresponding text attributes t_{v_i} and t_{v_j}, we generate a synthetic text $\hat{t}_{ij}$ by querying the LLM (denoted as $\mathcal{L}_\tau$): $\hat{t}_{ij} = \mathcal{L}_\tau(t_{v_i}, t_{v_j} \mid y_{v_i})$. This yields a new text-attributed node $\hat{v}_{ij}$ with label $y_{\hat{v}_{ij}} = y_{v_i}$. We collect the set of synthetic nodes as $\bar{\mathcal{V}} := \hat{v}_{ij}$ and their corresponding texts as $\bar{\mathcal{T}} := \hat{t}_{ij}$.

3.1.1 Staying on the Manifold

Let $\phi : \mathcal{T} \to \mathcal{X}$ be a text encoder, and $x_v = \phi(t_v)$ the embedding of node v with text $t_v \in \mathcal{T}$. For each class $c \in \{1, \ldots, C\}$, the class manifold is $\mathcal{M}_c = \{x_v \mid y_v = c\}$, with $\mathcal{M} = \bigcup_{c=1}^{C} \mathcal{M}_c$. Let (v_1, v_2) be a pair of *vicinal twins* with texts (t_1, t_2).

Theorem 3.1 (Manifold-Preserving Class-Consistent Generation)**.** *Let $\hat{t} \sim \mathcal{L}_\tau(t_1, t_2 \mid c)$ be a synthetic text from an LLM conditioned on class c. If $p(t \mid y = c)$ places mass at least $1 - \delta$ on $\mathcal{M}_c$, and $\mathcal{L}_\tau(\cdot \mid c)$ approximates $p(\cdot \mid y = c)$ within total-variation error ϵ, then $\hat{t}$ lies in $\mathcal{M}_c$ with probability at least $1 - (\delta + \epsilon)$.*

3.1.2 Boundary-Enriching Effect of LLM Samples

Let $\mathcal{X} \subseteq \mathbb{R}^d$ be the input space and $\mathcal{Y} = \{1, \ldots, C\}$ the label set. A classifier f outputs logits $f^{(c)}(x)$ for class $c \in \mathcal{Y}$ and the logit margin is denoted as γ. We augment $\mathcal{X}$ with m LLM-generated pairs $\{(\hat{x}_j, \hat{y}_j)\}_{j=1}^m$ to form $\mathcal{X}' := \mathcal{X} \cup \{(\hat{x}_j, \hat{y}_j)\}_{j=1}^m$.

Definition 3.2 (Boundary-Coverage Rate (BCR))**.** *A labeled sample (x, y) is a **boundary sample** if the majority of its k Nearest Neighbors have labels different from y [12]. Boundary-Coverage Rate (BCR) is the fraction of boundary samples in $\mathcal{X}'$.*

Theorem 3.3 (Margin Lower Bound)**.** *Let $\gamma_{\min}(\mathcal{X})$ and $\gamma_{\min}(\mathcal{X}')$ be the minimum margins of models trained on the original dataset $\mathcal{X}$ and augmented dataset $\mathcal{X}'$, respectively. Then, adapting [13], for some slack parameter $\eta > 0$, $\gamma_{\min}(\mathcal{X}') \geq \gamma_{\min}(\mathcal{X}) - \eta\,(1 - \text{BCR})$.*

3.1.3 Vicinal Risk Minimization with LLM-based Neighborhood

Let f_θ^{aug} denote the model trained by VRM [4] on the augmented dataset $\mathcal{X}'$, and let f_θ^{orig} denote the model trained without augmentation on $\mathcal{X}$, whose vicinal risk is denoted as R_{vrm}.

Theorem 3.4 (Boundary coverage ⇒ lower vicinal risk)**.** *Assume the loss function $\mathcal{J}$ is L-Lipschitz in its first argument. Then, for any tolerance parameter $\zeta > 0$, $R_{\mathrm{vrm}}(f_\theta^{\mathrm{aug}}) \leq R_{\mathrm{vrm}}(f_\theta^{\mathrm{orig}}) - L\,\gamma_0\,(\mathrm{BCR} - \zeta) + \mathcal{O}(\delta)$, where L is the Lipschitz constant of the loss, $\gamma_0 := \gamma_{\min}(\mathcal{X})$ is the minimum margin before augmentation.*

3.2 Graph as a Natural Filter

Let $\bar{\mathcal{V}}$ denote the set of synthetic nodes, and $\bar{\mathcal{T}} = \{\hat{t}_{\hat{v}} : \hat{v} \in \bar{\mathcal{V}}\}$ be their corresponding texts, where each $\hat{v}$ is a newly generated text-attributed node produced by *LLM-based Vicinal Risk Minimization* (see § 3.1). We employ an encoder $\phi : \mathcal{T} \cup \bar{\mathcal{T}} \to \mathcal{X}'$ to obtain textual representations.

3.2.1 Confidence function.

Using the frozen encoder ϕ, each node v (real or synthetic) is mapped to $z_v = \mathrm{enc}(\phi(t_v)) \in \mathbb{R}^p$ via an MLP encoder. For any edge (u, v), a predictor MLP outputs a logit $s_{u,v} = \mathrm{pred}(\langle z_u, z_v \rangle)$. We train enc and pred on the original graph with a `LogSigmoid` loss so true edges score higher than non-edges. At test time, logits are converted to pairwise confidence $\kappa(u, v) = \sigma(s_{u,v}) \in [0, 1]$, where σ is monotonic, making ranking by κ or s equivalent.

3.2.2 Confidence-aware edge assignment.

Let $\bar{\mathcal{V}}$ be the set of synthetic nodes, $\mathcal{V}$ the set of original nodes, and $\mathcal{V}' := \mathcal{V} \cup \bar{\mathcal{V}}$ the complete augmented node set. For each candidate pair $(u, \hat{v}) \in \mathcal{V}' \times \bar{\mathcal{V}}$, we compute $\kappa(u, \hat{v})$.

Global top-K selection. We form the score matrix over all pairs $\mathcal{P} = \mathcal{V}' \times \bar{\mathcal{V}}$ and select the top-K edges, $\bar{\mathcal{E}} = \arg\mathrm{topK}_{K,(u,\hat{v})\in\mathcal{P}}\, \kappa(u, \hat{v})$, with $K = k|\bar{\mathcal{V}}|$ (avg. k edges per synthetic node). The augmented graph is $\mathcal{G}' = (\mathcal{V}', \mathcal{E} \cup \bar{\mathcal{E}})$, used for downstream node classification.

3.2.3 Denoising in GNN Aggregation.

Aggregator. Adapting [14, 15], consider a generic message-passing layer that updates the hidden representation $h_v^{(\ell)} \in \mathbb{R}^p$ of node v at layer ℓ by blending its own transformed features with a weighted sum of its neighbors' transformed features:

$$h_v^{(\ell+1)} \;=\; \alpha\, \mathbf{W}\, h_v^{(\ell)} \;+\; (1-\alpha) \sum_{u \in \mathcal{N}(v)} \beta_{vu}\, \mathbf{W}\, h_u^{(\ell)}, \tag{1}$$

where $\alpha \in [0, 1]$ is a mixing coefficient, $\mathbf{W} \in \mathbb{R}^{p\times p}$ is a trainable weight matrix, $\mathcal{N}(v)$ denotes the neighbor set of v in $\mathcal{G}'$, and $\beta_{vu} \geq 0$ are neighbor weights with $\sum_{u\in\mathcal{N}(v)} \beta_{vu} = 1$.

Representation manifolds. After training, each class $c \in \{1, \ldots, C\}$ forms a representation manifold $\mathcal{M}'_c \subset \mathbb{R}^p$ in the learned feature space. For any set $\mathcal{S} \subset \mathbb{R}^p$, write $\mathrm{dist}(\xi, \mathcal{S}) = \inf_{y\in\mathcal{S}} \|\xi - y\|_2$ for the Euclidean distance from a point ξ to the closest point in $\mathcal{S}$.

Theorem 3.5 (Confidence ⇒ pulling)**.** *Let $\hat{v} \in \bar{\mathcal{V}}$ be synthetic with label c and define the confidence-filtered anchor set (for threshold $\tau \in (0, 1)$) $\mathcal{N}(\hat{v}) := \{\, u \in \mathcal{V} : \kappa(\hat{v}, u) \geq \tau \,\} \neq \varnothing$. Assume every neighbor $u \in \mathcal{N}(\hat{v})$ satisfies $\mathrm{dist}\big(h_u^{(\ell)}, \mathcal{M}'_c\big) \leq \varepsilon$ for some $\varepsilon \geq 0$. Then there exists a contraction factor $\varrho \in (0, 1)$ (independent of κ) such that $\mathrm{dist}\big(h_{\hat{v}}^{(\ell+1)}, \mathcal{M}'_c\big) \leq \varrho\, \mathrm{dist}\big(h_{\hat{v}}^{(\ell)}, \mathcal{M}'_c\big)$, and iterating* (1) *yields exponential convergence toward $\mathcal{M}'_c$ [16–18].*

Theorem 3.6 (No confidence ⇒ isolation)**.** *If $\kappa(\hat{v}, u) < \tau$ for all $u \in \mathcal{V}$, then $\mathcal{N}(\hat{v}) = \varnothing$ and the update reduces to $h_{\hat{v}}^{(\ell+1)} = \alpha\, \mathbf{W}\, h_{\hat{v}}^{(\ell)}$ for all ℓ. Thus $\hat{v}$ neither influences nor is influenced by $\mathcal{G}'$.*

4 Empirical Results

We investigate three research questions (RQs):

- **RQ1:** Can LLMs generate boundary-enriching samples via manifold interpolation to enable VRM for textual data?
- **RQ2:** Can confidence-based edge assignment effectively filter noisy samples?
- **RQ3:** Does the full framework outperform state-of-the-art methods on long-tailed TAGs?

We next outline the setup and present results. Overall, our method consistently surpasses existing baselines, demonstrating strong effectiveness for node classification on long-tailed TAGs.

4.1 Experimental Setup

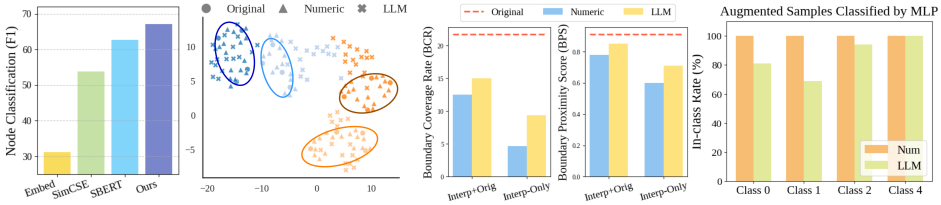
Datasets. Following prior work [9, 19, 20], we use three citation networks (Cora [21], PubMed, Citeseer [22]) and three larger Amazon datasets (Photo, Computer, Children [23]). Dataset statistics are in Table 9 (Appendix C).

Metrics. We report bAcc, macro-F1, GMean, and Acc over five runs with fixed seeds, and assess boundary enrichment with Boundary Coverage Ratio (BCR), Boundary Proximity Score (BPS), and In-Class Rate (ICR), defined in Appendix A.

Baselines. We compare against: (1) *LLM-based augmentation* (zero-shot and few-shot [10, 11]); and (2) *long-tailed graph learning* methods, including Oversampling [24], SMOTE [7], Embed-SMOTE [25], MixupForGraph [9], GraphSMOTE$_T$ and GraphSMOTE$_O$ [8], LTE4G [3], and Hier-Tail [2]. We also design three SAVE-TAGvariants with different VRM strategies (Appendix B).

