UNICG: A FRAMEWORK FOR UNIFYING GRAPH COMPREHENSION AND GENERATION

Anonymous authors
Paper under double-blind review

ABSTRACT

The pursuit of general-purpose intelligence has motivated unified architectures that both comprehend and generate content, with notable progress in language and vision. However, prevailing graph learning paradigms still treat these capabilities separately: comprehension models map graphs to predictions, whereas generative models synthesize graphs without being evaluated for comprehension. Consequently, enabling a single model to comprehend and generate graphs through a shared representation remains a major challenge. In this paper, we propose **UNICG**, a novel framework for Unifying graph Comprehension and Generation that addresses both the representational incompatibility and optimization interference between these capabilities. Specifically, to establish a common graph interface, UNICG combines lossless graph serialization with residual quantization of node semantics to construct a shared discrete graph vocabulary. To support both capabilities within one model, UNICG formulates generation as next-token prediction and comprehension as a pooled read-out over a single autoregressive backbone. To alleviate the asymmetric interference exposed by joint training, UNICG routes the two objectives through task-decoupled experts while retaining the shared backbone. Extensive experiments across 20 graph domains, four comprehension tasks, and three generation regimes against 13 baselines demonstrate that UNICG jointly supports both capabilities, achieving competitive comprehension performance while generating valid and novel graphs.

1 INTRODUCTION

The development of general-purpose models has increasingly brought comprehension and generation into a single architecture. In language and vision, unified models can interpret inputs and synthesize new content through a common modeling interface (Team et al., 2024; Wu et al., 2025). Graph learning has not achieved a comparable integration. Methods for graph comprehension infer node labels, edge relations, and graph properties (Liu et al., 2023; Wang et al., 2024; He et al., 2025; Sun et al., 2023), whereas graph generators learn distributions over structures and attributes (Vignac et al., 2023; Chen et al., 2025; Qin et al., 2025). These two families are developed with different representations, learning objectives, and evaluation settings. As a result, how to enable one model to comprehend and generate graphs remains a fundamental challenge.

We study this challenge as **unified graph comprehension and generation**, where a single model performs both task families over the same graph corpora and both capabilities are evaluated together. This joint perspective is important because the two capabilities provide complementary evidence about the learned representation. Comprehension evaluates whether the representation supports predictions about nodes, edges, and entire graphs. Generation further requires the model to produce complete graph instances whose validity, novelty, distributional fidelity, and controllability can be assessed. A generator, in turn, does not demonstrate that its internal representation supports predictive tasks. Evaluating both capabilities therefore provides a more complete account of the structural and semantic knowledge captured by a general-purpose graph model.

Research Question: *Can a single model comprehend and generate graphs over the same corpora through a shared representation?*

Two obstacles make this problem technically nontrivial. First, graph comprehension commonly maps an unordered graph to continuous node or graph embeddings, often followed by a non-invertible

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

pooling operation. Graph generation instead requires a complete and decodable description from which nodes, edges, and attributes can be recovered. Some comprehension methods quantize local structures or node representations to improve transfer (Wang et al., 2024; Yang et al., 2024), but their codes are optimized for predictive representation learning rather than complete graph reconstruction. Second, generative likelihood and supervised prediction impose different demands on shared parameters, allowing naive joint optimization to create asymmetric interference between the two objectives. Unified graph comprehension and generation therefore requires both a common representation designed for the two capabilities and an architecture that shares transferable knowledge while preserving their distinct optimization requirements.

In this paper, we propose **UniCG**, the first framework for **Uni**fying Graph **C**omprehension and **G**eneration within one shared multi-domain model over the same corpora. UniCG addresses the representational incompatibility and optimization interference through three coordinated designs. ❶ It constructs a **shared invertible graph vocabulary** by combining canonical structural serialization with residual quantization of node semantics, producing token sequences that are informative for prediction and decodable for generation. ❷ It formulates generation as next-token prediction and comprehension as a pooled read-out over **one autoregressive backbone**, allowing both capabilities to learn from the same graph description. ❸ It introduces **task-decoupled experts** that isolate objective-specific updates while retaining the shared backbone, thereby reducing the asymmetric interference exposed by naive joint training. We evaluate both capabilities under *Uni-Seen* and *Uni-Unseen* adaptation settings, which differ in whether the target domain participates in pre-training. We also examine few-shot comprehension to assess adaptation when target labels are limited. The contributions of this work are as follows:

- We formulate unified graph comprehension and generation and propose **UniCG**, the first framework that jointly supports and evaluates both capabilities within one shared multi-domain graph model over the same corpora.
- We develop an invertible graph vocabulary and a unified autoregressive backbone, together with task-decoupled experts that preserve shared knowledge while mitigating objective interference.
- Experiments on **20** domains, four comprehension tasks, three generation regimes, and **13** baselines show that one model can maintain competitive comprehension performance while generating valid and novel graphs, establishing an empirical foundation for unified graph modeling.

2 Related Work

2.1 Graph Models for Comprehension

Graph comprehension has progressed from task-specific graph neural networks to pre-trained models that transfer structural and semantic knowledge across tasks and domains (Liu et al., 2022; Hu et al., 2020b; Xie et al., 2022). Recent graph foundation models align heterogeneous feature spaces through prompts, coordinator nodes, masked autoencoding, or topology-aware pre-training. GCOPE (Zhao et al., 2024) connects domains through learnable coordinator nodes, SAMGPT (Yu et al., 2025) learns structure-alignment prompts, and UniGraph (He et al., 2025) uses masked autoencoding to learn transferable representations. Other methods quantize computation trees or node representations to facilitate transfer (Wang et al., 2024; Yang et al., 2024). These approaches are designed to predict node, edge, or graph properties, so their representations are pooled or encoded for discrimination and do not provide a complete graph description that supports direct generation. In our work, multi-domain comprehension is integrated with graph generation through an invertible structural serialization and discretized node semantics, allowing the same graph representation to support predictive read-out and graph reconstruction.

2.2 Graph Models for Generation

Graph generation models learn distributions over graph structures and attributes through autoregressive, diffusion-based, or flow-based objectives. Autoregressive models such as G2PT (Chen et al., 2025) serialize graphs and optimize next-token likelihood, discrete diffusion models such as DiGress (Vignac et al., 2023) learn to reverse categorical corruption processes, and discrete flow models such as DeFoG (Qin et al., 2025) generate graphs through continuous-time transition rates. These methods produce high-quality graph samples, but they are generally optimized as per-domain

generators and are not designed or evaluated to predict properties of their input graphs. In our work, generation is formulated over the same token sequence and autoregressive backbone used for comprehension, while task-decoupled experts accommodate the distinct optimization demands of the two capabilities. This design enables one multi-domain model to perform and jointly evaluate graph comprehension and generation over the same corpora.

3 PRELIMINARIES

Graphs, Domains, and the Two Task Families. A graph is $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \boldsymbol{X})$ with node set $\mathcal{V}$, edge set $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$, and node features $\boldsymbol{X} \in \mathbb{R}^{|\mathcal{V}| \times d}$. Graphs are drawn from one of M heterogeneous domains $\mathcal{D} = \{\mathcal{D}_1, \ldots, \mathcal{D}_M\}$ (e.g. citation, molecular, knowledge-graph, and synthetic structure). We study two families of tasks over such graphs. *Comprehension* maps a graph, or a node / edge, or graph pair within it, to a label y, encompassing node classification, graph classification, link prediction, and graph-level regression. *Generation* learns a distribution $p(\mathcal{G})$ from which new graphs are sampled, optionally conditioned on a target property. Conventional comprehension models in our comparison learn pooled encoders without a graph-decoding interface. Conventional generators learn $p(\mathcal{G})$ without an evaluated predictive read-out. Our primary goal is a *single* model that expresses and is evaluated on both families over the same corpora. Few-shot adaptation provides a secondary evaluation under limited target supervision.

Autoregressive Discrete Sequence Models. Given a sequence $\boldsymbol{s} = (s_1, \ldots, s_T)$ of tokens from a vocabulary $\mathcal{V}_{\text{tok}}$, a decoder-only transformer $f_{\boldsymbol{\theta}}$ factorizes its likelihood autoregressively, $p_{\boldsymbol{\theta}}(\boldsymbol{s}) = \prod_t p_{\boldsymbol{\theta}}(s_t \mid s_{<t})$, and is trained by minimizing the next-token negative log-likelihood. New sequences are produced by ancestral sampling, and a pooled function of the hidden states $\{\boldsymbol{h}_t\}$ yields a representation for prediction. Casting graphs as such sequences is therefore the enabling step for expressing generation *and* comprehension in one model, provided the graph-to-sequence map is invertible, which is the subject of Section 4.1.

Low-rank Adaptation. LoRA (Hu et al., 2022) adapts a shared weight $\boldsymbol{W} \in \mathbb{R}^{d \times d}$ by a low-rank update $\boldsymbol{W} + \Delta\boldsymbol{W}$ with $\Delta\boldsymbol{W} = \boldsymbol{B}\boldsymbol{A}$, $\boldsymbol{B} \in \mathbb{R}^{d \times r}$, $\boldsymbol{A} \in \mathbb{R}^{r \times d}$, and rank $r \ll d$. Multiple such updates can be attached to the same backbone and activated independently, giving a lightweight way to store task-specific capacity without duplicating the model. We use separate LoRA updates in Section 4.3 to reduce contention in task-specific parameters while retaining a shared backbone.

4 METHODOLOGY

In this section, we elaborate on the proposed UNICG framework, with its overall pipeline shown in Figure 1. To establish a common interface for both capabilities, we first encode graph structure and node semantics through an invertible discrete tokenization. To optimize the two objectives within one model, we process the resulting sequences with a shared decoder-only backbone and task-decoupled experts. Multi-domain pre-training then learns transferable graph knowledge before lightweight adaptation specializes each target domain.

4.1 INVERTIBLE TOKENIZATION OF GRAPH STRUCTURE AND SEMANTICS

A unified model requires a discrete interface that preserves the complete graph structure while accommodating the continuous attributes found in many domains. We construct this interface by combining deterministic structural serialization with residual quantization of node semantics.

Structure through Canonical BFS Serialization. For a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, the serializer $\mathcal{T}$ first determines a canonical node order π using breadth-first search from the highest-degree node with deterministic degree-based tie breaking. Each node is assigned an explicit index $\iota(v) = \texttt{IDX}_{\pi(v)}$. Let $\rho(v) = \tau(v)\,\iota(v)\,\sigma(v)$ denote a node record and let $\epsilon(u, w, \beta)$ denote an edge record containing two endpoint indices and the edge type β. The serializer emits a node block followed by an edge block,

$$\boldsymbol{s} = \Big[\texttt{<boc>}\ \underbrace{\rho(v_1)\cdots\rho(v_n)}_{\textit{Node block}}\ \texttt{<eoc>}\ \texttt{<bog>}\ \underbrace{\epsilon_1\cdots\epsilon_m}_{\textit{Edge block}}\ \texttt{<eog>}\Big]. \tag{1}$$

where $\tau(v)$ denotes the node type, $\sigma(v)$ contains the semantic codes defined below, and β denotes an edge type such as a molecular bond or a knowledge-graph relation. Structural and molecular edges use

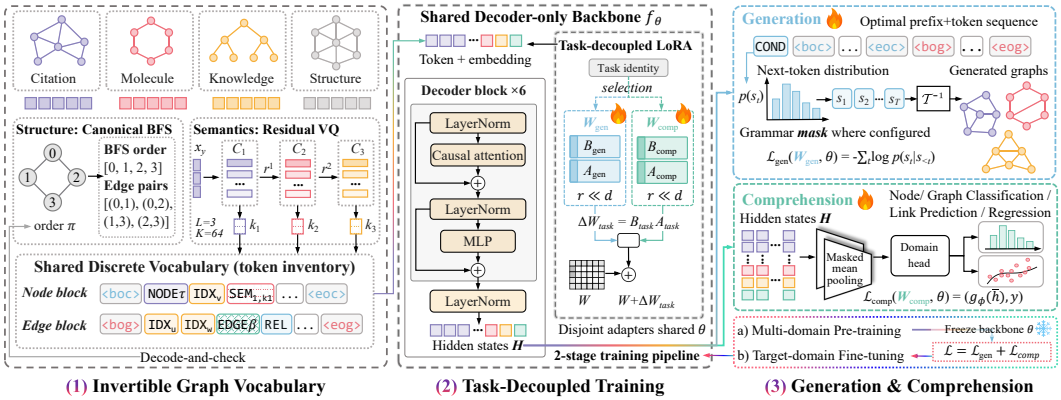


Figure 1: **Overview of UniCG. (1)** Invertible graph tokenization combines canonical structural serialization with residual quantization of node semantics. **(2)** Multi-domain pre-training and target-domain adaptation optimize one decoder-only backbone with task-decoupled experts. **(3)** Generation autoregressively decodes graph tokens, while comprehension pools contextual states for node-level, edge-level, and graph-level prediction.

$\epsilon(u, w, \beta) = [\iota(u), \iota(w), \beta]$, while knowledge-graph edges use $\epsilon(u, w, \beta) = [\iota(u), \beta, \iota(w)]$. Every edge refers to two previously declared node indices. Consequently, the node block recovers the vertex set and the edge block recovers the typed adjacency relation, giving

$$\mathcal{T}^{-1}(\mathcal{T}(\mathcal{G})) = \mathcal{G} \tag{2}$$

up to the canonical ordering. This explicit naming scheme retains isolated nodes, repeated node types, and typed edges, which are not recoverable from pooled or lossy graph codes.

Semantics through Residual Vector Quantization. Attributed graphs associate each node v with a continuous feature $\boldsymbol{x}_v$. We convert this feature into L semantic tokens using codebooks $\mathcal{C}_\ell = \{\boldsymbol{c}_{\ell,1}, \ldots, \boldsymbol{c}_{\ell,K}\}$. Starting from $\boldsymbol{r}_v^{(0)} = \boldsymbol{x}_v$, level ℓ selects the closest codeword and quantizes the remaining residual,

$$k_\ell(v) = \arg\min_{k \le K} \left\| \boldsymbol{r}_v^{(\ell-1)} - \boldsymbol{c}_{\ell,k} \right\|_2, \qquad \boldsymbol{r}_v^{(\ell)} = \boldsymbol{r}_v^{(\ell-1)} - \boldsymbol{c}_{\ell,k_\ell(v)}. \tag{3}$$

The semantic sequence and reconstructed feature are

$$\sigma(v) = \left[\mathrm{SEM}_{1,k_1(v)}, \ldots, \mathrm{SEM}_{L,k_L(v)}\right], \qquad \widehat{\boldsymbol{x}}_v = \sum_{\ell=1}^{L} \boldsymbol{c}_{\ell,k_\ell(v)}. \tag{4}$$

Successive levels encode the information left unexplained by earlier codewords. The representation therefore provides K^L possible semantic combinations using only LK vocabulary entries, increasing semantic fidelity without an impractically large flat codebook. Structure-only domains omit $\sigma(v)$, while attributed domains use the same structural grammar together with semantic tokens.

A Shared Discrete Graph Vocabulary. We divide $\mathcal{V}_{\text{tok}}$ by functional role so that the sequence explicitly distinguishes graph syntax, graph content, and task conditions.

- **Sequence and block tokens** include <pad>, <unk>, <eos>, <boc>, <eoc>, <bog>, and <eog>. They define sequence termination and separate node declarations from edge declarations, making the inverse parsing rule unambiguous.
- **Identity and type tokens** include $\mathtt{IDX}_i$, NODE, ATOM_a, EDGE, and BOND_b. Node indices preserve endpoint identity even when several nodes share the same type, while type tokens retain domain-specific structural or chemical semantics.
- **Semantic tokens** use $\mathrm{SEM}_{\ell,k}$ for codeword k at residual level ℓ. Their compositional form represents continuous node attributes without expanding the vocabulary to all cross-level code combinations.
- **Condition and relation tokens** include $\mathtt{COND}_k$ for property-controlled generation and $\mathtt{REL}_r$ for typed knowledge-graph edges and query relations. Encoding control variables as ordinary

tokens allows the same backbone to process graph content and generation conditions without an external encoder.

Each symbol is assigned one word-level token, preventing subword splitting from changing its identity. The maximum node count, semantic depth, atom and bond inventories, and relation inventory are provisioned across the configured domains. Every domain therefore uses the same embedding table and inverse serializer. Structure is preserved exactly, while node semantics are retained up to the residual quantization error. Appendix B.1 provides the construction and verification details.

4.2 Unified Decoder-Only Modeling for Generation and Comprehension

Given a serialized sequence $\boldsymbol{s} = [s_1, \ldots, s_T]$, the shared decoder-only transformer produces contextual states

$$\boldsymbol{H} = [\boldsymbol{h}_1, \ldots, \boldsymbol{h}_T] = f_{\boldsymbol{\theta}}([\boldsymbol{e}(s_1) + \boldsymbol{p}_1, \ldots, \boldsymbol{e}(s_T) + \boldsymbol{p}_T]), \tag{5}$$

where $\boldsymbol{e}$ and $\boldsymbol{p}$ are token and positional embeddings. The graph tokens and backbone parameters are shared by both capabilities. Generation uses the causal language-model head to extend the sequence, while comprehension pools the same contextual states and applies a task head.

Unconditional and Controlled Graph Generation. For unconditional generation, the prefix is empty and sampling begins from <boc>. For property-controlled generation, a continuous property is partitioned into discrete intervals and the desired interval k is represented by $\boldsymbol{c}_k = [\texttt{COND_}k]$. The training sequence is $[\texttt{COND_}k, \texttt{<boc>}, \ldots, \texttt{<eog>}, \texttt{<eos>}]$. In our controllable-generation experiments, the intervals correspond to graph-density ranges shared across the evaluated domains. Knowledge-graph sequences instead use $\boldsymbol{c}_r = [\texttt{REL_}r]$, allowing the query relation to specify the relational context. Given either an empty or task-specific prefix $\boldsymbol{c}$, the generation branch models

$$p(\boldsymbol{s} \mid \boldsymbol{c}) = \prod_{t=1}^{T} p_{\boldsymbol{\theta}, \boldsymbol{W}_{\text{gen}}}(s_t \mid \boldsymbol{c}, s_{<t}), \qquad \mathcal{L}_{\text{gen}} = -\sum_{t=1}^{T} \log p_{\boldsymbol{\theta}, \boldsymbol{W}_{\text{gen}}}(s_t \mid \boldsymbol{c}, s_{<t}). \tag{6}$$

At inference, controlled generation is prompted with $[\texttt{COND_}k, \texttt{<boc>}]$, while relation-conditioned generation is prompted with $[\texttt{REL_}r, \texttt{<boc>}]$. The decoder predicts the node block before the edge block according to the serialization grammar. Emitting <eog> and <eos> completes the sequence, after which $\widehat{\mathcal{G}} = \mathcal{T}^{-1}(\widehat{\boldsymbol{s}})$ returns the generated graph. Conditions and graph content are therefore modeled in one token space without a separate conditioning network.

Graph Comprehension from Contextual Token States. Each prediction target is converted into a graph sequence whose scope matches the task. Graph classification and graph-level regression serialize the complete graph. Let $\mathcal{N}_k(v) = \{u \in \mathcal{V} \mid \text{dist}_{\mathcal{G}}(u, v) \le k\}$ denote the k-hop neighborhood of v, and let $\mathcal{G}[S]$ denote the subgraph induced by a node set S. Node classification serializes $\mathcal{G}_v^{(k)} = \mathcal{G}[\mathcal{N}_k(v)]$ and assigns the center node v to $\texttt{IDX_0}$, making the prediction target explicit in the shared structural vocabulary. For a candidate triple (h, r, t), link prediction serializes the one-hop enclosing subgraph $\mathcal{G}_{h,t}^{(1)} = \mathcal{G}[\mathcal{N}_1(h) \cup \mathcal{N}_1(t)] \setminus \{(h, r, t)\}$. The two endpoints are assigned to $\texttt{IDX_0}$ and $\texttt{IDX_1}$, while the queried edge is excluded to prevent label leakage and $\texttt{REL_}r$ specifies its relation type. The resulting input is

$$\boldsymbol{s}_u = \begin{cases} \mathcal{T}(\mathcal{G}), & \text{Graph Classification / Regression,} \\ \mathcal{T}\left(\mathcal{G}_v^{(k)}\right), & \text{Node Classification,} \\ [\texttt{REL_}r, \mathcal{T}\left(\mathcal{G}_{h,t}^{(1)}\right)], & \text{Link Prediction.} \end{cases} \tag{7}$$

A padding mask m_t excludes padded positions when pooling the contextual states, and a lightweight domain head produces the prediction,

$$\bar{\boldsymbol{h}}_u = \frac{\sum_{t=1}^{T} m_t \boldsymbol{h}_t}{\sum_{t=1}^{T} m_t}, \qquad \widehat{y}_u = g_{\boldsymbol{\phi}_d}(\bar{\boldsymbol{h}}_u), \qquad \mathcal{L}_{\text{comp}} = \ell_d(\widehat{y}_u, y_u). \tag{8}$$

For node and graph classification, $g_{\boldsymbol{\phi}_d}$ outputs one logit per class and ℓ_d is cross-entropy. Link prediction uses the same classification path with two logits for true and corrupted candidate edges. For graph-level regression, the head outputs one value per target property, labels are standardized using the statistics available to the current training stage, and ℓ_d is mean-squared error. Predictions

are transformed back to the original target scale for evaluation. The same tokenizer and backbone therefore support graph synthesis together with node-level, edge-level, and graph-level comprehension. Within the comprehension branch, the tasks differ only in their input scope, head output dimension, and loss.

4.3 Task-Decoupled Experts for Objective-Specific Specialization

Autoregressive generation rewards locally coherent token transitions that assemble a valid global structure. Comprehension instead rewards hidden states that separate labels after sequence-level pooling. Applying both gradients to the same adaptation parameters can obscure these distinct signals. We retain a shared backbone and introduce one low-rank expert for each objective. For task $q \in \{\text{gen}, \text{comp}\}$, every adapted linear transformation becomes

$$\text{Linear}_q(\boldsymbol{h}) = \boldsymbol{W}\boldsymbol{h} + \Delta\boldsymbol{W}_q\boldsymbol{h}, \qquad \Delta\boldsymbol{W}_q = \boldsymbol{B}_q\boldsymbol{A}_q, \qquad \text{rank}(\Delta\boldsymbol{W}_q) \le r \ll d. \tag{9}$$

The shared matrix $\boldsymbol{W}$ captures graph regularities common to both objectives. The low-rank update $\Delta\boldsymbol{W}_q$ captures objective-specific transformations. Since task identity is known, routing is deterministic and introduces neither a learned gate nor an auxiliary balancing loss. The resulting objective is

$$\mathcal{L}_{\text{UniCG}}(\boldsymbol{\theta}, \boldsymbol{W}_{\text{gen}}, \boldsymbol{W}_{\text{comp}}, \boldsymbol{\phi}) = \mathcal{L}_{\text{gen}}\big(\boldsymbol{\theta}, \boldsymbol{W}_{\text{gen}}\big) + \mathcal{L}_{\text{comp}}\big(\boldsymbol{\theta}, \boldsymbol{W}_{\text{comp}}, \boldsymbol{\phi}\big), \qquad \boldsymbol{W}_{\text{gen}} \neq \boldsymbol{W}_{\text{comp}}. \tag{10}$$

Both objectives continue to shape $\boldsymbol{\theta}$, enabling cross-task transfer through the backbone, while the separate experts absorb updates specific to generation or comprehension. Section 5.4 compares this design with single-task models and direct sharing through one expert.