Models. Unless otherwise noted, text is generated with Llama3-8B-Instruct, encoded by Sentence-BERT (`all-MiniLM-L6-v2`). Node classification uses a GCN.

4.2 Semantic-aware Benefits (RQ1)



(a) Node Classification (b) UMAP (c) BCR and BPS (d) In-class Rate

Figure 4: LLM interpolation preserves manifolds and enriches boundaries on the Citeseer dataset.

On-Manifold. To compare on-manifold LLM-based and off-manifold numeric interpolation, we evaluate node classification on multiple benchmarks. Figure 4(a) shows results on Citeseer. *Embed* leverages LLaMA 3.2-1B embeddings for interpolation, while *SimCSE* and *SBERT* serve as alternative embedding-based baselines. Our LLM-based method using LLaMA 3.2-1B consistently outperforms all numeric interpolation baselines. Similar trends on other datasets can be found in Appendix A.

Boundary-enriching Effects. Figure 4(b) shows UMAP projections of Citeseer. Numeric interpolations remain inside the original boundary, while LLM-generated samples extend beyond, enlarging the VRM decision region.

We next report BCR and BPS (Figure 4c; Appendix A) under three settings: Original, Interp+Orig, and Interp-Only. Original data achieves the highest scores, confirming real data's proximity to decision boundaries. Importantly, LLM interpolation consistently surpasses numeric interpolation, better aligning with the original distribution while enriching boundaries.

For class consistency, we trained balanced MLP classifiers. On Citeseer (Figure 4d), the classifier reached 71.3 ± 0.7 accuracy. Numeric interpolations yield 100% In-Class Rate (ICR), while LLM interpolations achieve ICRs of 81%, 69%, 94%, and 100%, reflecting boundary-near samples occasionally shifting to neighboring classes. Additional datasets are reported in Appendix A.

4.3 Necessity of Confidence-based Edge Assignment (RQ2)

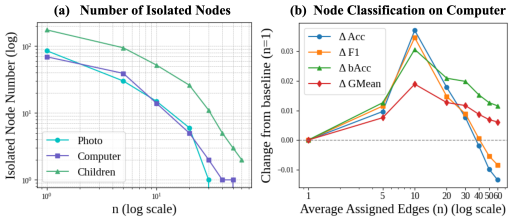


Figure 5: Top-k edge selection.

Isolating Noisy Samples. We instantiate the confidence function with a *top–k* rule, where each synthetic node retains $k = \#augmented\_nodes \times n$ highest-confidence edges. Varying n, Figure 5a shows isolated nodes (log-scaled) shrink as $\log(n)$ increases, while Figure 5b yields a bell-shaped accuracy curve (relative to n=1), confirming our hypothesis: With small k, many synthetic nodes become singletons (Theorem 3.6), leaving minority manifolds underpopulated. In contrast, large k induces excessive connectivity, leading to over-smoothing [26].

Effectiveness of Confidence Function. We compare edge duplication vs. confidence-based assignment ('C'). Table 2 shows that confidence-based assignment consistently improves performance under LLM interpolation. Similar gains hold for numeric interpolation, except on Cora and Photo, where off-manifold samples are better isolated by our method.

4.4 Enhancement in Long-tailed Graph Learning (RQ3)

Table 5 shows that our method, particularly SAVE-TAG$_S$, outperforms prior baselines in node classification on long-tailed text-attributed graphs.

Method	Cora				Pubmed				Citeseer			
	Acc	F1	GMean	bAcc	Acc	F1	GMean	bAcc	Acc	F1	GMean	bAcc
SMOTE	71.8±2.7	70.8±2.1	82.7±1.7	71.8±2.7	68.5±6.2	65.5±10.1	75.9±4.8	68.5±6.2	50.7±3.0	47.9±3.1	67.6±2.2	50.7±3.0
Oversampling	72.9±1.3	72.0±0.9	83.4±0.9	72.9±1.3	58.5±6.3	50.4±6.5	68.1±5.1	58.5±6.3	52.0±4.9	49.6±4.7	68.5±3.6	52.0±4.9
EmbedSMOTE	70.0±1.6	69.8±1.4	81.5±1.0	70.0±1.6	63.4±7.6	60.3±9.7	72.0±5.9	63.4±7.6	47.5±2.0	45.4±2.5	65.2±1.5	47.5±2.0
GraphSMOTE$_T$	74.4±4.0	**74.2±4.3**	84.4±2.6	74.4±4.0	67.9±5.0	67.8±5.0	75.5±3.9	67.9±5.0	50.5±2.8	48.1±3.2	67.4±2.1	50.5±2.8
GraphSMOTE$_O$	74.3±3.7	74.1±4.9	84.3±3.0	74.3±4.7	69.7±5.0	69.9±4.8	76.9±3.9	69.7±5.0	52.0±1.6	49.8±1.8	68.6±1.2	52.0±1.6
MixupForGraph	66.4±2.7	63.32±2.7	80.1±1.6	68.0±2.5	69.6±0.8	68.7±1.1	75.9±1.1	68.7±1.6	63.2±1.9	59.4±1.9	75.1±1.8	60.9±2.5
HierTail *	72.8±5.3	73.3±4.7	83.3±3.4	72.8±5.3	72.4±8.2	72.4±7.7	79.0±6.3	72.5±8.0	49.6±9.6	44.5±11.3	66.5±7.4	49.6±9.6
LTE4G *	74.2±3.8	74.1±3.8	84.3±2.4	74.2±3.8	72.9±1.5	72.6±1.9	79.4±1.2	72.9±1.5	51.4±3.8	49.2±3.7	68.1±2.8	51.4±3.8
SAVE-TAG$_O$	72.1±0.1	70.9±0.1	84.5±0.3	74.8±0.4	72.8±0.3	73.0±0.4	80.2±0.2	75.0±0.2	69.9±0.2	66.8±0.2	79.9±0.1	67.8±0.2
SAVE-TAG$_M$	73.0±0.4	71.8±0.3	85.1±0.2	75.8±0.4	71.3±0.2	71.2±0.3	79.1±0.1	73.6±0.3	69.5±0.3	65.9±0.4	79.3±0.3	66.9±0.5
SAVE-TAG$_S$	**76.4±0.3**	74.1±0.3	**85.8±0.2**	**76.6±0.3**	**76.3±0.7**	**76.4±0.8**	**82.5±0.5**	**77.6±0.7**	**69.9±0.3**	**66.9±0.3**	**80.1±0.3**	**68.1±0.4**

Table 1: Comparison of SAVE-TAG with previous long-tail graph learning baselines.

4.5 Ablation Studies

We conduct ablations (Table 2) showing that LLM interpolation outperforms numeric, and incorporating the confidence function (with subscript $_C$) further improves node classification (see Section 4.3).

	Method	Cora		PubMed		Citeseer		Photo		Computer		Children	
		F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc
SAVE-TAG$_S$	Origin	70.40	72.46	47.79	55.09	49.63	53.71	54.99	51.69	44.93	49.34	2.25	3.00
	Num	70.59	73.11	69.06	69.43	62.69	65.89	57.29	55.74	48.90	54.47	15.87	16.01
	Num$_C$	70.09	72.44	69.19	69.63	65.19	68.79	56.05	54.97	49.31	54.69	19.32	**18.87**
	LLM	72.54	74.51	72.54	72.50	66.51	69.19	58.40	56.75	52.81	60.17	17.94	17.82
	LLM$_C$	**74.05**	**76.36**	**76.43**	**76.35**	**66.85**	**69.89**	**59.20**	**58.65**	**56.89**	**66.26**	**19.69**	18.64
SAVE-TAG$_O$	Origin	70.40	72.46	47.79	55.09	49.63	53.71	54.99	51.69	44.93	49.34	2.25	3.00
	Num	71.04	73.80	69.44	69.77	63.12	66.28	57.08	55.08	49.47	55.78	15.75	15.80
	Num$_C$	70.89	72.76	69.01	69.43	64.75	68.24	54.44	51.31	48.85	54.57	18.13	17.50
	LLM	70.09	72.86	71.44	71.51	**66.99**	69.58	59.32	58.19	52.45	59.45	18.06	18.31
	LLM$_C$	**73.21**	**75.32**	**72.96**	**72.80**	66.82	**69.90**	**63.62**	**66.17**	**57.03**	**65.12**	**21.81**	**22.99**
SAVE-TAG$_M$	Origin	70.40	72.46	47.79	55.09	49.63	53.71	54.99	51.69	44.93	49.34	2.25	3.00
	Num	**73.21**	**75.32**	70.48	70.73	63.52	67.24	57.92	56.04	49.47	53.85	15.93	16.68
	Num$_C$	71.86	71.18	72.66	72.46	64.10	68.89	56.74	54.57	53.78	60.03	19.07	18.10
	LLM	72.30	74.21	71.34	71.38	66.93	69.69	**59.53**	58.06	52.45	55.77	**18.82**	**19.98**
	LLM$_C$	70.54	72.51	**76.18**	**76.26**	**67.96**	**71.60**	59.45	**58.60**	**57.83**	**66.98**	**22.16**	**24.50**

Table 2: **Ablation Studies.** Node classification under three variants of SAVE-TAG using GCN. *Origin* uses no augmentation; *Num* and *LLM* apply numeric and LLM-based interpolation. Subscript $_C$ denotes confidence-based edge generation; lack of subscript implies naive edge duplication.

5 Related Works

Here we outline relevant concepts and advances; see Appendix D for details.

- *Vicinal Risk Minimization (VRM):* VRM [27] extends ERM [5] by estimating risk from local vicinity distributions, enhancing model generalization. It underlies methods like Mixup [28] and SMOTE [7], which create synthetic samples via interpolation.
- *Long-tailed Graph Learning:* In graph learning, long-tailed node classification is typically addressed via hierarchical task grouping like HierTail [2] and expert models like LTE4G [3].
- *Large Language Model (LLM) Data Augmentation:* Generally, LLMs augment data by generating contextually relevant text via zero-shot or few-shot instruction-based prompts [10, 29].

6 Conclusion

This paper presents a novel interpolation framework for long-tailed node classification that unifies semantic and structural signals. With LLMs, we generate manifold-consistent, boundary-enriching samples to extend VRM to the textual domain. To mitigate out-of-distribution generation, we introduce a confidence-based edge assignment that filters synthetic nodes via graph connectivity. Theoretically and empirically, our method outperforms numeric interpolation, standard LLM augmentation, and prior long-tailed graph learning across benchmarks.

*Modified code while running on PubMed dataset to prevent out-of-memory issues due to scalability.

References

[1] Zhengyang Mao, Wei Ju, Siyu Yi, Yifan Wang, Zhiping Xiao, Qingqing Long, Nan Yin, Xinwang Liu, and Ming Zhang. Learning knowledge-diverse experts for long-tailed graph classification. *ACM Trans. Knowl. Discov. Data*, 19(2), January 2025. ISSN 1556-4681. doi: 10.1145/3705323. URL https://doi.org/10.1145/3705323.

[2] Haohui Wang, Baoyu Jing, Kaize Ding, Yada Zhu, Wei Cheng, Si Zhang, Yonghui Fan, Liqing Zhang, and Dawei Zhou. Mastering long-tail complexity on graphs: Characterization, learning, and generalization. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '24, page 3045–3056, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400704901. doi: 10.1145/3637528.3671880. URL https://doi.org/10.1145/3637528.3671880.