4.4 Two-Stage Multi-Domain Learning and Target Adaptation

The learning process separates broad representation acquisition from efficient target specialization. Multi-domain pre-training first exposes the shared backbone to heterogeneous graph structures and semantics. Target-domain adaptation then reuses this representation while adjusting only the lightweight task paths and prediction head. This separation lets a single pre-trained checkpoint support domains with different exposure and budgets without relearning the common graph interface.

Stage I: Multi-domain Pre-training. A round-robin scheduler draws batches from M domains and alternates generation and comprehension updates. The backbone, both experts, and the active domain head are jointly optimized as

$$(\boldsymbol{\theta}^\star, \boldsymbol{W}^\star_{\text{gen}}, \boldsymbol{W}^\star_{\text{comp}}, \boldsymbol{\phi}^\star) = \underset{\boldsymbol{\theta}, \boldsymbol{W}_{\text{gen}}, \boldsymbol{W}_{\text{comp}}, \boldsymbol{\phi}}{\arg\min} \sum_{d=1}^{M} \Big(\mathcal{L}^{(d)}_{\text{gen}} + \mathcal{L}^{(d)}_{\text{comp}} \Big). \tag{11}$$

Interleaving domains prevents corpus size from determining the update frequency. Interleaving objectives allows the backbone to acquire reusable structural and semantic patterns while each expert develops its own optimization path.

Stage II: Target-domain Adaptation. For a target domain $\mathcal{D}_m$, we restore the pre-trained parameters, freeze the backbone, token embedding, and output projection, and initialize a fresh prediction head. Adaptation updates the two experts and the new head through

$$(\widehat{\boldsymbol{W}}_{\text{gen}}, \widehat{\boldsymbol{W}}_{\text{comp}}, \widehat{\boldsymbol{\phi}}_m) = \underset{\boldsymbol{W}_{\text{gen}}, \boldsymbol{W}_{\text{comp}}, \boldsymbol{\phi}_m}{\arg\min} \Big(\mathcal{L}^{(m)}_{\text{gen}} + \mathcal{L}^{(m)}_{\text{comp}} \Big) \quad \text{with} \quad \boldsymbol{\theta} = \boldsymbol{\theta}^\star. \tag{12}$$

Freezing the backbone preserves the multi-domain representation and reduces adaptation to approximately 2.36M of 13.4M parameters. Comprehension uses either the full target training split or a k-shot support set. Generation remains unsupervised and learns from target-domain graph sequences. Uni-Seen adaptation specializes a domain that participated in pre-training, whereas Uni-Unseen adaptation transfers the same framework to a held-out domain. Appendix C.3 specifies these settings, and Appendix B.5 provides the complete algorithms and optimization details.

5 Experiments

We evaluate whether one model can support graph comprehension and generation, and study the architectural choices that make their joint optimization effective. Specifically, we address the following research questions:

Table 1: Performance (%, mean±std over five runs) of unified graph comprehension and generation under Uni-Seen. Comprehension uses accuracy for node classification and ROC-AUC for graph classification and link prediction. Generation uses V.U.N. Best results are presented in **bold**, and runner-ups are underlined.

Task	**Node-level (Acc.)**			**Graph-level (ROC-AUC)**		**Link-level (ROC-AUC)**	
Method	Cora	PubMed	Arxiv	BACE	BBBP	WN18RR	FB15k-237
				Comprehension ↑			
GCN [11]	26.72 ±0.33	36.00 ±1.01	30.00 ±0.32	52.79 ±0.08	62.12 ±0.04	62.99 ±0.30	56.91 ±0.30
GAT [22]	29.97 ±0.62	36.07 ±1.04	35.65 ±0.92	52.31 ±0.10	61.87 ±0.11	63.71 ±0.15	58.69 ±0.97
SAMGPT [32]	52.27 ±4.44	68.13 ±4.26	35.44 ±2.49	59.86 ±1.79	70.24 ±0.99	60.29 ±1.86	58.82 ±0.42
MDGPT [31]	36.23 ±4.76	62.47 ±6.41	33.28 ±3.96	57.11 ±3.45	70.02 ±0.45	62.53 ±1.79	59.15 ±0.98
MDGFM [24]	45.23 ±5.68	63.23 ±5.70	27.90 ±1.81	51.93 ±2.64	66.43 ±3.48	53.48 ±5.47	51.99 ±5.49
GCOPE [34]	54.43 ±2.69	64.67 ±3.09	50.20 ±1.27	61.62 ±0.35	**73.35 ±0.39**	71.68 ±0.77	65.57 ±0.42
GFT [25]	29.33 ±0.00	36.80 ±1.06	14.17 ±1.96	52.02 ±3.95	65.78 ±1.18	60.58 ±4.87	44.81 ±2.95
UniGraph2 [5]	60.17 ±0.91	**75.96 ±0.93**	51.27 ±3.14	60.89 ±0.73	70.99 ±1.58	67.71 ±2.64	63.49 ±2.09
BRIDGE [33]	55.17 ±1.31	69.57 ±1.65	**52.97 ±1.59**	61.08 ±0.84	72.16 ±0.56	72.91 ±0.81	65.63 ±1.01
UniCG	**64.47 ±0.62**	72.07 ±1.82	52.60 ±1.33	**65.13 ±2.59**	67.87 ±1.93	**95.66 ±0.70**	**88.61 ±1.57**
				Generation (V.U.N. %) ↑			
DiGress [23]	21.95 ±6.13	19.45 ±11.45	19.69 ±10.93	0.94 ±1.12	0.23 ±0.47	58.12 ±6.05	92.34 ±5.94
DeFoG [15]	**36.33 ±4.16**	**46.10 ±12.72**	**62.50 ±9.75**	2.11 ±1.83	2.73 ±1.40	**73.67 ±4.18**	**100.00 ±0.00**
UniCG	36.03 ±2.53	36.70 ±2.54	37.33 ±3.04	**58.49 ±6.32**	**59.92 ±2.93**	54.37 ±3.12	92.59 ±1.96

- **RQ1:** Can a single model comprehend and generate graphs from the same corpora? (▷ Section 5.2)
- **RQ2:** Does unified modeling preserve few-shot graph comprehension? (▷ Section 5.3)
- **RQ3:** How do the architectural variants affect the two capabilities? (▷ Section 5.4)
- **RQ4:** How sensitive is UniCG to expert capacity and vocabulary fidelity? (▷ Section 5.5)

5.1 Experimental Settings

Datasets. We assemble **20** graph domains from five groups. Citation and co-purchase networks support node classification. Molecular graphs support property prediction. Knowledge graphs support link prediction. Synthetic structure domains support both regression and generation, while controlled synthetic domains isolate the effects of structure and semantic quantization. Molecular generation additionally uses QM9 and ZINC250k. Dataset statistics and further details are in Appendix C.1.

Baselines. We compare against **13** baselines. Comprehension baselines include GCN (Kipf & Welling, 2017), GAT (Veličković et al., 2018), SAMGPT (Yu et al., 2025), MDGPT (Yu et al., 2024), MDGFM (Wang et al., 2025), GCOPE (Zhao et al., 2024), GFT (Wang et al., 2024), UniGraph2 (He et al., 2025), and BRIDGE (Yuan et al., 2025). Generation baselines include an Erdős–Rényi reference, G2PT (Chen et al., 2025), DiGress (Vignac et al., 2023), and DeFoG (Qin et al., 2025). The first group provides no graph-generation interface, while the second provides no predictive read-out evaluated on comprehension tasks.

Adaptation settings. We evaluate UniCG and the baselines under three settings that vary target-domain exposure and supervision. Appendix C.3 specifies the data available in each setting.

- **Uni-Seen with full-data adaptation:** pre-training includes the target domain, followed by adaptation on its complete training split (▷ Table 1 and Appendix E).
- **Uni-Unseen with full-data adaptation:** pre-training excludes the target domain, followed by adaptation on its complete training split (▷ Appendices E and G).
- **Uni-Unseen with few-shot adaptation:** pre-training excludes the target domain, after which comprehension uses k labelled examples per class and generation uses unlabeled target graphs (▷ Table 2 and Appendix F).

Metrics. We report accuracy for node classification, ROC-AUC for molecular classification and link prediction, and R^2 for graph regression. Generation is evaluated by V.U.N., which measures the fraction of graphs that are simultaneously valid, unique, and novel, together with the mean MMD of degree, clustering, and spectral statistics. The shared decoder-only backbone has 10.65M parameters.

Table 2: Performance (%, mean±std over five runs) of **5-shot graph comprehension** under Uni-Unseen. Node-level results use accuracy. Graph- and link-level results use ROC-AUC. Best results are presented in **bold**, and runner-ups are underlined.

Task	Node-level (Acc.)			Graph-level (ROC-AUC)		Link-level (ROC-AUC)	
Method	Cora	PubMed	Arxiv	BACE	BBBP	WN18RR	FB15k-237
GCN [11]	28.33 ±2.00	35.20 ±3.47	12.93 ±3.53	48.51 ±1.71	55.29 ±7.47	59.76 ±0.79	57.14 ±7.71
GAT [22]	26.67 ±2.62	31.93 ±4.78	15.73 ±2.33	48.07 ±3.09	54.42 ±7.01	58.09 ±1.32	55.30 ±6.67
SAMGPT [32]	43.80 ±4.33	55.47 ±6.51	11.00 ±5.69	55.96 ±2.65	62.40 ±9.67	60.49 ±0.69	53.62 ±9.27
MDGPT [31]	41.47 ±4.16	50.53 ±4.27	13.10 ±2.37	57.09 ±2.07	62.11 ±9.90	59.65 ±1.27	56.72 ±6.69
MDGFM [24]	43.53 ±2.75	50.67 ±8.43	13.37 ±5.10	55.72 ±4.68	64.92 ±7.07	55.58 ±2.94	51.94 ±7.65
GCOPE [34]	30.27 ±5.50	40.83 ±5.71	18.08 ±6.47	55.96 ±2.44	64.37 ±1.42	59.26 ±1.03	56.89 ±7.59
GFT [25]	35.50 ±8.35	36.00 ±1.33	14.46 ±4.41	49.53 ±1.15	49.76 ±1.28	51.34 ±1.56	50.01 ±0.03
UniGraph2 [5]	38.00 ±6.01	45.00 ±1.91	22.04 ±5.43	55.17 ±2.42	63.49 ±7.77	**65.03 ±3.56**	**61.15 ±1.65**
BRIDGE [33]	37.66 ±2.73	52.67 ±3.85	21.57 ±6.64	53.93 ±1.97	**68.71 ±2.00**	58.81 ±0.90	57.90 ±8.56
UniCG	**44.20 ±0.98**	**56.73 ±4.17**	**24.47 ±1.38**	**61.13 ±3.10**	64.94 ±6.41	63.33 ±2.14	60.55 ±2.30

Six layers of width 384 and rank-32 LoRA experts give UniCG approximately 13.4M parameters. Complete implementation and evaluation details appear in Appendix C.

5.2 RQ1: Unified Comprehension and Generation over Shared Corpora

Table 1 evaluates comprehension and generation on the same seven corpora, where only UniCG supports both blocks, since the comprehension baselines do not decode graphs and the generation baselines do not provide task predictions.