[3] Sukwon Yun, Kibum Kim, Kanghoon Yoon, and Chanyoung Park. Lte4g: Long-tail experts for graph neural networks. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, CIKM '22, page 2434–2443, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450392365. doi: 10.1145/3511808.3557381. URL https://doi.org/10.1145/3511808.3557381.

[4] Olivier Chapelle, Jason Weston, Léon Bottou, and Vladimir Vapnik. Vicinal risk minimization. *Advances in neural information processing systems*, 13, 2000.

[5] Vladimir Vapnik and Vlamimir Vapnik. Statistical learning theory wiley. *New York*, 1(624):2, 1998.

[6] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

[7] Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16: 321–357, 2002.

[8] Tianxiang Zhao, Xiang Zhang, and Suhang Wang. Graphsmote: Imbalanced node classification on graphs with graph neural networks. In *Proceedings of the 14th ACM international conference on web search and data mining*, pages 833–841, 2021.

[9] Yiwei Wang, Wei Wang, Yuxuan Liang, Yujun Cai, and Bryan Hooi. Mixup for node and graph classification. In *Proceedings of the Web Conference 2021*, pages 3663–3674, 2021.

[10] Zhuoyan Li, Hangxiao Zhu, Zhuoran Lu, and Ming Yin. Synthetic data generation with large language models for text classification: Potential and limitations. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 10443–10461, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.647. URL https://aclanthology.org/2023.emnlp-main.647/.

[11] Jianxiang Yu, Yuxiang Ren, Chenghua Gong, Jiaqi Tan, Xiang Li, and Xuecang Zhang. Leveraging large language models for node generation in few-shot learning on text-attributed graphs, 2024. URL https://arxiv.org/abs/2310.09872.

[12] Hui Han, Wen-Yuan Wang, and Bing-Huan Mao. Borderline-smote: a new over-sampling method in imbalanced data sets learning. In *Proceedings of the 2005 International Conference on Advances in Intelligent Computing - Volume Part I*, ICIC'05, page 878–887, Berlin, Heidelberg, 2005. Springer-Verlag. ISBN 3540282262. doi: 10.1007/11538059_91. URL https://doi.org/10.1007/11538059_91.

[13] Junsoo Oh and Chulhee Yun. Provable benefit of mixup for finding optimal decision boundaries. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org, 2023.

[14] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks, 2017. URL https://arxiv.org/abs/1609.02907.

[15] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks, 2018. URL https://arxiv.org/abs/1710.10903.

[16] Qimai Li, Zhichao Han, and Xiao-Ming Wu. Deeper insights into graph convolutional networks for semi-supervised learning, 2018. URL https://arxiv.org/abs/1801.07606.

[17] Fernando Gama, Joan Bruna, and Alejandro Ribeiro. Stability properties of graph neural networks. *IEEE Transactions on Signal Processing*, 68:5680–5695, 2020. ISSN 1941-0476. doi: 10.1109/tsp.2020.3026980. URL http://dx.doi.org/10.1109/TSP.2020.3026980.

[18] Matthieu Cordonnier, Nicolas Keriven, Nicolas Tremblay, and Samuel Vaiter. Convergence of message passing graph neural networks with generic aggregation on large random graphs, 2025. URL https://arxiv.org/abs/2304.11140.

[19] Xingcheng Fu, Yuecen Wei, Qingyun Sun, Haonan Yuan, Jia Wu, Hao Peng, and Jianxin Li. Hyperbolic geometric graph representation learning for hierarchy-imbalance node classification. In *Proceedings of the ACM Web Conference 2023*, WWW '23, page 460–468. ACM, April 2023. doi: 10.1145/3543507.3583403. URL http://dx.doi.org/10.1145/3543507.3583403.

[20] Mengting Zhou and Zhiguo Gong. Graphsr: A data augmentation algorithm for imbalanced node classification, 2023. URL https://arxiv.org/abs/2302.12814.

[21] Andrew Kachites McCallum, Kamal Nigam, Jason Rennie, and Kristie Seymore. Automating the construction of internet portals with machine learning. *Information Retrieval*, 3:127–163, 2000.

[22] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. Collective classification in network data. *AI magazine*, 29(3):93–93, 2008.

[23] Hao Yan, Chaozhuo Li, Ruosong Long, Chao Yan, Jianan Zhao, Wenwen Zhuang, Jun Yin, Peiyan Zhang, Weihao Han, Hao Sun, et al. A comprehensive study on text-attributed graphs: Benchmarking and rethinking. *Advances in Neural Information Processing Systems*, 36:17238–17264, 2023.

[24] Nitesh V Chawla. C4. 5 and imbalanced data sets: investigating the effect of sampling method, probabilistic estimate, and decision tree structure. In *Proceedings of the ICML*, volume 3, page 66. CIBC Toronto, ON, Canada, 2003.

[25] Shin Ando and Chun Yuan Huang. Deep over-sampling framework for classifying imbalanced data. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2017, Skopje, Macedonia, September 18–22, 2017, Proceedings, Part I 10*, pages 770–785. Springer, 2017.

[26] Jintang Li, Wangbin Sun, Ruofan Wu, Yuchang Zhu, Liang Chen, and Zibin Zheng. Over-smoothing: A nightmare for graph contrastive learning?, 2024. URL https://arxiv.org/abs/2306.02117.

[27] Olivier Chapelle, Jason Weston, Léon Bottou, and Vladimir Vapnik. Vicinal risk minimization. In *Proceedings of the 14th International Conference on Neural Information Processing Systems*, NIPS'00, page 395–401, Cambridge, MA, USA, 2000. MIT Press.

[28] Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization, 2018. URL https://arxiv.org/abs/1710.09412.

[29] Yichuan Li, Kaize Ding, Jianling Wang, and Kyumin Lee. Empowering large language models for textual data augmentation, 2024. URL https://arxiv.org/abs/2404.17642.

[30] Heinrich Jiang, Been Kim, Melody Y. Guan, and Maya Gupta. To trust or not to trust a classifier, 2018. URL https://arxiv.org/abs/1805.11783.

[31] Matthias Fey and Jan Eric Lenssen. Fast graph representation learning with pytorch geometric, 2019. URL https://arxiv.org/abs/1903.02428.

[32] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*, 2019.

[33] Zhikai Chen, Haitao Mao, Hang Li, Wei Jin, Hongzhi Wen, Xiaochi Wei, Shuaiqiang Wang, Dawei Yin, Wenqi Fan, Hui Liu, et al. Exploring the potential of large language models (llms) in learning on graphs. *ACM SIGKDD Explorations Newsletter*, 25(2):42–61, 2024.

[34] Zhikai Chen, Haitao Mao, Jingzhe Liu, Yu Song, Bingheng Li, Wei Jin, Bahare Fatemi, Anton Tsitsulin, Bryan Perozzi, Hui Liu, and Jiliang Tang. Text-space graph foundation models: Comprehensive benchmarks and new insights, 2024. URL `https://arxiv.org/abs/2406.10727`.

[35] Geoffrey Grimmett and David Stirzaker. *Probability and random processes*. Oxford university press, 2020.

[36] Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

Contents of Appendix

A Additional experimental results

A.1 LLM as Vicinal Risk Minimization.

We evaluate the effectiveness of LLM-based interpolation within the Vicinal Risk Minimization (VRM) framework. As shown in Table 3, interpolation-based prompting significantly enhances the quality of synthesized textual data, resulting in improved performance on downstream node classification tasks.

Method	Cora				Pubmed				Citeseer			
	Acc	F1	GMean	bAcc	Acc	F1	GMean	bAcc	Acc	F1	GMean	bAcc
Original	72.5±1.4	70.4±1.1	84.5±0.6	74.7±0.9	55.1±8.2	47.8±15.0	62.2±7.9	51.5±9.7	53.7±0.1	49.6±2.2	71.4±0.9	55.9±1.3
Zero-shot	68.3±0.5	63.3±0.7	81.1±0.4	69.4±0.7	71.4±0.4	71.4±0.5	78.3±0.4	72.3±0.5	64.8±0.5	62.1±0.5	78.0±0.3	65.2±0.5
Few-shots	73.9±0.6	**72.7±0.6**	85.4±0.2	76.1±0.3	72.0±0.3	72.0±0.3	79.2±0.2	73.6±0.3	68.2±0.9	65.6±0.7	79.8±0.4	67.8±0.5
LLM_O	73.7±0.7	72.1±0.7	85.3±0.3	76.1±0.4	71.5±0.3	71.4±0.3	78.8±0.3	73.1±0.4	**69.6±0.8**	**67.0±0.6**	**80.5±0.4**	**68.8±0.5**
LLM_M	74.2±0.4	72.6±0.3	**85.6±0.1**	**76.5±0.2**	71.9±0.4	72.0±0.4	79.0±0.2	73.4±0.3	69.5±0.6	66.8±0.4	80.3±0.2	68.6±0.3
LLM_S	**74.5±0.4**	72.5±0.5	85.5±0.2	76.3±0.3	**72.5±0.4**	**72.5±0.4**	**79.4±0.3**	**73.8±0.3**	69.2±0.6	66.5±0.5	80.1±0.3	68.3±0.5

Table 3: Comparison of VRM-based LLM interpolation with standard LLM augmentation. Subscripts $_O$, $_M$, and $_S$ denote the prompting strategies of their corresponding SAVE-TAG variants, but without edge assignment. Best and second-best results are emboldened and underlined, respectively.

A.2 Semantic-aware Benefits.

UMAP Visualization. Figure 6 shows the UMAP visualization across all six datasets, with the boundary of the original data circled. As we can see, for all the datasets, numeric interpolated samples tend to stay within the original boundary, while LLM-generated samples often extend beyond it, enlarging the VRM decision region.

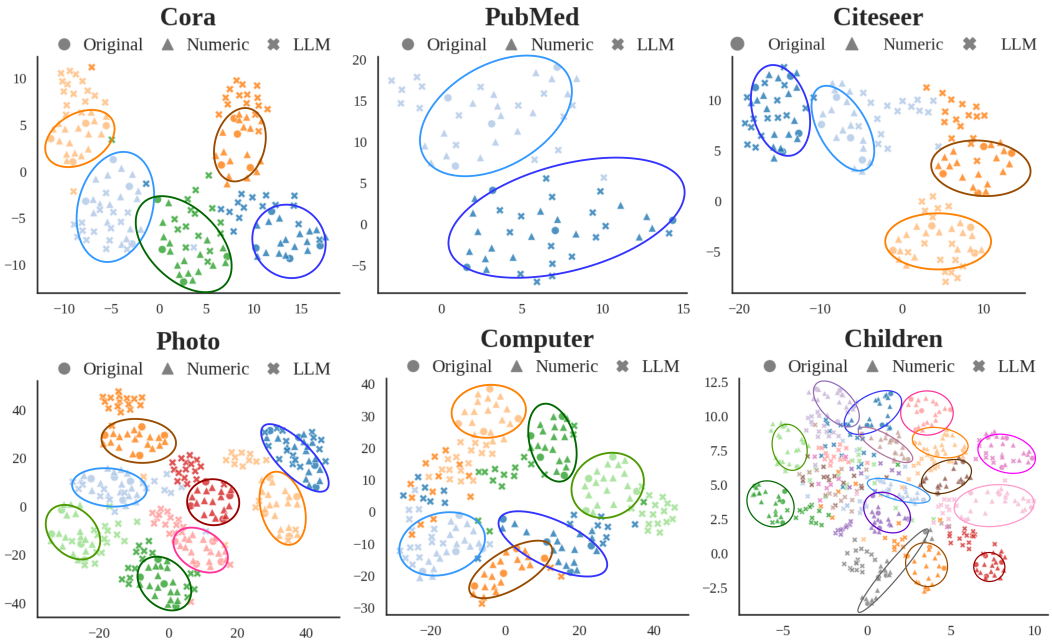


Figure 6: UMAP projection of the original and interpolated samples across various benchmarks—the original data boundary is outlined, with numeric samples mostly staying within and LLM-generated samples extending beyond.

Boundary Coverage Rate (BCR) and Boundary Proximity Score (BPS). To further assess boundary-enriching effects, we report the Boundary Coverage Ratio (BCR), as defined in Definition 3.2, and the Boundary Proximity Score (BPS). BPS is computed as the reciprocal of the Trust Score [30], measuring the ratio between the distance to the nearest out-of-class centroid and the distance to the in-class centroid. Figure 7 presents the evaluation results using BCR and BPS metrics

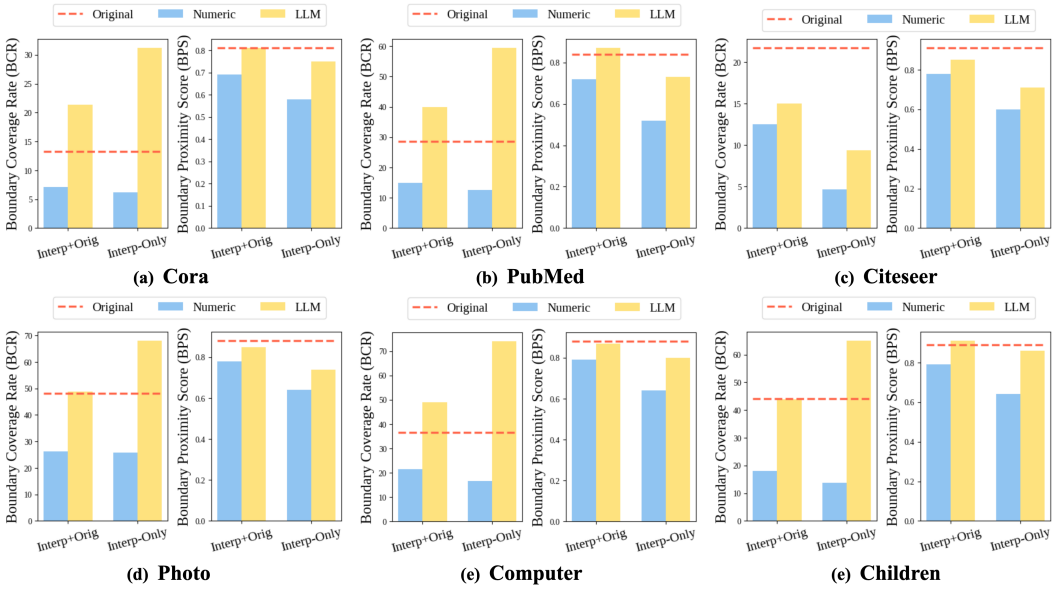


Figure 7: Boundary Coverage Rate (BCR) and Boundary Proximity Score (BPS) for numeric (blue) vs. LLM (yellow) interpolation in Interp+Orig and Interp-Only settings across six datasets; the red dashed line marks the original baseline.

across all datasets. We report performance under three settings: original data, the combination of original and interpolated samples (Interp+Orig), and interpolated samples alone (Interp-Only).

Across all benchmarks, our LLM-based interpolation consistently outperforms numeric interpolation on both metrics, suggesting that LLM-generated samples align more effectively with decision boundaries. Particularly in some cases, LLM-based interpolation even exceeds the original data in the BCR, highlighting its potential to extend and enrich the original decision boundary.

In-Class Rate (ICR). In addition to BCR and BPS, we report the In-Class Rate (ICR), defined as the percentage of samples predicted with the same label as their ground truth. Implementation details for these metrics are provided in Section 4.2.We train a separate MLP classifier on each dataset using a balanced set of class samples to assess class consistency. The average In-Class Rate (ICR)—the proportion of augmented samples classified into their intended class—is reported in Figure 8.

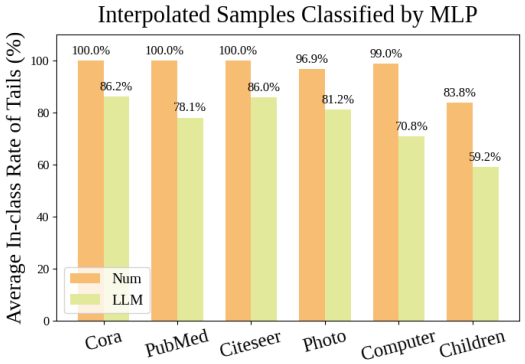


Figure 8: Node Classification Improvement.

Numerical interpolation achieves 100% ICR on Cora, PubMed, and Citeseer, and nearly 100% on others, reflecting high class consistency. In contrast, LLM-based samples show lower ICRs, indicating proximity to decision boundaries that causes occasional misclassification by MLP.

Node classification Performance We evaluate the impact of LLM-based and numeric interpolation on node classification across multiple benchmarks. Results are presented in Figure 9. *Embed* utilizes LLaMA 3.2-1B embeddings for interpolation, while *SimCSE* and *SBERT* serve as embedding-based numeric baselines. Across all datasets, the our LLM-based approach with LLaMA 3.2-1B consistently outperforms all numeric interpolation baselines.

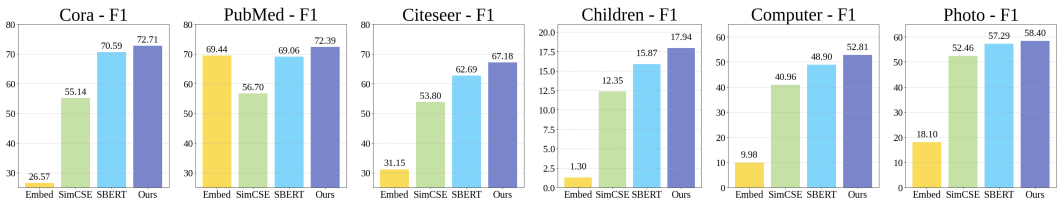
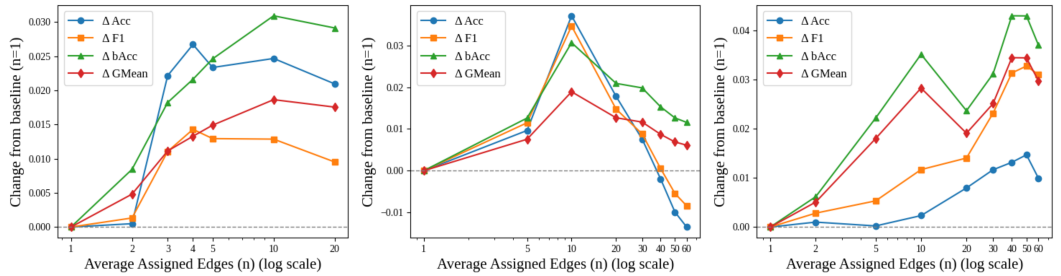


Figure 9: Node classification performance (F1) under various implementation of SAVE-TAG$_S$. *Embed* uses Llama3.2-1B embeddings for interpolation. *SimCSE* and *SBERT* are compared as alternative embedding baselines. Our LLM-based method with Llama3.2-1B consistently outperforms.



(a) Photo **(c) Computer** **(b) Children**

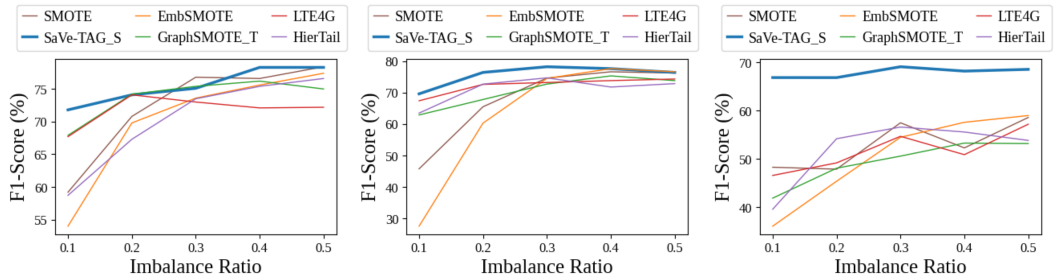
Figure 10: Impact of average assigned edges on performance for Photo (a), Children (b), and Computer (c). We report changes from the baseline (n=1) across four metrics: ΔAcc, ΔF1, ΔbAcc, and ΔGMean.

A.3 Necessity of Topology-aware Filters

We instantiate Definition **??** using a *top–k* rule, where each synthetic node retains its k highest-confidence edges. The value of k is set to $\#augmented\_nodes \times n$, and we vary n to study its effect on node isolation and classification performance. Figure 10 reveals a bell-shaped performance curve with respect to $\log(n)$ on Photo, Computer, and Children, measured relative to n=1. More discussion can be found in Section 4.3.

A.4 Enhancement in Long-tailed Graph Learning.

Stability. We also adjusted the imbalance ratio, selecting from $\{0.1, 0.2, 0.3, 0.4, 0.5\}$, to examine how our model performs under different levels of imbalance. As shown in Figure 11, SAVE-TAG$_S$ (the bold line) exhibits less performance drop as the imbalance ratio decreases, demonstrating less fluctuation compared to other baselines.



(b) Cora **(b) PubMed** **(b) Citeseer**

Figure 11: F1 score (%) versus imbalance ratio on (a) Cora, (b) PubMed, and (c) Citeseer, comparing other baselines of long-tailed graph learning and with our SAVE-TAG$_S$ (the bold line).

Classifier Variants. To assess the generalization of our method across different classifier architectures, we conduct additional ablation studies using one GNN-based backbone (SAGE) and one non-graph-based model (MLP). Since MLP does not incorporate structural information (i.e., it lacks

neighborhood aggregation), the confidence-based edge assignment has no effect. Therefore, we report only the results without the subscript $_C$. As shown in Table 4, SAVE-TAG effectively addresses long-tailed classification with both GNNs and non-graph-based models. Its strong performance with MLP highlights its potential for text classification and broader applications.