Comprehension. UniCG achieves the best performance on Cora, BACE, WN18RR, and FB15k-237, and ranks second on PubMed and Arxiv. On the two relational datasets, it outperforms the strongest external baseline by more than 20 points. The gains span node-, graph-, and link-level prediction, showing that the unified token sequence supports several prediction units instead of favoring one task family.

Generation. UniCG performs best on BACE and BBBP, remains comparable to DeFoG on Cora (36.0 vs. 36.3), and ranks second on four of the seven corpora. Dedicated generators lead on WN18RR, where Table E.5 additionally measures distributional fidelity. Overall, UniCG ranks among the top two methods in six of seven columns for each capability, providing the central evidence that one parameterization can support both sides of the unified evaluation. Full Uni-Seen and Uni-Unseen results are reported in Appendix E.

5.3 RQ2: Few-Shot Adaptation to Unseen Graph Domains

Few-shot adaptation evaluates whether the unified representation remains useful when target supervision is limited, which is a common adaptation setting for graph foundation models. In the representative 5-shot Uni-Unseen results of Table 2, UniCG ranks first on Cora, PubMed, Arxiv, and BACE, and second on the remaining three datasets. On the three node-classification datasets, the gains over the runner-up are 0.4, 1.3, and 2.4 points, respectively, while the gain on BACE reaches 4.0 points. On WN18RR and FB15k-237, the gaps to the best baseline are only 1.7 and 0.6 points. The pattern shows that the representation learned jointly with generation remains effective when target supervision is scarce and transfers across node-, graph-, and link-level prediction. Complete 1-, 5-, and 10-shot results are provided in Appendix F.

5.4 RQ3: Contributions of the Architectural Components

We conduct ablation studies to analyze the effectiveness of the task-decoupled architecture:

- ***Comp.Only*:** removes the generation objective, providing a matched comprehension specialist.
- ***Gen.Only*:** removes the comprehension objective, providing a matched generation specialist.
- ***Naive*:** routes both objectives through one shared expert, testing direct joint optimization.

Results are shown in Table 3. Under Uni-Seen comprehension, *Naive* achieves strong performance but suffers substantial generation degradation, indicating interference from direct parameter sharing.

Table 3: Performance (%, mean±std over five runs) of architectural variants under Uni-Seen and 5-shot Uni-Unseen. Comprehension uses accuracy for node classification and ROC-AUC for graph classification and link prediction. Generation uses V.U.N. Best results are presented in **bold**, and runner-ups are underlined.

Task	Node-level (Acc.)			Graph-level (ROC-AUC)		Link-level (ROC-AUC)	
Variant	Cora	PubMed	Arxiv	BACE	BBBP	WN18RR	FB15k-237
Comprehension, Uni-Seen ↑							
Comp.Only	63.47 ±1.76	71.40 ±1.68	50.67 ±1.19	62.73 ±1.24	67.29 ±1.80	92.95 ±0.85	85.38 ±0.55
Naive	**64.87 ±1.69**	**73.73 ±1.27**	**53.50 ±0.33**	61.64 ±1.29	67.54 ±0.49	**97.37 ±0.27**	**93.49 ±0.28**
UniCG	64.47 ±0.62	72.07 ±1.82	52.60 ±1.33	**65.13 ±2.59**	**67.87 ±1.93**	95.66 ±0.70	88.61 ±1.57
Generation, Uni-Seen (V.U.N. %) ↑							
Gen.Only	**41.05 ±3.14**	**38.25 ±3.64**	37.27 ±3.63	58.20 ±4.68	59.61 ±2.25	48.20 ±2.04	92.20 ±1.16
Naive	32.24 ±4.22	34.31 ±4.48	31.66 ±4.06	12.89 ±3.79	12.37 ±1.61	**85.55 ±3.33**	91.18 ±2.56
UniCG	36.03 ±2.53	36.70 ±2.54	**37.33 ±3.04**	**58.49 ±6.32**	**59.92 ±2.93**	54.37 ±3.12	**92.59 ±1.96**
Comprehension, 5-shot Uni-Unseen ↑							
Comp.Only	**44.67 ±4.21**	**61.73 ±6.86**	21.40 ±4.59	60.42 ±9.03	**68.64 ±2.72**	63.51 ±3.89	51.68 ±3.05
Naive	43.53 ±1.09	56.40 ±3.73	22.47 ±3.15	56.66 ±4.95	60.41 ±4.29	**65.60 ±2.97**	58.99 ±1.90
UniCG	44.20 ±0.98	56.73 ±4.17	**24.47 ±1.38**	**61.13 ±3.10**	64.94 ±6.41	63.33 ±2.14	**60.55 ±2.30**

Task decoupling largely recovers generation performance while maintaining competitive comprehension. In the 5-shot setting, UniCG also shows strong transfer across domains, although some domain-dependent interactions remain. Additional MMD results and frozen-backbone diagnostics are provided in Appendix H.

5.5 RQ4: Sensitivity to Expert Capacity and Vocabulary Fidelity

Sensitivity Analysis of LoRA Rank and Scaling. We vary the LoRA rank r and scaling factor α in Figure 2 across node classification, graph classification, and link prediction.

Sensitivity Analysis of RVQ Depth and Codebook Size. We further evaluate the RVQ depth L and codebook size K, which control the fidelity of semantic tokenization in Figure 3 and Figure 4. Additional comprehension and generation sensitivity results are provided in Appendix H.

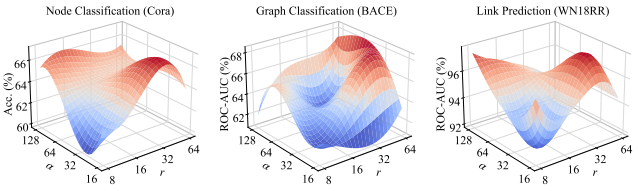


Figure 2: Sensitivity of $r \times \alpha$ grid on representative domains.

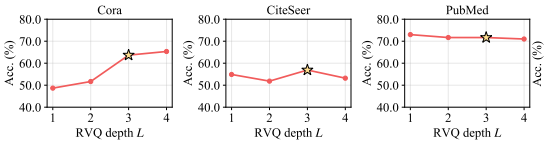


Figure 3: Sensitivity of RVQ depth L.

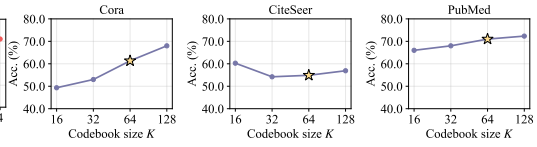


Figure 4: Sensitivity of codebook size K.

6 Conclusion

We introduced **UniCG**, a framework that enables one model to comprehend and generate graphs over the same corpora. UniCG constructs a shared discrete interface by combining lossless structural serialization with residual quantization of node semantics. A decoder-only backbone then supports autoregressive graph generation and pooled graph comprehension, while task-decoupled experts preserve task-specific optimization paths within the shared model. Experiments across 20 graph domains, four comprehension tasks, three generation regimes, and 13 baselines show that UniCG maintains competitive predictive performance while producing valid and novel graphs. These results establish unified comprehension and generation as a viable design principle for graph learning and provide a foundation for future graph models that integrate prediction, synthesis, and interactive graph reasoning within a single architecture.

AI USE STATEMENT

In this work, we used generative AI tools to revise the text.

ETHICS STATEMENT

This work uses established public graph benchmarks and does not involve human participants or the collection of personal data. The proposed framework is general-purpose, and downstream applications should follow the licenses of their datasets and the safety requirements of their deployment domains.

REPRODUCIBILITY STATEMENT

We provide the information required to reproduce the study throughout the paper and appendices. Appendix B specifies the graph vocabulary, serialization, model architecture, and training algorithms. Appendix C documents dataset processing, baseline definitions, adaptation settings, metrics, decoding, and experiment-specific configurations. Appendix D reports model hyperparameters, implementation sources and licenses, and the computing environment. Appendices E–H report complete results beyond the main tables.

REFERENCES

Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *Science*, 286 (5439):509–512, 1999. doi: 10.1126/science.286.5439.509.

Xiaohui Chen, Yinkai Wang, Jiaxing He, Yuanqi Du, Soha Hassoun, Xiaolin Xu, and Li-Ping Liu. Graph generative pre-trained transformer. In *International Conference on Machine Learning (ICML)*, 2025. arXiv:2501.01073.

Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. Convolutional 2d knowledge graph embeddings. In *AAAI Conference on Artificial Intelligence*, 2018.

Paul Erdős and Alfréd Rényi. On the evolution of random graphs. *Publications of the Mathematical Institute of the Hungarian Academy of Sciences*, 5(1):17–60, 1960.

Yufei He, Yuan Sui, Xiaoxin He, and Bryan Hooi. UniGraph: Learning a unified cross-domain foundation model for text-attributed graphs. In *KDD*, pp. 448–459, 2025.

Paul W. Holland, Kathryn Blackmond Laskey, and Samuel Leinhardt. Stochastic blockmodels: First steps. *Social Networks*, 5(2):109–137, 1983. doi: 10.1016/0378-8733(83)90021-7.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations (ICLR)*, 2022.

Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. Open graph benchmark: Datasets for machine learning on graphs. *NeurIPS*, 33:22118–22133, 2020a.

Weihua Hu, Bowen Liu, Joseph Gomes, Marinka Zitnik, Percy Liang, Vijay Pande, and Jure Leskovec. Strategies for pre-training graph neural networks. In *ICLR*, 2020b.

John J. Irwin, Teague Sterling, Michael M. Mysinger, Erin S. Bolstad, and Ryan G. Coleman. ZINC: A free tool to discover chemistry for biology. *Journal of Chemical Information and Modeling*, 52 (7):1757–1768, 2012. doi: 10.1021/ci3001277.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*, 2017.

Yixin Liu, Ming Jin, Shirui Pan, Chuan Zhou, Yu Zheng, Feng Xia, and S Yu Philip. Graph self-supervised learning: A survey. *IEEE TKDE*, 35(6):5879–5900, 2022.

Zemin Liu, Xingtong Yu, Yuan Fang, and Xinming Zhang. GraphPrompt: Unifying pre-training and downstream tasks for graph neural networks. In *WWW*, pp. 417–428, 2023.

Péter Mernyei and Cătălina Cangea. Wiki-CS: A wikipedia-based benchmark for graph neural networks. *arXiv preprint arXiv:2007.02901*, 2020.

Yiming Qin, Manuel Madeira, D. Thanou, and Pascal Frossard. Defog: Discrete flow matching for graph generation. In *International Conference on Machine Learning (ICML)*, 2025. arXiv:2410.04263.

Raghunathan Ramakrishnan, Pavlo O. Dral, Matthias Rupp, and O. Anatole von Lilienfeld. Quantum chemistry structures and properties of 134 kilo molecules. *Scientific Data*, 1, 2014. doi: 10.1038/sdata.2014.22.

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.

Oleksandr Shchur, Maximilian Mumme, Aleksandar Bojchevski, and Stephan Günnemann. Pitfalls of graph neural network evaluation. *arXiv preprint arXiv:1811.05868*, 2018.

Xiangguo Sun, Hong Cheng, Jia Li, Bo Liu, and Jihong Guan. All in one: Multi-task prompting for graph neural networks. In *KDD*, pp. 2120–2131, 2023.

Chameleon Team, Mingda Chen, Jacob Kahn, and Shang-Wen Li. Chameleon: Mixed-modal early-fusion foundation models. *arXiv preprint arXiv:2405.09818*, 2024.

Kristina Toutanova and Danqi Chen. Observed versus latent features for knowledge base and text inference. In *Workshop on Continuous Vector Space Models and their Compositionality*, 2015.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations (ICLR)*, 2018.

Clément Vignac, Igor Krawczuk, Antoine Siraudin, Bohan Wang, V. Cevher, and P. Frossard. Digress: Discrete denoising diffusion for graph generation. In *International Conference on Learning Representations (ICLR)*, 2023. arXiv:2209.14734.

Shuo Wang, Bokui Wang, Zhixiang Shen, Boyan Deng, and Zhao Kang. Multi-domain graph foundation models: Robust knowledge transfer via topology alignment. In *ICML*, 2025.

Zehong Wang, Zheyuan Zhang, Nitesh Chawla, Chuxu Zhang, and Yanfang Ye. GFT: Graph foundation model with transferable tree vocabulary. *NeurIPS*, 37:107403–107443, 2024.

Chengyue Wu, Xiaokang Chen, Zhiyu Wu, Yiyang Ma, Xingchao Liu, Zizheng Pan, Wen Liu, Zhenda Xie, et al. Janus: Decoupling visual encoding for unified multimodal understanding and generation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025. arXiv:2410.13848.

Zhenqin Wu, Bharath Ramsundar, Evan N. Feinberg, Joseph Gomes, Caleb Geniesse, Aneesh S. Pappu, K. Leswing, and V. Pande. Moleculenet: a benchmark for molecular machine learning. *Chemical Science*, 9(2):513–530, 2018.

Yaochen Xie, Zhao Xu, Jingtun Zhang, Zhengyang Wang, and Shuiwang Ji. Self-supervised learning of graph neural networks: A unified review. *IEEE TPAMI*, 45(2):2412–2429, 2022.

Ling Yang, Ye Tian, Minkai Xu, Zhongyi Liu, Shenda Hong, Wei Qu, Wentao Zhang, Bin Cui, et al. Vqgraph: Rethinking graph representation space for bridging gnns and mlps. In *International Conference on Learning Representations (ICLR)*, 2024. arXiv:2308.02117.

Zhilin Yang, William W. Cohen, and Ruslan Salakhutdinov. Revisiting semi-supervised learning with graph embeddings. In *International Conference on Machine Learning*, pp. 40–48. PMLR, 2016.

Xingtong Yu, Chang Zhou, Yuan Fang, and Xinming Zhang. Text-free multi-domain graph pre-training: Toward graph foundation models. *arXiv preprint arXiv:2405.13934*, 2024.

Xingtong Yu, Zechuan Gong, Chang Zhou, Yuan Fang, and Hui Zhang. SAMGPT: Text-free graph foundation model for multi-domain pre-training and cross-domain adaptation. In *WWW*, pp. 1142–1153, 2025.

Haonan Yuan, Qingyun Sun, Junhua Shi, Xingcheng Fu, Bryan Hooi, Jianxin Li, and Philip S. Yu. How much can transfer? bridge: Bounded multi-domain graph foundation model with generalization guarantees. In *International Conference on Machine Learning (ICML)*, 2025.

Haihong Zhao, Aochuan Chen, Xiangguo Sun, Hong Cheng, and Jia Li. All in one and one for all: A simple yet effective method towards cross-domain graph pretraining. In *KDD*, pp. 4443–4454, 2024.

Appendix

Table of Contents

A Notation **14**

B Method Implementation Details **14**
B.1 Construction of the Shared Graph Vocabulary 15
B.2 Deterministic and Invertible Graph Serialization 15
B.3 Residual Quantization of Node Attributes 15
B.4 Backbone and Objective-Specific Expert Routing 15
B.5 Two-Stage Pre-training and Target Adaptation 16

C Experiment Details **18**
C.1 Dataset Details 19
C.2 Baseline Details 20
C.3 Adaptation Settings 21
C.4 Metrics and Decoding 21
C.5 Experiment-Specific Settings 22

D Implementation Details **23**
D.1 Implementation Details of UniCG 23
D.2 Implementation Details of Baselines 23
D.3 Hardware and Software Configurations 24

E Additional Results: Uni-Seen and Uni-Unseen Evaluation **24**
E.1 Full-Data Adaptation on Seen Domains 24
E.2 Full-Data Adaptation to Unseen Domains 28

F Additional Results: Few-Shot Comprehension **28**
F.1 One-Shot Adaptation to Unseen Domains 28
F.2 Five-Shot Adaptation to Unseen Domains 30
F.3 Ten-Shot Adaptation to Unseen Domains 31

G Additional Results: Graph Generation **34**

H Additional Results: Ablation and Sensitivity Analysis **34**
H.1 A Frozen-Backbone Test of Expert Decoupling 34
H.2 Sensitivity across Expert Capacity and Vocabulary Fidelity 35

I Discussion and Limitations **36**

A NOTATION

Table A.1 summarizes the notation used throughout the paper.

Notations	Descriptions
	Graphs and domains
$\mathcal{G} = (\mathcal{V}, \mathcal{E}, \boldsymbol{X})$	Graph with node set $\mathcal{V}$, edge set $\mathcal{E}$, and node-feature matrix $\boldsymbol{X}$
$\mathcal{V},\ \mathcal{E}$	The set of nodes and the set of edges in the graph
$\boldsymbol{X} \in \mathbb{R}^{n \times d}$	Node feature matrix, $n = \lvert\mathcal{V}\rvert$ nodes of input dimension d
$\boldsymbol{x}_v \in \mathbb{R}^d$	Continuous feature vector of node v
$\mathcal{D} = \{\mathcal{D}_1, \ldots, \mathcal{D}_M\}$	The set of M graph domains participating in a run
$\mathcal{D}_m$	The domain held out under Uni-Unseen (unseen in pre-training)
$\mathcal{C}_d$	The token corpus of domain d, from which training batches $\mathcal{B}$ are drawn
	Vocabulary and serialization
$\mathcal{T} : \mathcal{G} \mapsto \boldsymbol{s}$	The invertible graph-to-sequence serialization map
$\mathcal{T}^{-1}$	Its inverse, decoding a token sequence back into a graph
$\boldsymbol{s} = [s_1, \ldots, s_T]$	The token sequence of a graph, of length T
$\mathcal{V}_{\text{tok}}$	The shared token vocabulary, the union of the seven token groups
$\mathcal{S}$	Special tokens, including the node/edge block delimiters
π	The canonical node order (BFS from the highest-degree node)
$\iota(v) = \texttt{IDX}_{\pi(v)}$	Node-index token, the explicit name that makes $\mathcal{T}$ invertible
$\tau(v)$	Node-type token (atom type, or a single `NODE` type)
β	Edge-type token (bond type, or a single `EDGE` type)
$\sigma(v)$	The semantic token string of node v, i.e. its RVQ codes
$\texttt{REL}_r$	Relation token naming a typed edge in a knowledge graph
$\texttt{COND}_k$	Condition token naming a target property bucket
$n_{\max}$	Maximum node count over the participating domains
	Residual vector quantization
L	Number of residual quantization levels
K	Codebook size (number of centroids) per level
$\mathcal{C}_\ell = \{\boldsymbol{c}_{\ell,1}, \ldots, \boldsymbol{c}_{\ell,K}\}$	The codebook at level ℓ
$\boldsymbol{r}_v^{(\ell)}$	The residual of node v after the first ℓ quantization levels
$k_\ell(v)$	The code index assigned to node v at level ℓ
$\texttt{SEM}_{\ell,k}$	The semantic token for code k at level ℓ
	Model, experts, and objectives
$f_{\boldsymbol{\theta}}$	The shared decoder-only transformer backbone
$\boldsymbol{\theta}$	Backbone parameters, trained from scratch and shared by both tasks
$\boldsymbol{h}_t$	Final-layer hidden state at position t
$\bar{\boldsymbol{h}}_{\boldsymbol{\theta}}(\boldsymbol{s})$	Mask-mean-pooled sequence representation used for comprehension
$g_{\boldsymbol{\phi}}$	Per-domain comprehension head with parameters $\boldsymbol{\phi}$
$\boldsymbol{W}_{\text{shared}}$	The *single* shared adapter of the *Naive* variant
$\boldsymbol{W}_{\text{gen}},\ \boldsymbol{W}_{\text{comp}}$	The two task-decoupled LoRA experts of UNICG, each $\boldsymbol{W}_{\text{task}} = \boldsymbol{B}_{\text{task}}\boldsymbol{A}_{\text{task}}$
$\Delta\boldsymbol{W} = \boldsymbol{B}\boldsymbol{A}$	The low-rank update applied to a frozen weight matrix $\boldsymbol{W}$
r	LoRA rank of each expert
$\mathcal{L}_{\text{gen}}$	Generation loss, the next-token cross-entropy over $\boldsymbol{s}$
$\mathcal{L}_{\text{comp}}$	Comprehension loss, cross-entropy or squared error on the pooled read-out
$\mathcal{L}_{\text{joint}}$	The naive joint objective over one shared adapter (Eq. (B.1))
$\mathcal{L}_{\text{UNICG}}$	Our objective over two decoupled experts (Eq. (10))
y	The comprehension target (a class label or a regression value)
M	Number of domains in a run (round-robin period of a training step)
k	Number of labelled examples per class in the k-shot setting

Table A.1: Summary of notations and descriptions.

B METHOD IMPLEMENTATION DETAILS

This section provides the details required to reimplement UNICG. Appendices B.1 through B.3 describe the vocabulary, serializer, and quantizer. Appendix B.4 specifies the backbone, experts,

and read-out head, while Appendix B.5 presents pre-training and adaptation in Algorithms 1 and 2. Implementation and optimization hyperparameters are consolidated with the experimental setup in Appendix D.

B.1 Construction of the Shared Graph Vocabulary

The main text introduces the token groups and their roles in Section 4.1. This section records the construction details required for reimplementation. The vocabulary $\mathcal{V}_{\text{tok}}$ is built once per run and wrapped in a word-level tokenizer with a whitespace pre-tokenizer, so that one graph token is exactly one vocabulary entry. This is necessary because a sub-word tokenizer could split `IDX_17` into pieces that no longer identify a node, thereby destroying the bijection required for generation. Two implementation-level separators, `<sepc>` and `<sepg>`, delimit adjacent node records and adjacent edge records, respectively. They make variable-length record boundaries explicit for the inverse parser without encoding graph content.

The vocabulary is rebuilt per run from the schema inventories of all configured domains, including a domain held out from graph-level pre-training under Uni-Unseen. Its size is determined by the resulting maxima: $n_{\max}$ index tokens, LK semantic tokens, one token per distinct atom and bond type, and one per distinct relation. The resulting multi-domain vocabulary contains a few hundred entries. A held-out domain therefore uses a provisioned schema while none of its graphs contribute to pre-training. Uni-Unseen is transductive in schema and inductive in graph data (Appendix C.3).

B.2 Deterministic and Invertible Graph Serialization

The serialization map $\mathcal{T}$ and its canonical order are defined in the main text (Eq. (1)). Three implementation points complete it. First, the canonical order π is a breadth-first traversal seeded at the highest-degree node with ties broken by descending degree, which removes the $n!$ orderings a graph would otherwise admit and gives the autoregressive factorization a single consistent target. Without this order, the model would fit a distribution over permutations rather than graphs. Second, node and edge blocks are emitted separately, so every node is named before any edge refers to it, and a partially generated sequence is always decodable. Third, **invertibility is checked, not assumed**: during corpus construction every sequence is decoded back through $\mathcal{T}^{-1}$ and compared against its source graph, and any sample whose decoded node count disagrees is discarded. Structure is therefore lossless by construction. Node semantics pass through the quantizer below and are recovered up to its quantization error.