	Model	Method	Cora		PubMed		Citeseer		Photo		Computer		Children	
			F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc
SAVE-TAG$_S$	MLP	Origin	50.31	52.34	25.51	42.59	61.70	64.64	4.25	7.48	3.09	4.61	1.06	2.03
		Num	58.28	61.35	64.63	68.62	61.70	64.64	34.96	34.74	28.19	31.77	15.31	14.85
		LLM	**64.13**	**66.80**	**72.77**	**72.07**	**64.04**	**66.25**	**43.26**	**40.79**	**37.65**	**40.26**	**16.64**	**16.83**
	SAGE	Origin	67.5	70.2	60.7	62.4	52.9	55.5	47.4	44.2	36.5	38.0	2.5	4.0
		Num	70.6	73.1	68.3	68.6	65.0	68.0	54.2	52.1	43.1	47.3	18.7	18.6
		Num$_C$	70.3	72.4	69.2	69.6	65.8	69.4	50.6	48.9	49.3	54.7	19.4	18.7
		LLM	72.5	74.5	72.5	72.5	**67.4**	**70.0**	**57.7**	56.0	48.6	53.7	20.1	**20.8**
		LLM$_C$	**74.0**	**76.4**	**76.4**	**76.3**	66.6	69.3	57.6	**56.1**	**52.0**	**59.1**	**20.3**	20.3
SAVE-TAG$_O$	MLP	Origin	50.31	52.34	25.51	42.59	61.70	64.64	4.25	7.48	3.09	4.61	1.06	2.03
		Num	57.85	60.97	66.06	66.46	60.23	63.38	35.47	35.01	28.65	32.32	15.89	15.32
		LLM	**63.02**	**64.12**	**70.25**	**69.88**	**64.68**	**67.07**	**43.34**	**42.15**	**38.18**	**42.74**	**18.65**	**20.38**
	SAGE	Origin	67.5	70.2	60.7	62.4	52.9	55.5	47.4	44.2	36.5	38.0	2.5	4.0
		Num	70.6	73.8	69.0	69.3	65.3	68.6	54.6	53.5	43.0	47.1	18.5	19.0
		Num$_C$	69.1	70.9	68.3	68.6	65.5	68.9	49.0	46.1	42.9	46.0	18.6	19.1
		LLM	**72.1**	**73.7**	71.4	71.6	**67.5**	**70.3**	57.8	56.1	48.1	53.9	20.9	21.8
		LLM$_C$	69.2	70.1	**72.5**	**72.3**	66.2	69.0	**58.0**	**57.3**	**52.5**	**59.9**	**22.5**	**23.8**
SAVE-TAG$_M$	MLP	Origin	50.31	52.34	25.51	42.59	61.70	64.64	4.25	7.48	3.09	4.61	1.06	2.03
		Num	61.66	62.89	67.88	67.77	61.06	63.57	36.98	35.39	29.25	33.24	15.95	15.95
		LLM	**62.94**	**64.47**	**71.12**	**70.56**	**64.52**	**66.64**	**44.84**	**42.28**	**38.47**	**41.47**	**18.80**	**20.09**
	SAGE	Origin	67.5	70.2	60.7	62.4	52.9	55.5	47.4	44.2	36.5	38.0	2.5	4.0
		Num	71.9	74.0	69.8	70.0	64.8	68.7	56.0	54.1	46.0	51.4	19.4	19.8
		Num$_C$	69.8	70.1	69.9	69.7	64.3	68.6	53.6	51.7	46.6	51.4	19.2	19.6
		LLM	**72.3**	**74.2**	71.5	71.3	**68.0**	70.8	58.4	56.7	49.1	54.8	21.7	23.1
		LLM$_C$	70.5	72.5	**75.7**	**75.2**	67.8	**70.9**	**58.6**	**57.8**	**50.6**	**57.4**	**23.9**	**26.5**

Table 4: **Ablation Studies.** Node classification performance using SAGE and MLP under three variants of SAVE-TAG. *Origin* denotes no augmentation; *Num* and *LLM* apply numeric and LLM-based interpolation, respectively. The subscript $_C$ indicates the use of confidence-based edge generation, while its absence denotes naive edge duplication. The best results for both MLP and SAGE are emboldened, and the second-best results for SAGE are underlined.

Method	Cora				Pubmed				Citeseer			
	Acc	F1	GMean	bAcc	Acc	F1	GMean	bAcc	Acc	F1	GMean	bAcc
SMOTE	71.8±2.7	70.8±2.1	82.7±1.7	71.8±2.7	68.5±6.2	65.5±10.1	75.9±4.8	68.5±6.2	50.7±3.0	47.9±3.1	67.6±2.2	50.7±3.0
Oversampling	72.9±1.3	72.0±0.9	83.4±0.9	72.9±1.3	58.5±6.3	50.4±6.5	68.1±5.1	58.5±6.3	52.0±4.9	49.6±4.7	68.5±3.6	52.0±4.9
EmbedSMOTE	70.0±1.6	69.8±1.4	81.5±1.0	70.0±1.6	63.4±7.6	60.3±9.7	72.0±5.9	63.4±7.6	47.5±2.0	45.4±2.5	65.2±1.5	47.5±2.0
GraphSMOTE$_T$	74.4±4.0	**74.2±4.3**	84.4±2.6	74.4±4.0	67.9±5.0	67.8±5.0	75.5±3.9	67.9±5.0	50.5±2.8	48.1±3.2	67.4±2.1	50.5±2.8
GraphSMOTE$_O$	74.3±3.7	74.1±4.9	84.3±3.0	74.3±4.7	69.7±5.0	69.9±4.8	76.9±3.9	69.7±5.0	52.0±1.6	49.8±1.8	68.6±1.2	52.0±1.6
MixupForGraph	66.4±2.7	63.32±2.7	80.1±1.6	68.0±2.5	69.6±0.8	68.7±1.1	75.9±1.1	68.7±1.6	63.2±1.9	59.4±1.9	75.1±1.8	60.9±2.5
HierTail *	72.8±5.3	73.3±4.7	83.3±3.4	72.8±5.3	72.4±8.2	72.4±7.7	79.0±6.3	72.5±8.0	49.6±9.6	44.5±11.3	66.5±7.4	49.6±9.6
LTE4G *	74.2±3.8	74.1±3.8	84.3±2.4	74.2±3.8	72.9±1.5	72.6±1.9	79.4±1.2	72.9±1.5	51.4±3.8	49.2±3.7	68.1±2.8	51.4±3.8
SAVE-TAG$_O$	72.1±0.1	70.9±0.1	84.5±0.3	74.8±0.4	72.8±0.3	73.0±0.4	80.2±0.2	75.0±0.2	69.9±0.2	66.8±0.2	79.9±0.1	67.8±0.2
SAVE-TAG$_M$	73.0±0.4	71.8±0.3	85.1±0.2	75.8±0.4	71.3±0.2	71.2±0.3	79.1±0.1	73.6±0.3	69.5±0.3	65.9±0.4	79.3±0.3	66.9±0.5
SAVE-TAG$_S$	**76.4±0.3**	74.1±0.3	**85.8±0.2**	**76.6±0.3**	**76.3±0.7**	**76.4±0.8**	**82.5±0.5**	**77.6±0.7**	**69.9±0.3**	**66.9±0.3**	**80.1±0.3**	**68.1±0.4**

Table 5: Comparison of SAVE-TAG with previous long-tail graph learning baselines.

B Implementation Details

B.1 Code Implementation

Algorithm Our codes are provided in the following link:
`https://shorturl.at/zUlDQ`
The pseudocode of SAVE-TAG is provided in Algorithm 1.

Computing Environment and Resources We implement our model under PyG [31] and Sentence Transformer [32] modules. Experiments are conducted on a NVIDIA GeForce RTX 3090 and the OS was Ubuntu 22.04.4 LTS with 128GB RAM.

Algorithm 1: SAVE-TAG

Input: Attributed graph $\mathcal{G} = (\mathcal{V}, \mathcal{T}, \mathcal{E})$ with texts $\mathcal{X}$ and labels $\mathcal{Y}$;
vicinal order k; temperature–controlled LLM $\mathcal{L}_\tau$;
frozen encoder ϕ; confidence network κ_ω;
edge budget k_{edge}; confidence threshold τ

Output: Augmented graph $\mathcal{G}' = (\mathcal{V}', \mathcal{T}', \mathcal{E}')$; trained classifier f_θ

Initialize $\bar{\mathcal{T}}$, $\bar{\mathcal{V}}$, $\bar{\mathcal{E}}$ as empty sets
Identify tail classes: $\mathcal{C}_t \leftarrow \{c \mid \text{freq}(c) < \text{median freq}\}$
$\mathcal{V}_t \leftarrow \{v \in \mathcal{V} \mid y_v \in \mathcal{C}_t\}$
Function `DataAugment(`v, u`)`**:**
    $\hat{t} \leftarrow \mathcal{L}_\tau\big(t_v, t_u \,\big|\, y_v\big)$
    Initialize new node $\hat{v}$ with text $\hat{t}$ and label $y_{\hat{v}} = y_v$
    return $\hat{v}$

`// Generate vicinal twins and interpolate with the LLM`
for $v \in \mathcal{V}_t$ **do**
    $\mathcal{N}_k^{(\text{tail})}(v) \leftarrow \{u \in \mathcal{V}_t \mid u \in \mathcal{N}_k(v)\}$
    for $u \in \mathcal{N}_k^{(\text{tail})}(v)$ **do**
        $\hat{v} \leftarrow$ `DataAugment(`v, u`)`
        Append $\hat{v}$ to $\bar{\mathcal{V}}$, its text $\hat{t}$ to $\bar{\mathcal{T}}$

`// Pre-train confidence function on original graph`
$\kappa_\omega \leftarrow \text{train}(\{z_v = \phi(t_v)\}_{v \in \mathcal{V}})$

`// Edge assignment guided by confidence`
for $\hat{v} \in \bar{\mathcal{V}}$ **do**
    $z_{\hat{v}} \leftarrow \phi(\hat{t}_{\hat{v}})$
    $s_u \leftarrow \kappa_\omega(z_u)\langle z_{\hat{v}}, z_u\rangle$ **for** $u \in \mathcal{V}$
    $\mathcal{N}_{k_{\text{edge}}}(\hat{v}) \leftarrow \arg\text{top}_{k_{\text{edge}}} s_u$
    if $\max_u s_u \geq \tau$ **then**
        $\bar{\mathcal{E}} \leftarrow \bar{\mathcal{E}} \cup \{(\hat{v}, u) \mid u \in \mathcal{N}_{k_{\text{edge}}}(\hat{v})\}$
    else
        pass;

Form augmented graph:
$\mathcal{V}' \leftarrow \mathcal{V} \cup \bar{\mathcal{V}}$,
$\mathcal{T}' \leftarrow \mathcal{T} \cup \bar{\mathcal{T}}$,
$\mathcal{E}' \leftarrow \mathcal{E} \cup \bar{\mathcal{E}}$
Train GNN f_θ on $\mathcal{G}'$
Result: $\mathcal{G}'$, f_θ

B.2 Hyperparameters

The detailed hyperparameters of our setups for reproducibility are listed below:

- **Text Encoding:** We utilize the Sentence Transformer (`all-MiniLM-L6-v2`) [32], for its efficiency in semantic comprehension. This model strikes an excellent balance between being lightweight and delivering strong performance. A comparison with other embeddings is provided in Figure 9.
- **Node Classification:** By default, we use GCN as our model, consisting of 2 hidden layers with 64 neurons each. The dropout rate is 0.5 and the model is trained for 1000 epochs with a learning rate of 0.01 (same configuration as we switch to SAGE when conducting experiments in Table 4).
- **Confidence Function:** We employ MLP models composed of 1 hidden layer with 256 neurons and set with a dropout rate of 0, which are trained for 1000 epochs with a learning rate of 0.001.
- **Text Generation**: By default, we leverage pre-trained Llama3-8B-Instruct from Meta for text generation, configured with bfloat16 and default parameters.

- **Data Augmentation**: Due to the limited number of training nodes in our low-labeled, imbalanced settings, we selected $k = 3$ for the k-nearest neighbors in our LLM-based methods.
- **Edge generation**: For small datesets (Cora, PubMed, and Citeseer), we set $k = |\mathcal{V}| \times 20$ and add the top k highest prediction score edges to the graph. For larger datasets (Photo, Computer, and Children), we increase k to $|\mathcal{V}| \times 40$ correspondingly.

B.3 Datasets

In this part, we include brief descriptions of each dataset. To realistically simulate long-tailed distribution scenarios typical in real-world graphs, we follow the standard protocol from prior work [2, 3, 8, 9, 19, 20]. Specifically, we randomly select 20 nodes from each majority (head) class, and $20 \times$ *imbalance_ratio* nodes from each minority class. For Cora and PubMed, we downloaded the preprocessed datasets from `https://github.com/CurryTang/Graph-LLM` [33]. As to Citeseer, Photo, Children, and Computer datasets, the preprocessed graph data are downloaded from `https://github.com/CurryTang/TSGFM` [34] and their detailed descriptions can be found in `https://github.com/sktsherlock/TAG-Benchmark` [23]. The category names for each dataset are listed below, with their indices corresponding to the numeric labels in the dataset.

- **Cora**: [‘Rule Learning’, ‘Neural Networks’, ‘Case Based’, ‘Genetic Algorithms’, ‘Theory’, ‘Reinforcement Learning’, ‘Probabilistic Methods’]
- **PubMed**: [‘Diabetes Mellitus, Experimental’, ‘Diabetes Mellitus Type 1’, ‘Diabetes Mellitus Type 2’]
- **Citeseer**:[‘Agents’, ‘ML(Machine Learning)’, ‘IR(Information retrieval)’, ’DB(Database)’, ‘HCI (Human-computer Interaction)’, ‘AI(Artificial Intelligence)’]
- **Photo**:[‘Video Surveillance’,"Accessories’,‘Binoculars & Scopes’, ‘Video’, ‘Lighting & Studio’, ‘Bags & Cases’,‘Tripods & Monopods’, ‘Flashes’, ‘Digital Cameras’, ‘Film Photography’, ‘Lenses’, ‘Underwater Photography’]
- **Children**: [‘Literature & Fiction’, ‘Animals’, ‘Growing Up & Facts of Life’, ‘Humor’, ‘Cars Trains & Things That Go’, ‘Fairy Tales Folk Tales & Myths’, ‘Activities Crafts & Games’, ‘Science Fiction & Fantasy’, ‘Classics’, ‘Mysteries & Detectives’, ‘Action & Adventure’, ‘Geography & Cultures’, ‘Education & Reference’, ‘Arts Music & Photography’, ‘Holidays & Celebrations’, ‘Science Nature & How It Works’, ‘Early Learning’, ‘Biographies’, ‘History’, “Children’s Cookbooks", ‘Religions’, ‘Sports & Outdoors’, ’Comics & Graphic Novels’, ’Computers & Technology’]
- **Computer**: [‘Computer Accessories & Peripherals’, ‘Tablet Accessories’, ‘Laptop Accessories’, ‘Computers & Tablets’, ‘Computer Components’, ‘Data Storage’, ‘Networking Products’, ‘Monitors’, ‘Servers’, ‘Tablet Replacement Part’]

B.4 Baselines Details

LLM-based Data Augmentation Baselines.

- *Zero-shot* [10, 11]: Utilizing a pre-trained LLM to generate textual attributes for a given target label without providing specific examples.
- *Few-shot* [10, 11]: Prompting an LLM to generate textual attributes for a given target label, providing examples for in-context learning.

Long-tailed Graph Learning Baselines.

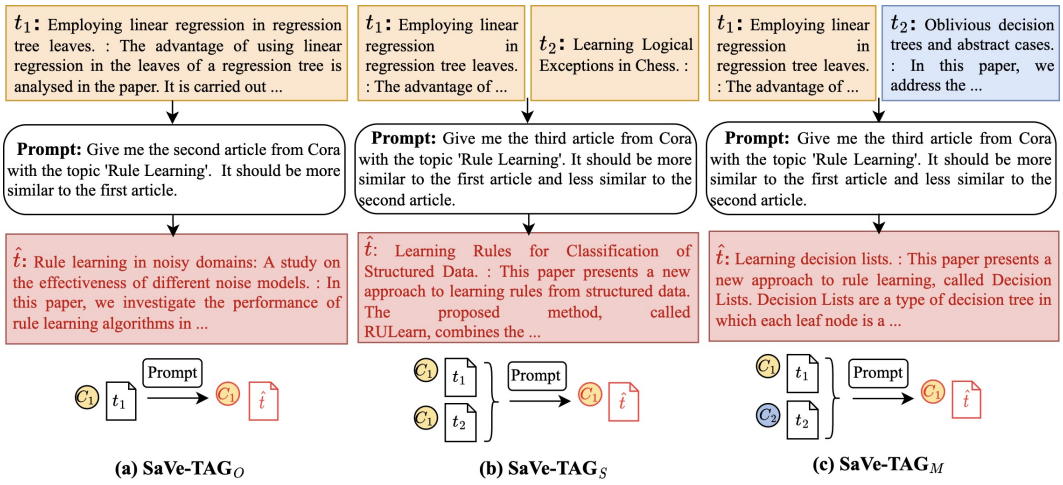
- *Oversampling* [24]: Repeatedly duplicates node embeddings that belong to the minority classes.
- *SMOTE* [7]: Generating synthetic nodes by interpolating tail nodes with neighbors and assigning edges by copying their neighbors’ edges.
- *Embed-SMOTE* [25]: Applying SMOTE on hidden layer embeddings rather than the input features.
- *MixupForGraph* [9]: Synthesizing additional data for minority classes by interpolating randomly paired hidden representation as well as their labels

- *GraphSMOTE* [8]: Interpolating minority nodes with their nearest neighbors and generating new edges using a co-trained link predictor, with two variants denoted by subscript $_T$ and $_O$, depending on whether the predicted edges are discrete or continuous.
- *LTE4G* [3]: Clustering nodes by class and degree, training subset-specific experts, and distilling them into head and tail students for classification.
- *HierTail* [2]: Extracting shared class information via hierarchical task grouping and balancing head–tail gradient contributions.

  To comprehensively evaluate our method, we design and implement three variants of SAVE-TAG, each corresponding to a specific VRM-based strategy. Detailed prompting strategies for each variant are provided in the Appendix B.

  – **SAVE-TAG$_O$:** Mirrors the oversampling method but generate a texts similar to the existing one rather than simple duplication from the conventional method.
  – **SAVE-TAG$_S$:** Inspired by SMOTE [7], interpolates with nearest neighbors within the same class.
  – **SAVE-TAG$_M$:** Extends SMOTE [7] by interpolating embeddings with nearest neighbors from potentially different classes, inspired by Mixup techniques.

SAVE-TAG Variants. A case study illustration is provided in Figure 12.



(a) SaVe-TAG$_O$ **(b) SaVe-TAG$_S$** **(c) SaVe-TAG$_M$**

Figure 12: A case study illustrating three variants our LLM-based interpolation: (a) SAVE-TAG$_O$, (b) SAVE-TAG$_S$, and (c) SAVE-TAG$_M$.

System:	You are a helpful AI assistant for generating `{Task}` from `{Dataset}`, where each `{Text}` follows the format `<START>` `{Format}` `<END>`.
User:	Give me the first `{Text}` from `{Dataset}` with topic $[\mathcal{C}_1]$.
Assistant:	`<START>` $[t_1]$ `<END>`.
User:	Give me the second `{Text}` from `{Dataset}` with topic $[\mathcal{C}_2]$.
Assistant:	`<START>` $[t_2]$ `<END>`.
User:	Give me the third `{Text}` from `{Dataset}` with topic $[\mathcal{C}_2]$. It should be more similar to the first `{Text}` and less similar to the second `{Text}`.
Assistant:	...

Table 6: Prompt template for SAVE-TAG$_S$ and SAVE-TAG$_M$ ($\mathcal{C}_1 = \mathcal{C}_2$ for SAVE-TAG$_O$).

System:	You are a helpful AI assistant for generating `{Task}` from `{Dataset}`, where each `{Task}` follows the format `<START>` `{Format}` `<END>`.
User:	Give me the first `{Task}` from `{Dataset}` with topic $[\mathcal{C}_1]$.
Assistant:	`<START>` $[t_1]$ `<END>`.
User:	Give me the second `{Task}` from `{Dataset}` with topic $[\mathcal{C}_1]$. It should be more similar to the first `{Task}`.
Assistant:	...

Table 7: Prompt template for SAVE-TAG$_O$ across all datasets.

`{Dataset}`	`{Task}`	`{Text}`	`{Format}`
Cora	'new academic articles'	'article'	'[New Title] : [New Abstract]'\n'
Pubmed	'new academic articles'	'article'	'Title: [New Title]\n Abstract: [New Abstract]'
Citeseer	'new academic articles'	'article'	'[New Title] : [New Abstract]'\n'
Photo	'reviews of products from Amazon'	'review'	'Review: [New Review]'
Computer	'reviews of products from Amazon'	'review'	'Review: [New Review]'
Children	'new book descriptions'	'book description'	'Title: [New Title]\n Book Description: [New Description]'

Table 8: Input parameters for different datasets in the prompt templates.

B.5 Prompt Design

The prompt template for SAVE-TAG$_O$ is shown in Table 7, while SAVE-TAG$_S$ and SAVE-TAG$_M$ share a common template in Table 6, with the constraint of shared class ($\mathcal{C}_1 = \mathcal{C}_2$) when using SAVE-TAG$_S$. Both templates are dataset-specific, with their parameters detailed in Table 8.

C Efficiency and Scalability Analysis

Name	# Nodes	# Edges	# Class	# Tail	Ave Text Len	# Batch	Prompts/Batch	Time/Prompt
Cora	2,708	10,858	7	5	891	20	4	3.06s
PubMed	19,717	88,670	3	2	1649	8	4	4.51s
Citeseer	3186	8450	6	4	1022	16	4	3.50s
Photo	48,362	500,939	12	8	804	64	2	4.60s
Computer	87,229	721,081	10	6	499	24	2	4.09s
Children	76,875	1,554,578	24	15	1255	240	1	7.21s

Table 9: **Dataset Details.** We recorded the following statistics on all datasets: the number of nodes, edges, and classes; the number of tail (minority) classes; the average text length of node attributes; the total number of batches; the number of prompts per batch (batch size); and the average time per prompt (generated entry) during generation.

C.1 Time Costs

For scalability analysis, we report the statistics regarding the size of the graph (number of edges, nodes, average length of text attributes, etc.) in the Table 9.

To facilitate text generation, we generate samples in batches. While larger batches reduce overall runtime, memory constraints limit us to small batch sizes. We then compute the average LLM inference time per generated sample. The data regarding batch size and inference time per prompt are also reported in Table 9.

C.2 Adaptability for Small Language Models

Figure 13 shows the performance and efficiency of SAVE-TAG$_S$ using language models with different sizes, including Llama-3-8B-Instruct, Llama-3.2-1B-Instruct, and Llama-3.2-3B-Instruct. As expected, smaller models require less inference time for text generation. Notably, the F1 score remains relatively stable across model sizes, highlighting the adaptability of our method to lightweight, open-source LLMs—a promising direction for reducing generation costs.

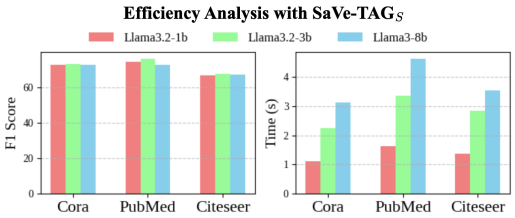


Figure 13: The F1 score and generation time per entry of LLM variants across different datasets implementing SAVE-TAG$_S$.