B.3 Residual Quantization of Node Attributes

The residual quantizer that turns continuous node features into `SEM` tokens is defined in the main text (Section 4.1): L levels, each a codebook of K centroids fit by k-means on the residual left by the previous level, giving K^L effective clusters from LK centroids. We use $L = 3$ and $K = 64$, so three tokens per node address $64^3 \approx 2.6 \times 10^5$ feature clusters using 192 vocabulary entries. A flat codebook at the same resolution would require 2.6×10^5 entries.

The quantizer uses deterministic k-means without a learned encoder and is fit once before backbone training. All method variants therefore use the same fixed vocabulary. Attributed graphs use their dataset-native node features without an external language-model encoder. We quantize 2-hop ego-graphs capped at 30 nodes, bounding sequence length while retaining each center node's local neighbourhood.

We ablate L in Appendix H to measure how semantic resolution affects comprehension through the discrete interface.

B.4 Backbone and Objective-Specific Expert Routing

Backbone. We use a decoder-only transformer based on the GPT-2 architecture, comprising 6 layers, 6 heads, width 384, a context length of 1024, and 10.65M backbone parameters. The complete model contains 13.3–13.5M parameters depending on the vocabulary size. The model is initialized from scratch without pre-trained language or graph weights.

Two experts. Both objectives run on this one backbone through separate low-rank adapters (Hu et al., 2022): a generation expert $\boldsymbol{W}_{\text{gen}}$ and a comprehension expert $\boldsymbol{W}_{\text{comp}}$, each of rank $r = 32$ with scaling $\alpha = 2r$. The low-rank updates are applied to the combined attention projection, the attention output projection, and the MLP expansion projection. Each expert adds 1.18M parameters, so unification costs about 10% over a single-task model. The *Naive* variant is exactly this model with *one* adapter serving both objectives. This change isolates the effect of sharing objective-specific parameters.

Ablation variants. The direct joint baseline routes both objectives through one shared adapter and optimizes their summed loss:

$$\mathcal{L}_{\text{joint}}(\boldsymbol{\theta}, \boldsymbol{W}_{\text{shared}}, \boldsymbol{\phi}) = \mathcal{L}_{\text{gen}}\big(\boldsymbol{\theta}, \boldsymbol{W}_{\text{shared}}\big) + \mathcal{L}_{\text{comp}}\big(\boldsymbol{\theta}, \boldsymbol{W}_{\text{shared}}, \boldsymbol{\phi}\big). \tag{B.1}$$

This formulation defines *Naive*. Disabling the comprehension or generation branch yields *Gen.Only* or *Comp.Only*, respectively. These variants preserve the backbone, data, training budget, and optimization settings, isolating the effect of task-specific parameter sharing.

The router is parameter-free. Task identity is known at training and inference time, so a lookup activates the corresponding expert. The architecture introduces no gating network, load-balancing loss, or learned mixture weights.

The backbone is trained during pre-training and frozen during adaptation. Since the backbone is initialized from scratch, multi-domain pre-training updates it jointly with both experts. Target-domain adaptation then freezes the backbone and updates only the experts and prediction head. Pre-training learns the shared representation, while adaptation specializes its task paths under a restricted parameter budget. Appendix H reports a frozen-backbone diagnostic that measures the interference retained in the shared representation.

Comprehension head. Comprehension is implemented through a predictive read-out. We mask-mean-pool the final hidden states, $\bar{\boldsymbol{h}}_{\boldsymbol{\theta}}(\boldsymbol{s}) = (\sum_t m_t \boldsymbol{h}_t)/\sum_t m_t$, and apply a per-domain linear head $g_{\boldsymbol{\phi}}$, with cross-entropy for classification and mean-squared error on normalized targets for regression. Regression targets are standardized using statistics computed *only* from the labels the current stage is permitted to see, which is the k-shot support subset during few-shot adaptation, so no full-train statistic leaks into a few-shot number. The reported comprehension results measure the predictive information retained in the pooled graph representation.

B.5 Two-Stage Pre-training and Target Adaptation

Algorithm 1 learns the shared multi-domain representation, and Algorithm 2 specializes the resulting checkpoint to a target domain. Both stages retain the same generation and comprehension branches, while their trainable parameter sets and available data differ.

Pre-training. At step t the scheduler selects domain $d = \mathcal{D}[t \bmod M]$, cycling round-robin so every domain contributes equally regardless of corpus size. The generation expert is activated and a batch of sequences is drawn from $\mathcal{C}_d$ for a next-token cross-entropy gradient. The comprehension expert is then activated and a second batch is drawn for the pooled-read-out gradient. Both gradients accumulate into the shared backbone before a single AdamW step updates the backbone, both experts, and the head of the active domain, with gradients clipped to norm 1.0. The interleave ratio is 1:1 by default. The thinning factors ρ_{gen} and ρ_{comp} in Algorithm 1 take a task's gradient only every ρ steps, which is how the interleave ablation of Appendix H is realized. The three reference variants are the same loop with branches disabled: *Gen.Only* takes no comprehension step, *Comp.Only* takes no generation step, and *Naive* routes both through one adapter.

Adaptation. Adaptation restores the pre-trained checkpoint, initializes a fresh target-domain head, and runs the interleaved objectives for $E_2 = 500$ steps. The backbone, token embedding, and output projection remain frozen, while the two experts and the new head receive gradients. Applying rank-32 experts to the attention and MLP projections gives 1.18M parameters per expert and 2.36M in total. The complete model contains 13.3–13.5M parameters depending on the vocabulary size, so adaptation updates 17.5–17.8% of the model. The optimizer is reinitialized at the beginning of this stage.

Freezing the backbone preserves the representation acquired during multi-domain pre-training while allowing the objective-specific paths to specialize. Adaptation uses a k-shot support set with k

Algorithm 1: Pre-training pipeline of UNICG.

Input: Domains $\mathcal{D} = \{\mathcal{D}_1, \ldots, \mathcal{D}_M\}$, schema inventory $\mathcal{S}_{\text{cfg}}$, serializer $\mathcal{T}$, RVQ configuration (L, K), backbone $f_{\boldsymbol{\theta}}$, experts $\boldsymbol{W}_{\text{gen}}, \boldsymbol{W}_{\text{comp}}$, heads $g_{\boldsymbol{\phi}}$, steps E_1, batch size B, learning rate η, and interleave factors $\rho_{\text{gen}}, \rho_{\text{comp}}$.

Output: Pre-trained backbone $\boldsymbol{\theta}^\star$, experts $\boldsymbol{W}^\star_{\text{gen}}, \boldsymbol{W}^\star_{\text{comp}}$, heads $\boldsymbol{\phi}^\star$.

```
// Construct the vocabulary once before training
```
Fit RVQ codebooks $\{\mathcal{C}_\ell\}_{\ell=1}^{L}$ by k-means on node features pooled over $\mathcal{D}$
Build $\mathcal{V}_{\text{tok}}$ from $\mathcal{S}_{\text{cfg}}$, including held-out-domain schema under Uni-Unseen
foreach $\mathcal{G}$ *in every domain* **do**
    $\boldsymbol{s} \leftarrow \mathcal{T}(\mathcal{G})$ using canonical order π `// in Eq. (1)`
    if $\mathcal{T}^{-1}(\boldsymbol{s}) \neq \mathcal{G}$ **then** discard $\boldsymbol{s}$ `// verify invertibility`
end
Initialize $\boldsymbol{\theta}, \boldsymbol{W}_{\text{gen}}, \boldsymbol{W}_{\text{comp}}, \boldsymbol{\phi}$ randomly
```
// Interleave domain and task updates
```
for $t = 1, 2, \ldots, E_1$ **do**
    $d \leftarrow \mathcal{D}[t \bmod M]$ `// select domains round-robin`
    Zero gradients
    if $t \bmod \rho_{gen} = 0$ **then**
        SETEXPERT($\boldsymbol{W}_{\text{gen}}$)
        Sample a graph-sequence batch $\mathcal{B} \sim \mathcal{C}_d$
        Compute $\mathcal{L}_{\text{gen}} \leftarrow -\sum_t \log p_{\boldsymbol{\theta}}(s_t \mid s_{<t})$ on $\mathcal{B}$
        Backpropagate $\mathcal{L}_{\text{gen}}$
    end
    if $t \bmod \rho_{comp} = 0$ **then**
        SETEXPERT($\boldsymbol{W}_{\text{comp}}$)
        Sample a labelled batch $\mathcal{B}' \sim \mathcal{C}_d$
        $\bar{\boldsymbol{h}} \leftarrow$ mask-mean-pool of final hidden states of $\mathcal{B}'$
        Compute $\mathcal{L}_{\text{comp}} \leftarrow$ CE or MSE$\big(g_{\boldsymbol{\phi}_d}(\bar{\boldsymbol{h}}),\, \boldsymbol{y}\big)$
        Backpropagate $\mathcal{L}_{\text{comp}}$
    end
    Clip gradients to norm 1.0
    Apply one AdamW step to $\{\boldsymbol{\theta}\} \cup \{\boldsymbol{W}_{\text{gen}}, \boldsymbol{W}_{\text{comp}}\} \cup \{\boldsymbol{\phi}_d\}$
end

labelled examples per class for classification and $10k$ labelled graphs for regression. A shared random seed gives every method the same support set. Episodes index independent support-set draws, and the reported mean±std is taken over them.

The k-shot budget applies to the labelled comprehension objective. Generation remains unsupervised and draws from the target training graphs. Few-shot results are reported under Uni-Unseen, where target labels are absent from pre-training. The *Comp.Only* reference is trained from scratch using the same k labels and support sets.

Generation at inference. Following the adaptation specified by the selected setting, generation requires no labels or additional generation-specific fitting. The generation expert is activated, sampling starts from <boc>, and tokens are drawn autoregressively until <eog>, after which $\mathcal{T}^{-1}$ decodes the sequence into a graph. Every sampled sequence contributes to evaluation, with decoding failures counted as invalid. Controllable generation prepends $\texttt{COND}_k$ to the prompt and is otherwise identical.

Constrained decoding. The grammar mask defines the legal token set at each position and renormalizes the next-token distribution over that set. An edge may reference only nodes declared in the node block, the node block contains at least two nodes, and an edge is completed before the edge block closes. Domain-specific validity is then evaluated after decoding using Table C.2. This procedure gives token sequences the syntactic well-formedness that adjacency-based generators obtain from their matrix representation. On the node-level corpora, the fraction of decodable sequences increases from 0.42 to 0.93. We use the full 1024-token context so that the serialized Arxiv graphs are

Algorithm 2: Few-shot adaptation pipeline of UNICG.

Input: Pre-trained $\boldsymbol{\theta}^\star, \boldsymbol{W}^\star_{\text{gen}}, \boldsymbol{W}^\star_{\text{comp}}$, target domain $\mathcal{D}_m$, label budget k, episodes E, adaptation steps E_2, and batch size B.

Output: Comprehension metric on the held-out test split of $\mathcal{D}_m$, averaged over episodes.

// Run generation for UNICG and *Naive*
// Skip adaptation for *Gen.Only* as specified in Appendix C.3

for $e = 1, 2, \ldots, E$ **do**
  Restore $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta}^\star$, $\boldsymbol{W}_{\text{gen}} \leftarrow \boldsymbol{W}^\star_{\text{gen}}$, $\boldsymbol{W}_{\text{comp}} \leftarrow \boldsymbol{W}^\star_{\text{comp}}$
  Initialize a fresh head $g_{\boldsymbol{\phi}_m}$
  Freeze $\boldsymbol{\theta}$ and the embedding/output projection
  Keep $\boldsymbol{W}_{\text{gen}}, \boldsymbol{W}_{\text{comp}}, \boldsymbol{\phi}_m$ trainable // 2.36M of≈13.4M
  Reinitialize the AdamW state
  // Share each sampled support set across methods
  $\mathcal{S}_e \leftarrow \textsc{ShotPool}(\mathcal{D}_m, k, \text{seed} = e)$ from the training split only
  Compute label-normalization statistics from $\mathcal{S}_e$
  for $t = 1, 2, \ldots, E_2$ **do**
    $\textsc{SetExpert}(\boldsymbol{W}_{\text{gen}})$
    Sample $\mathcal{B} \sim \mathcal{C}_m$ from the unlabeled target training graphs
    Compute $\mathcal{L}_{\text{gen}}$ on $\mathcal{B}$ and backpropagate
    $\textsc{SetExpert}(\boldsymbol{W}_{\text{comp}})$
    Sample $\mathcal{B}' \sim \mathcal{S}_e$ with replacement when $|\mathcal{S}_e| < B$
    $\bar{\boldsymbol{h}} \leftarrow$ mask-mean-pool of final hidden states
    Compute $\mathcal{L}_{\text{comp}} \leftarrow$ CE or MSE$\big(g_{\boldsymbol{\phi}_m}(\bar{\boldsymbol{h}}),\, \boldsymbol{y}\big)$ and backpropagate
    Clip gradients to norm 1.0
    Apply one AdamW step to $\{\boldsymbol{W}_{\text{gen}}, \boldsymbol{W}_{\text{comp}}\} \cup \{\boldsymbol{\phi}_m\}$ // $\boldsymbol{\theta}$ remains frozen
  end
  Evaluate $g_{\boldsymbol{\phi}_m} \circ \bar{\boldsymbol{h}}_{\boldsymbol{\theta}}$ on the untouched test split of $\mathcal{D}_m$
end
Report mean±std over the E episodes

covered by the generation budget. Grammar masking is used for Tables 1 and E.5. Tables G.1 and G.2 use plain ancestral sampling.

Computational cost. Pre-training is E_1 steps, each a forward-backward pass over B sequences of length at most T through a 6-layer transformer of width h, giving $\mathcal{O}(E_1 B(T^2 h + T h^2))$ for a single objective and twice that at the 1:1 interleave, since generation and comprehension each take one pass. The two experts add $\mathcal{O}(E_1 BThr)$ with $r \ll h$, which is negligible against the backbone term, so unification roughly doubles the per-step cost of a single-task model while adding only a small parameter overhead. Vocabulary construction is $\mathcal{O}(NLKd)$ for k-means over N nodes of dimension d, done once and amortized over the whole run, and serialization is $\mathcal{O}(|\mathcal{V}| \log |\mathcal{V}| + |\mathcal{E}|)$ per graph, dominated by the degree sort. Adaptation is far cheaper at $E_2 = 500$ steps, and cheaper still per step, since the frozen backbone needs no parameter gradients and only the 2.36M expert parameters and the head are updated. Under Uni-Unseen one pre-training run serves every label budget, since the target's labels are never seen during pre-training, so the $1/5/10$-shot tables cost one pre-training plus three short adaptations rather than three full runs.

The default 1:1 interleave equally weights the two objectives through their update frequency. Separate experts isolate task-specific updates, while the backbone remains shared and can retain cross-task interactions. The interleave ablation measures the effect of changing this implicit weighting.

C EXPERIMENT DETAILS

This appendix consolidates the experimental specification. We first describe the datasets and baselines, then define the adaptation settings, metrics, and decoding procedure, and finally map each research question to its corresponding configuration and complete results.

Table C.1: Statistics of the graph domains used for comprehension and generation. $|\mathcal{V}|$ and $|\mathcal{E}|$ are corpus averages. #Out denotes the number of classes or regression targets, and RVQ marks continuously attributed domains.

Domain	Source / type	#Graphs	Avg. $\lvert\mathcal{V}\rvert$	Avg. $\lvert\mathcal{E}\rvert$	#Out	Metric	RVQ
Synthetic structure							
`Tree`	Random tree	400	19.4	18.4	5	R^2	–
`ER`	Erdős–Rényi	400	20.0	53.4	5	R^2	–
`BA`	Barabási–Albert	400	20.8	37.6	5	R^2	–
`SBM`	Stochastic block model	400	18.3	35.0	5	R^2	–
`Planar`	Random planar	400	19.1	46.7	5	R^2	–
Text-attributed node classification							
`Cora`	Citation	1500	18.4	27.1	7	Accuracy	✓
`CiteSeer`	Citation	1481	10.4	14.8	6	Accuracy	✓
`PubMed`	Citation	1500	23.1	30.8	3	Accuracy	✓
`WikiCS`	Wiki articles	1461	29.5	157.9	10	Accuracy	✓
`Amazon-Computers`	Co-purchase	1472	29.5	122.4	10	Accuracy	✓
`Amazon-Photo`	Co-purchase	1478	29.4	145.3	8	Accuracy	✓
`Arxiv`	Citation (OGB)	3000	25.8	63.1	40	Accuracy	✓
Molecule graph classification							
`BACE`	BACE-1 inhibition	1509	34.1	36.9	2	ROC-AUC	–
`BBBP`	Blood–brain barrier	1970	24.0	25.8	2	ROC-AUC	–
`HIV`	HIV replication	8000	23.2	24.6	2	ROC-AUC	–
Knowledge-graph link prediction							
`WN18RR`	WordNet	6000	11.1	13.1	2	ROC-AUC	–
`FB15k-237`	Freebase	6000	19.8	54.4	2	ROC-AUC	–
Controlled diagnostics							
`GraphFamily`	Generator ID	800	18.8	36.3	5	Accuracy	–
`CitSynth`	Synthetic citation	301	30.0	60.3	5	Accuracy	✓
`TextGraph`	Synthetic attributed	300	23.6	45.3	4	R^2	✓
Molecule generation							
`Molecule`	QM9-derived	8000	6.5	6.1	5	R^2	–

C.1 Dataset Details

The evaluation covers heterogeneous graph domains whose nodes, edges, attributes, prediction units, and output spaces differ substantially. This breadth tests whether one discrete interface can support both comprehension and generation without relying on domain-specific tokenizers.

Dataset coverage. The corpus spans synthetic structures, attributed citation and co-purchase networks, molecular graphs, knowledge graphs, and controlled synthetic diagnostics. Every graph is represented with the vocabulary defined in Appendix B.1. The same serializer is therefore used across all domains.

Dataset statistics. Table C.1 summarizes the source, graph scale, output space, evaluation metric, and use of residual quantization for each domain. The collection ranges from small molecular graphs to attributed ego-graphs with substantially denser connectivity, providing variation in both sequence length and semantic content.

Domain-specific construction.

- **Synthetic structures** include `Tree`, `ER`, `BA`, `SBM`, and `Planar`. The generators comprise uniform random trees, Erdős–Rényi graphs (Erdős & Rényi, 1960), Barabási–Albert graphs (Barabási & Albert, 1999), stochastic block models (Holland et al., 1983), and random planar graphs. Each graph is reduced to its largest connected component. Comprehension predicts node count, edge count, density, triangle count, and average clustering using R^2, while generation evaluates the corresponding structural validity and distributional statistics.

- **Attributed node-classification graphs** include `Cora`, `CiteSeer`, `PubMed`, `WikiCS`, `Amazon-Computers`, `Amazon-Photo`, and `Arxiv` (Sen et al., 2008; Yang et al., 2016; Mernyei & Cangea, 2020; Shchur et al., 2018; Hu et al., 2020a). For each labelled node, we extract a two-hop ego-graph, map the center to `IDX_0`, cap the graph size, quantize the dataset-

native node features into SEM tokens, and predict the center-node class. These domains share a task while differing in label space, feature space, and topology.

- **Molecular property graphs** include BACE, BBBP, and HIV from MoleculeNet (Wu et al., 2018). Molecules are kekulized, hydrogens are removed, and atoms and bonds retain their discrete types. Binary assay outcomes are evaluated by ROC-AUC, which remains informative for imbalanced datasets such as HIV.
- **Knowledge graphs** include WN18RR (Dettmers et al., 2018) and FB15k-237 (Toutanova & Chen, 2015). For a candidate pair (h, t), we extract the enclosing subgraph, encode typed edges with REL_r, and prepend the query relation. True triples form positive examples, while corrupted triples form negative examples for binary link prediction.
- **Molecular generation graphs** include QM9 (Ramakrishnan et al., 2014) and ZINC250k (Irwin et al., 2012). These de-novo design datasets are evaluated using chemical validity, valence validity, uniqueness, novelty, and distributional statistics.
- **Controlled diagnostics** include GraphFamily, CitSynth, and TextGraph. GraphFamily predicts the generator that produced a graph. CitSynth and TextGraph control the relationship between node features and labels, isolating the effect of semantic quantization.

Task coverage. The datasets instantiate five complementary task types.

- **Structure regression** measures whether serialized graphs retain global structural statistics.
- **Node classification** measures whether local semantic information survives feature quantization.
- **Graph classification** evaluates graph-level predictive information for molecular assays.
- **Link prediction** evaluates relational information between entity pairs.
- **Graph generation** measures validity, uniqueness, novelty, and distributional fidelity after decoding.

The first four tasks evaluate comprehension, while the fifth evaluates generation. V.U.N. combines validity, uniqueness, and novelty, complementing MMD by distinguishing new samples from memorized training graphs.

C.2 Baseline Details

We compare UniCG with **13** baselines spanning task-specific graph networks, comprehension-oriented graph foundation models, and graph generators. The three groups evaluate complementary aspects of the proposed framework.

- **Task-specific graph neural networks.** GCN (Kipf & Welling, 2017) performs spectral message passing with normalized neighborhood aggregation. GAT (Veličković et al., 2018) replaces fixed aggregation weights with learned attention coefficients. Both models are trained directly on each target task and provide conventional graph-learning references without multi-domain pre-training.
- **Comprehension-oriented graph foundation models.** SAMGPT (Yu et al., 2025) aligns domains through feature and structure prompts. MDGPT (Yu et al., 2024) learns cross-domain prompts followed by target adaptation. MDGFM (Wang et al., 2025) introduces topology-aware alignment for multi-domain pre-training. GCOPE (Zhao et al., 2024) connects disjoint graph domains through coordinator nodes. GFT (Wang et al., 2024) constructs transferable tree tokens through vector quantization. UniGraph2 (He et al., 2025) applies expert routing to masked-feature reconstruction. BRIDGE (Yuan et al., 2025) combines domain-specific feature alignment with mixture-of-experts prompting. These methods evaluate whether the shared representation of UniCG retains competitive predictive information across domains.
- **Graph generation models.** The Erdős–Rényi reference samples graphs using the empirical edge density of each domain. G2PT (Chen et al., 2025) provides the closest autoregressive comparison by modeling graph token sequences with a separate vocabulary and model for each domain. DiGress (Vignac et al., 2023) performs discrete denoising with a marginal transition process and reverse posterior. DeFoG (Qin et al., 2025) generates discrete graphs through continuous-time flow matching and rate-matrix sampling. These baselines compare unified autoregressive generation with per-domain autoregressive, diffusion, and flow-based alternatives.

Relation to Latent Graph Diffusion. Latent Graph Diffusion (LGD) (**?**) provides an important complementary view of unification. It casts regression and classification as conditional generation and performs denoising in a continuous graph latent space. Its empirical setting differs from ours because LGD trains its encoder, diffusion model, and task-specific decoder separately for each task, while cross-task pre-training is presented as a future extension. In contrast, our evaluation asks one shared multi-domain model to perform direct graph comprehension and unconditional graph generation over the same corpora under matched adaptation settings. These differences in task formulation and model sharing make LGD unsuitable as a directly comparable graph-generation baseline for numerical evaluation. We therefore discuss LGD as a related unified generative formulation rather than including it in the comparison tables.