D Related Works

Vicinal Risk Minimization (VRM) VRM [27] is a principle in statistical learning theory that extends the traditional Empirical Risk Minimization (ERM) framework [5]. Introduced by Chapelle et al., VRM proposes estimating the expected risk by considering a vicinity distribution around each training sample, rather than relying solely on the empirical distribution. This approach aims to improve generalization by incorporating local neighborhoods of data points into the learning process. VRM has been foundational in developing techniques like Mixup [6] and SMOTE [7], which generate synthetic training examples through linear interpolations of existing data points, thereby enhancing model generalization capabilities.

- Mixup [6]: Given $(x_i, y_i), (x_j, y_j)$, draw $\lambda \sim \mathrm{Beta}(\alpha, \alpha)$ and set
$$\tilde{x} = \lambda x_i + (1-\lambda)x_j, \quad \tilde{y} = \lambda y_i + (1-\lambda)y_j.$$
- SMOTE [7]: For minority sample x_i, choose a neighbor x_k, draw $\lambda \sim U(0,1)$ and define
$$\tilde{x} = x_i + \lambda\,(x_k - x_i), \quad \tilde{y} = y_i = y_j.$$

Long-tail Graph Learning In many real-world graph-structured datasets, the distribution of node classes follows a long-tailed pattern, where a few classes (head classes) have abundant samples, while many others (tail classes) have scarce representations. This imbalance poses significant challenges for graph neural networks (GNNs), which may become biased towards

- *Hierarchical Task Grouping:* Formulating long-tailed classification as a multi-task learning problem, where each class prediction is treated as a separate task. This approach allows for shared representations among related tasks, improving performance on tail classes [2].
- *Expert Models:* Training specialized models (experts) for different subsets of classes or nodes, such as the LTE4G [3], which considers both class and degree long-tailedness in graphs.

LLM Data Augmentation Large Language Models (LLMs) have revolutionized data augmentation strategies, especially in natural language processing tasks. By generating synthetic data that mirrors the distribution and characteristics of real data, LLMs can enhance model training in low-resource settings. Key approaches include:

- *Instruction-based Augmentation:* Empowering LLMs with task-specific instructions to generate diverse and contextually relevant data [29].
- *Zero-shot or Few-shot Augmentation:* Enabling LLMs to generate synthetic examples with minimal or no task-specific labels via in-context learning[10].

E Ethics and Broader Impact

SAVE-TAG LLMs to generate synthetic text for augmenting long-tailed node classification on text-attributed graphs. By mitigating class imbalance, it improves classification performance on widely used benchmark datasets such as Cora, Citeseer, and PubMed. The use of lightweight, open-source LLMs and one-time data augmentation helps reduce computational cost, potentially enabling broader access to graph learning techniques in low-resource settings.

Our study is conducted entirely on publicly available, anonymous datasets, and does not involve any personally identifiable or sensitive information. The generated synthetic data is stored in files and used exclusively for research purposes. While our method does not raise significant ethical concerns in its current scope, we encourage practitioners to clearly label synthetic data in downstream applications to avoid confusion between real and generated content.

F Detailed Proof

F.1 Proof of Theorem 3.1 (Manifold-Preserving Class-Consistent Generation)

Notation recap. Fix a class label $c \in \{1, \dots, C\}$.

- $p_c := p(\cdot \mid y{=}c)$ is the true class-conditional distribution over texts $t \in \mathcal{T}$.
- $q_c := \mathcal{L}_\tau(\cdot \mid c)$ is the LLM's conditional sampling distribution.
- $\phi : \mathcal{T} \to \mathbb{R}^{d'}$ is the frozen encoder mapping a text to its representation.
- $\mathcal{M}_c := \{\phi(t) : y{=}c\}$ is the class-c representation manifold.
- $A := \phi^{-1}(\mathcal{M}_c) = \{\, t \in \mathcal{T} : \phi(t) \in \mathcal{M}_c \,\}$ is the on-manifold event in text space.
- $D_{\mathrm{TV}}(p, q) := \sup_B |p(B) - q(B)|$ denotes total-variation distance between probability measures p, q on $\mathcal{T}$.

Assumptions for this theorem.

A1. *True on-manifold mass.* $p_c(A) \geq 1 - \delta$ for some $\delta \in [0, 1)$.

A2. *LLM approximation in TV.* $D_{\mathrm{TV}}(p_c, q_c) \leq \epsilon$ for some $\epsilon \in [0, 1)$.

A3. *Measurability.* A is measurable so that $p_c(A)$ and $q_c(A)$ are well defined.[†]

Proof. By Assumption **A2**. and the definition of total variation,

$$q_c(A) \;\geq\; p_c(A) - D_{\mathrm{TV}}(p_c, q_c).$$

Applying Assumption **A1**. yields

$$q_c(A) \;\geq\; (1 - \delta) - \epsilon \;=\; 1 - (\delta + \epsilon).$$

By the definition of A and q_c, $q_c(A) = \Pr_{\hat{t} \sim \mathcal{L}_\tau(\cdot|c)}\big[\phi(\hat{t}) \in \mathcal{M}_c\big]$, which proves

$$\Pr\big[\phi(\hat{t}) \in \mathcal{M}_c\big] \;\geq\; 1 - (\delta + \epsilon).$$

F.2 Proof of Theorem 3.3 (Margin lower bound)

Notation recap. $\gamma(x)$ is the logit margin of a point x, $\gamma_{\min}(S) = \min_{x \in S} \gamma(x)$, and $\gamma_0 := \gamma_{\min}(\mathcal{D})$ is the smallest margin on the *original* training set $\mathcal{D}$. After augmentation, we have $\tilde{\mathcal{D}} = \mathcal{D} \cup \widehat{\mathcal{D}}_{\mathrm{bd}} \cup \widehat{\mathcal{D}}_{\mathrm{in}}$, where $\widehat{\mathcal{D}}_{\mathrm{bd}}$ (resp. $\widehat{\mathcal{D}}_{\mathrm{in}}$) holds all synthetic *boundary* (resp. *interior*) samples and $\mathrm{BCR} = \frac{|\widehat{\mathcal{D}}_{\mathrm{bd}}|}{|\widehat{\mathcal{D}}_{\mathrm{bd}}| + |\widehat{\mathcal{D}}_{\mathrm{in}}|} \in [0, 1]$.

Assumptions for this theorem.

A1. *Margin monotonicity.* Retraining on $\tilde{\mathcal{D}}$ does not decrease any margin that was already $\geq \gamma_0$. Thus, every point in $\mathcal{D} \cup \widehat{\mathcal{D}}_{\mathrm{in}}$ keeps margin $\geq \gamma_0$ under the new model f_θ^{aug}.

A2. *Boundary–sample slack.* For every boundary sample $\hat{x}$ we have the (possibly loose) bound $|\gamma(\hat{x})| \leq \delta$ with a fixed constant $\delta > 0$.

A3. *Conservative ordering.* We choose $\delta \geq \gamma_0$. This turns **A2**. into a *trivial worst-case cap* (it can never be violated) and lets the algebra below produce a bona-fide lower bound—that is, the right-hand side never exceeds the true minimum margin.

[†]This holds automatically when ϕ is measurable and $\mathcal{M}_c$ is Borel.

Two exhaustive cases. Let $m_{\mathrm{bd}} := |\widehat{\mathcal{D}}_{\mathrm{bd}}|$ and $m_{\mathrm{in}} := |\widehat{\mathcal{D}}_{\mathrm{in}}|$.

Case 1: $\boldsymbol{m_{\mathrm{bd}} = 0}$ *(BCR* $= 0$*).* The augmented set contains only $\mathcal{D} \cup \widehat{\mathcal{D}}_{\mathrm{in}}$. By Assumption **A1**. every margin is still $\geq \gamma_0$, hence $\gamma_{\min}(\tilde{\mathcal{D}}) \geq \gamma_0$. The bound in the statement reduces to $\gamma_0 - \delta(1-0) = \gamma_0$, so the inequality is tight in this corner case.

Case 2: $\boldsymbol{m_{\mathrm{bd}} > 0}$ *(BCR* > 0*).* Now at least one boundary sample is present. By **A2**. its margin is $\geq -\delta$ and, because $\delta \geq \gamma_0$ (**A3**.), also $\geq \gamma_0 - \delta$. Combining with Assumption **A1**. for all other points gives

$$\gamma_{\min}(\tilde{\mathcal{D}}) \;\geq\; \min\{\gamma_0,\; \gamma_0 - \delta\} \;=\; \gamma_0 - \delta.$$

Because BCR$\in (0, 1]$, $\gamma_0 - \delta(1 - \mathrm{BCR}) \;\leq\; \gamma_0 - \delta$, and the claimed inequality again holds.

Unifying the two cases. Observe that in **Case 1** $(1 - \mathrm{BCR}) = 1$ while in **Case 2** $(1 - \mathrm{BCR}) < 1$. Hence the single formula

$$\boxed{\gamma_{\min}(\tilde{\mathcal{D}}) \;\;\geq\;\; \gamma_0 - \delta\big(1 - \mathrm{BCR}\big)}$$

is simultaneously valid for both scenarios, completing the proof.

F.3 Proof of Theorem 3.1 (On-Manifold Vicinal Risk Theorem)

Notation recap. For each original training pair (t_i, c_i):

- $\mathcal{V}_{\mathcal{L}_\tau}(t_i)$ is the *vicinal distribution* produced by LLM-based interpolation.
- $\hat{t} \sim \mathcal{V}_{\mathcal{L}_\tau}(t_i)$ is one synthetic neighbor.
- $\phi : \mathcal{T} \to \mathbb{R}^{d'}$ is the frozen encoder; $\mathcal{M}_{c_i} = \{\phi(t) : y = c_i\}$ is the class-c_i manifold.
- $\mathbf{L}(\cdot, \cdot)$ is the loss (cross-entropy, hinge, ...).
- f_θ is the classifier that we are analyzing.

Assumptions for this theorem.

A1. *Manifold-preservation.* For every i and any neighbor $\hat{t} \sim \mathcal{V}_{\mathcal{L}_\tau}(t_i)$,

$$\Pr\big[\phi(\hat{t}) \notin \mathcal{M}_{c_i}\big] \;\leq\; \delta,$$

where $\delta \in (0, 1)$ is the constant proved in Theorem 3.1.

A2. *Uniform bound.* The same δ works for *all* training indices i (if they differ, use the worst-case $\delta = \max_i \delta_i$).

A3. *Finite loss.* $\mathbf{L}(f_\theta(\phi(\hat{t})), c_i)$ is integrable so that all expectations are well defined.[‡]

Vicinal risk decomposition. The VRM objective is

$$R_{vrm}(f_\theta) = \frac{1}{n} \sum_{i=1}^{n} \mathbb{E}_{\hat{t} \sim \mathcal{V}_{\mathcal{L}_\tau}(t_i)} \big[\mathbf{L}(f_\theta(\phi(\hat{t})), c_i)\big].$$

Fix i temporarily. Apply **A1.** and the law of total expectation [35]:

$$\mathbb{E}_{\hat{t}}\big[\mathbf{L}(\cdot)\big] = (1 - \delta)\, \mathbb{E}\big[\mathbf{L}(\cdot) \mid \phi(\hat{t}) \in \mathcal{M}_{c_i}\big] + \;\delta\, \mathbb{E}\big[\mathbf{L}(\cdot) \mid \phi(\hat{t}) \notin \mathcal{M}_{c_i}\big].$$

Define

$$R_{\text{on}} := \frac{1}{n} \sum_{i=1}^{n} \mathbb{E}\big[\mathbf{L}(\cdot) \mid \phi(\hat{t}) \in \mathcal{M}_{c_i}\big], \qquad R_{\text{off}} := \frac{1}{n} \sum_{i=1}^{n} \mathbb{E}\big[\mathbf{L}(\cdot) \mid \phi(\hat{t}) \notin \mathcal{M}_{c_i}\big].$$

Summing the displayed equality over $i = 1, \ldots, n$ and using **A2.** yields

$$\boxed{R_{vrm}(f_\theta) = (1 - \delta)\, R_{\text{on}} + \delta\, R_{\text{off}}}$$

with $0 \leq \delta \ll 1$ coming from Theorem 3.1. Because $(1 - \delta)$ is the probability of drawing an *on-manifold* neighbor, the VRM risk is evaluated on-manifold with probability at least $1 - \delta$, completing the proof.

[‡]A bounded loss (e.g. cross-entropy with label smoothing) or a sub-Gaussian surrogate satisfies this automatically.

F.4 Proof of Theorem 3.4 (Boundary-Coverage ⇒ Vicinal-Risk Reduction)

Notation recap.

- $\tilde{\mathcal{D}} = \mathcal{D} \cup \widehat{\mathcal{D}}$ is the augmented training set; $\widehat{\mathcal{D}}$ contains m synthetic points.
- $\widehat{\mathcal{D}}_{\mathrm{bd}}$ / $\widehat{\mathcal{D}}_{\mathrm{in}}$ synthetic *boundary* / *interior* subsets; $\mathrm{BCR} := |\widehat{\mathcal{D}}_{\mathrm{bd}}|/m$.
- $\gamma(\hat{x})$ logit margin of $\hat{x}$; $\gamma_0 := \min_{x \in \mathcal{D}} \gamma(x)$.
- δ upper bound on $|\gamma(\hat{x})|$ for $\hat{x} \in \widehat{\mathcal{D}}_{\mathrm{bd}}$ (Theorem 3.3).
- $\mathcal{J}$ supervised loss and L is the Lipschitz constant of $\mathbf{L}$ in its first (logit) argument.
- f_θ^{orig} model trained on $\mathcal{D}$; f_θ^{aug} model after VRM training on $\tilde{\mathcal{D}}$.

Assumptions used in the proof.

A1. *Lipschitz loss.* $\mathcal{J}(z_1, y) - \mathcal{J}(z_2, y) \le L\|z_1 - z_2\|_2$ [36].

A2. *Non-decreasing margins.* VRM retraining never *reduces* the margin of a synthetic point: $\gamma_{\mathrm{aug}}(\hat{x}) \ge \gamma_{\mathrm{orig}}(\hat{x})$.

A3. *Boundary slack.* $|\gamma(\hat{x})| \le \delta$ for all $\hat{x} \in \widehat{\mathcal{D}}_{\mathrm{bd}}$.

A4. *Interior margin.* $\gamma(\hat{x}) \ge \gamma_0$ for all $\hat{x} \in \widehat{\mathcal{D}}_{\mathrm{in}}$.

(Last three assumptions are justified in Appendix F.2.)

Pointwise loss drop. For any synthetic $(\hat{x}, \hat{y})$, combine **A1. + A2.** to obtain

$$\mathcal{J}\big(f_\theta^{\mathrm{aug}}(\hat{x}), \hat{y}\big) - \mathcal{J}\big(f_\theta^{\mathrm{orig}}(\hat{x}), \hat{y}\big) \;\le\; -L\,\gamma(\hat{x}). \tag{S1}$$

Separate boundary vs. interior sums. Sum (S1) over the two synthetic subsets and divide by m:

$$\begin{aligned} R_{vrm}\big(f_\theta^{\mathrm{aug}}\big) - R_{vrm}\big(f_\theta^{\mathrm{orig}}\big) &\le -\frac{L}{m} \sum_{\hat{x} \in \widehat{\mathcal{D}}_{\mathrm{bd}}} \gamma(\hat{x}) - \frac{L}{m} \sum_{\hat{x} \in \widehat{\mathcal{D}}_{\mathrm{in}}} \gamma(\hat{x}) \\ &\overset{\text{A3.,A4.}}{\le} -L\,\delta\,\frac{|\widehat{\mathcal{D}}_{\mathrm{bd}}|}{m} \;-\; L\,\gamma_0\,\frac{|\widehat{\mathcal{D}}_{\mathrm{in}}|}{m}. \end{aligned}$$

Recognize the fractions as BCR and $1 - \mathrm{BCR}$:

$$R_{vrm}\big(f_\theta^{\mathrm{aug}}\big) - R_{vrm}\big(f_\theta^{\mathrm{orig}}\big) \;\le\; -L\,\gamma_0\,\mathrm{BCR} \;+\; L\,\delta\,(1 - \mathrm{BCR}). \tag{S2}$$

Express the bound in compact form. Choose $\eta = \delta/\gamma_0$ $(0 < \eta < 1)$ and rewrite (S2) as

$$R_{vrm}\big(f_\theta^{\mathrm{aug}}\big) - R_{vrm}\big(f_\theta^{\mathrm{orig}}\big) \;\le\; -L\,\gamma_0\,\big[\mathrm{BCR} - \eta\big] \;+\; \mathcal{O}(\delta),$$

where the $\mathcal{O}(\delta)$ term hides constants independent of BCR when δ is small. Hence whenever $\mathrm{BCR} > \eta = \delta/\gamma_0$ (the typical case in our experiments), the vicinal risk strictly *decreases* after VRM training.

F.5 Proof of Theorem 3.5 (Pulling Well-Aligned Nodes)

Notation.

- $h_v^{(\ell)}$ – representation of node v after ℓ GNN layers.
- $\mathcal{M}_c$ – representation manifold of class c in $\mathbb{R}^p$.
- $d_\ell := \mathrm{dist}\big(h_{\hat{v}}^{(\ell)}, \mathcal{M}_c\big)$ – Euclidean distance of synthetic node $\hat{v}$ to $\mathcal{M}_c$.
- $\mathbf{W}$ – linear map inside the GNN layer; $\alpha \in (0, 1)$, $\beta_{\hat{v}u} \ge 0$ with $\sum_{u \in \mathcal{N}'(\hat{v})} \beta_{\hat{v}u} = 1$.

Layer-wise update. For node $\hat{v}$ the generic message-passing rule (Equation 1) reads

$$h_{\hat{v}}^{(\ell+1)} = \alpha\,\mathbf{W} h_{\hat{v}}^{(\ell)} + (1 - \alpha) \sum_{u \in \mathcal{N}'(\hat{v})} \beta_{\hat{v}u}\,\mathbf{W} h_u^{(\ell)}. \tag{U}$$

Assumptions.

A1. *Non-expansive layer:* spectral norm $\|\mathbf{W}\|_2 \le L < 1$.

A2. *Well-aligned neighbors:* $\operatorname{dist}\big(h_u^{(\ell)}, \mathcal{M}_c\big) \le \varepsilon$ for every $u \in \mathcal{N}'(\hat{v})$.

A3. *Convexity of weights:* $\alpha \in (0,1)$ and $\{\beta_{\hat{v}u}\}$ form a convex combination.

(All three conditions are satisfied by common GNNs and our confidence-based edge assignment.)

Contractive bounds after applying $\mathbf{W}$. By **A1.**, linear mapping $\mathbf{W}$ shrinks distances by at most factor L:

$$\operatorname{dist}\big(\mathbf{W}h_{\hat{v}}^{(\ell)}, \mathbf{W}\mathcal{M}_c\big) \le L\, d_\ell, \qquad \operatorname{dist}\big(\mathbf{W}h_u^{(\ell)}, \mathbf{W}\mathcal{M}_c\big) \le L\,\varepsilon \text{ (by \textbf{A2.})}. \tag{B}$$

Distance after aggregation. Because the right-hand side of (U) is a convex combination (**A3.**), the triangle inequality and (B) give

$$d_{\ell+1} := \operatorname{dist}\big(h_{\hat{v}}^{(\ell+1)}, \mathcal{M}_c\big) \;\le\; \alpha L d_\ell \;+\; (1-\alpha)L\varepsilon. \tag{C}$$

Geometric contraction (when $d_\ell \ge \varepsilon$). If the current distance satisfies $d_\ell \ge \varepsilon$, then $(1-\alpha)L\varepsilon \le (1-\alpha)Ld_\ell$, so (C) becomes

$$d_{\ell+1} \le \underbrace{\big[\alpha L + (1-\alpha)L\big]}_{=\,\alpha L}\, d_\ell =: \gamma\, d_\ell,$$

with

$$\gamma := \alpha L \quad \text{and} \quad 0 < L < 1,\ 0 < \alpha < 1 \implies \gamma \in (0,1).$$

Exponential convergence. Iterate the inequality whenever $d_k \ge \varepsilon$:

$$d_\ell \;\le\; \gamma^{\ell-\ell_0}\, d_{\ell_0} \quad (\ell \ge \ell_0),$$

demonstrating exponential decay toward $\mathcal{M}_c$. Once the distance falls below ε, neighbors and self-term are already within the same ε-tube, so all subsequent distances stay bounded by $L\varepsilon$.

G Notation Table

Table 10: Summary of notation used throughout the paper.

Symbol	Meaning
$\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{T})$	Attributed graph: nodes $\mathcal{V}$, edges $\mathcal{E}$, texts $\mathcal{T}$
$\mathcal{G}' = (\mathcal{V}', \mathcal{E}', \mathcal{T}')$	Augmented graph: augmented nodes $\mathcal{V}'$, augmented edges $\mathcal{E}'$, augmented texts $\mathcal{T}'$
$n = \lvert\mathcal{V}\rvert$	Number of nodes
$\mathcal{X} \in \mathbb{R}^{n \times d}$	Node feature matrix (d features per node)
$\mathcal{Y} = \{y_v\}$	Node labels, $y_v \in \{1, \ldots, C\}$
$\mathcal{P}$	Empirical distribution on $(\mathcal{X}, \mathcal{Y})$
f_θ	Node-level classifier (e.g. GNN)
$\phi : \mathcal{T} \to \mathbb{R}^{d'}$	Frozen text encoder; $z_v = \phi(t_v)$
$\mathcal{M}_c$	Representation manifold of class c; $\mathcal{M} = \bigcup_c \mathcal{M}_c$
$\mathcal{C}_t$	Set of tail (minority) classes
$\mathcal{N}_k(v)$	k-hop / k-NN neighborhood of node v
$\mathcal{N}_k^{(\text{tail})}(v)$	Vicinal twins of v (tail nodes in $\mathcal{N}_k(v)$)
$\mathcal{L}_\tau$	Temperature-controlled LLM generator (temperature τ)
$\hat{t}_{ij}$	Class-consistent text generated from vicinal twins (v_i, v_j)
$\bar{\mathcal{V}}, \bar{\mathcal{T}}$	Sets of synthetic nodes and their texts
κ_ω	Confidence function $\kappa : [0, 1]$ for edge assignment
$\mathcal{N}_k(\hat{v})$	Top-k neighbors chosen for synthetic node $\hat{v}$
α, β_{vu}	Self-weight and neighbor weights in GNN aggregator
$h_v^{(\ell)}$	Hidden representation of v at layer ℓ
$\mathbf{W}$	Linear transformation inside a GNN layer
$\gamma(x)$	Logit margin of sample x
γ_0	Minimum margin on original training set
BCR	Boundary-Coverage Rate (Def. 3.2)
R_{vrm}	Vicinal risk under VRM
L	Lipschitz constant of the loss $\mathbf{L}$
δ, η	Constants in margin / risk bounds