Evaluation consistency. All baselines use the same dataset splits, target-domain support sets, and task metrics as UniCG. The reproduced DiGress and DeFoG variants share the same denoiser architecture, data pipeline, and optimization budget, allowing their transition mechanisms to be compared under matched conditions. G2PT retains its per-domain training design, which distinguishes specialized autoregressive modeling from the shared multi-domain parameterization of UniCG.

C.3 Adaptation Settings

Our evaluation varies two factors. Domain exposure determines whether target-domain graphs participate in multi-domain pre-training, while the label budget determines how much target supervision is available during adaptation. We study the following settings.

- **Uni-Seen with full-data adaptation.** The target domain participates in multi-domain pre-training. The resulting checkpoint is then adapted on the complete target training split, evaluating both comprehension and generation on domains represented during pre-training.
- **Uni-Unseen with full-data adaptation.** The target domain is removed from multi-domain pre-training. The source-domain checkpoint is then adapted on the complete target training split, measuring transfer to a new graph domain when target data become available only during adaptation.
- **Uni-Unseen with few-shot adaptation.** The target domain is excluded from pre-training and comprehension receives $k \in \{1, 5, 10\}$ labelled examples per class. Structural regression uses $10k$ labelled graphs. Generation remains unsupervised and uses the unlabeled target training graphs.

Target-domain exposure. The shared vocabulary is provisioned from the schema inventories of all configured domains, including the held-out target. Target graphs do not enter Uni-Unseen pre-training. The setting is therefore transductive in schema and inductive in graph data. During adaptation, UniCG and *Naive* use unlabeled target graphs for generation, while comprehension follows the selected label budget. The reported *Gen.Only* and per-domain G2PT rows do not use target adaptation and consequently measure zero-shot generation under a different exposure setting.

Data splits. Every domain uses a deterministic $80/20$ train/test split with seed 12345. Comprehension is evaluated on held-out test graphs. Generation draws new samples and compares their statistics and novelty with the same test split. Few-shot support sets are sampled from the training split and are shared across methods for each random seed.

C.4 Metrics and Decoding

The evaluation combines predictive metrics for comprehension with validity, novelty, diversity, and distributional metrics for generation. All methods use the same metric implementation within each dataset.

Comprehension metrics. We report accuracy for multi-class node and graph classification, ROC-AUC for molecular classification and link prediction, and R^2 for structural regression. ROC-AUC is used for binary tasks with imbalanced class distributions, including the approximately 3% positive rate of `HIV`.

Generation metrics. V.U.N. is the fraction of generated graphs that are simultaneously *valid*, *unique* within the sampled set, and *novel* with respect to the training graphs. Validity follows the domain-specific predicates in Table C.2. We also report maximum mean discrepancy between generated and held-out graphs. The reported MMD averages degree, clustering-coefficient, and spectral descriptors.

Four-node orbit counts are included in the released per-descriptor results and excluded from the average because enumeration is capped on larger corpora.

For molecule generation, validity denotes successful RDKit parsing. Valence validity measures the fraction of atoms whose total bond order remains within the allowed valence. Uniqueness and novelty follow the definitions used for the other graph domains.

Decoding settings. Autoregressive generation uses ancestral sampling at temperature 1.0 without beam search or nucleus filtering. Tables 1 and E.5 apply the serialization grammar described in Appendix B.5 to mask illegal token continuations. The mask constrains sequence syntax, while domain validity is evaluated after decoding. Every sampled sequence contributes to the metrics, with undecodable or domain-invalid outputs counted as invalid. Tables G.1 and G.2 use plain ancestral sampling.

Table C.2: Validity criteria for each generation domain. The same criterion is applied to UNICG and every baseline within a dataset.

Domain group	Validity criterion	Definition source
`Tree`	Tree structure	Generating process
`Planar`	Planar structure	Generating process
`SBM`	Consistent block structure	Generating process
`ER`, `BA`	Connected graph	Corpus construction
Ego-graphs (`Cora`, `PubMed`, `Arxiv`, and others)	Connected graph	Ego-graph extraction
Property molecules (`BACE`, `BBBP`, `HIV`)	Connected graph	Molecular graph construction
Knowledge graphs (`WN18RR`, `FB15k-237`)	Decodable graph	Typed relational serialization
`QM9`, `ZINC250k`	Chemically valid graph	RDKit validation

Evaluation on shared corpora. Comprehension reads held-out graphs from a corpus, while generation produces new graphs and compares them with the held-out split of the same corpus. The two capabilities are therefore aligned at the corpus level even though their prediction targets and metrics differ.

C.5 EXPERIMENT-SPECIFIC SETTINGS

This section maps each research question to its setting and complete result tables.

RQ1: Unified Comprehension and Generation. Table 1 compares UNICG with nine comprehension baselines and two graph generators on seven representative corpora under Uni-Seen. Appendix E extends the comparison to all domains and reports the Uni-Unseen counterpart.

RQ2: Few-shot Comprehension. The 1-, 5-, and 10-shot experiments use Uni-Unseen pre-training and identical support sets across methods. The main paper reports the representative 5-shot table, while Appendix F contains every task family and budget.

RQ3: Architectural Ablations. Table 3 compares *Gen.Only*, *Comp.Only*, *Naive*, and UNICG under matched data, splits, training steps, and hyperparameters. Appendix H adds the frozen-backbone diagnostic.

RQ4: Hyperparameter Sensitivity. The LoRA grid varies rank r and scaling factor α while holding the remaining settings fixed. The vocabulary sweeps vary RVQ depth L and codebook size K. Representative comprehension results appear in Section 5.5, while Appendix H reports the remaining comprehension and generation surfaces.

Dedicated generation baselines and transfer. Appendix G compares UNICG with the ER reference, G2PT, DiGress, and DeFoG on structural and molecular generation. Uni-Seen rows share target training data, while Uni-Unseen rows follow the target-exposure settings in Appendix C.3. G2PT provides the autoregressive per-domain reference.

Controllable and molecular generation. For controllable generation, token `COND_`k identifies one of three density bins and defines $p(\mathcal{G} \mid \texttt{COND}_k)$. At evaluation, realized graph density determines whether the requested condition is satisfied. Molecular generation on `QM9` and `ZINC250k` uses the metric suite above and compares with DiGress, DeFoG, and the ER reference.

D Implementation Details

This appendix reports the implementation choices needed to reproduce UniCG and the baselines, followed by the hardware and software environment used for all experiments.

D.1 Implementation Details of UniCG

The following configuration is shared by UniCG and its architectural variants unless stated otherwise.

Model architecture. The decoder-only transformer contains six layers, six attention heads, hidden width 384, and context length 1024. The backbone has 10.65M parameters and is initialized from scratch. Each task expert is a rank-32 LoRA module with scaling factor 64. The low-rank updates are applied to the combined attention projection, the attention output projection, and the MLP expansion projection. The complete model contains approximately 13.4M parameters, of which 2.36M are updated during target-domain adaptation.

Vocabulary and graph construction. The shared vocabulary uses three RVQ levels with 64 entries per level. Attributed node tasks use two-hop ego-graphs capped at 30 nodes. Canonical serialization and constrained decoding use a maximum sequence length of 1024. Section 5.5 and Appendix H report the vocabulary-depth and codebook-size sweeps.

Optimization. All variants use AdamW with learning rate 2×10^{-4}, batch size 64, and gradient clipping at 1.0. Multi-domain pre-training runs for 8000 steps with a 1:1 generation-to-comprehension interleave. Target-domain adaptation runs for 500 steps with the backbone, token embedding, and output projection frozen. The architecture, optimization budget, and data splits remain fixed across the architectural variants.

Generation configuration. We sample 300–500 graphs per domain at temperature 1.0. Controllable generation partitions graph density into three quantile bins computed over the pooled structure-domain corpus and draws 128 samples for each condition. UniCG and the reproduced generation baselines use identical sample counts and evaluation functions.

D.2 Implementation Details of Baselines

We provide the baseline implementations with their respective licenses as follows.

- `GCN` (Kipf & Welling, 2017): [MIT License] https://github.com/pyg-team/pytorch_geometric.
- `GAT` (Veličković et al., 2018): [MIT License] https://github.com/pyg-team/pytorch_geometric.
- `SAMGPT` (Yu et al., 2025): [License Unspecified] https://github.com/blue-soda/SAMGPT.
- `MDGPT` (Yu et al., 2024): [License Unspecified] https://anonymous.4open.science/r/MDGPT.
- `MDGFM` (Wang et al., 2025): [License Unspecified] https://github.com/wbkzwqtzw/MDGFM.
- `GCOPE` (Zhao et al., 2024): [License Unspecified] https://github.com/cshhzhao/GCOPE.
- `GFT` (Wang et al., 2024): [MIT License] https://github.com/Zehong-Wang/GFT.
- `UniGraph2` (He et al., 2025): [MIT License] https://github.com/yf-he/UniGraph2.
- `BRIDGE` (Yuan et al., 2025): [MIT License] https://github.com/RingBDStack/BRIDGE.
- `G2PT` (Chen et al., 2025): [MIT License] https://github.com/tufts-ml/G2PT.
- `DiGress` (Vignac et al., 2023): [MIT License] https://github.com/cvignac/DiGress.

- DeFoG (Qin et al., 2025): [MIT License]
  https://github.com/manuelmlmadeira/DeFoG.

For all baselines, we follow the configurations recommended by the original papers or public implementations when they are available. We otherwise tune the learning rate and task-specific optimization settings on the validation split. All methods use the same data splits, support sets, and evaluation functions. The Erdős–Rényi reference is implemented directly from the empirical node-count and edge-density distributions and therefore has no external implementation dependency.

D.3 Hardware and Software Configurations

All experiments are conducted with the following hardware and software environment.

- **Operating system:** Ubuntu 20.04 LTS.
- **CPU:** Intel Xeon Platinum 8358 at 2.60 GHz with 1 TB DDR4 memory.
- **GPU:** NVIDIA Tesla A100 SXM4 with 80 GB memory.
- **Software:** CUDA 12.1, Python 3.11.15, PyTorch[D.1] 2.5.1, PyTorch Geometric[D.2] 2.7.0, Transformers 4.44.2, PEFT 0.11.1, and RDKit 2026.3.2.

E Additional Results: Uni-Seen and Uni-Unseen Evaluation

The main paper's Table 1 reports the **Uni-Seen full-data** setting. Every evaluation domain participates in pre-training, followed by adaptation on its complete 80% training split. This appendix extends the comprehension results across all 20 domains and provides the corresponding **Uni-Unseen full-data** results. Together with Appendix F, the experiments cover the domain-exposure and label-budget axes defined in Appendix C.3. Full-data adaptation tests whether the unified representation remains effective when target supervision is abundant and separates the effect of domain exposure from that of label scarcity.

All numbers are the mean over **five seeds**, with the standard deviation reported in each table. The held-out 20% test split is identical to that used for few-shot adaptation, making the two label-budget regimes directly comparable. Tables E.1–E.4 present paired Uni-Seen and Uni-Unseen blocks for every task family.

E.1 Full-Data Adaptation on Seen Domains

Node, graph, and link prediction. Under Uni-Seen adaptation, UniCG remains competitive with specialized comprehension baselines across all three prediction levels. It obtains the strongest results on Amz-Photo, BACE, and HIV, while ranking close to the leading method on the remaining real domains. The *Naive* variant is particularly strong on node classification and knowledge-graph link prediction, showing that direct parameter sharing can favor predictive representations when target-domain supervision is abundant. This advantage does not extend to generation, as shown by the ablation results in Section 5.4.

Synthetic classification and structural regression. The synthetic domains reveal a clearer distinction between task families. Performance on GraphFam remains challenging for all joint variants, while UniCG is competitive on CitSynth. In contrast, UniCG achieves the best or tied-best R^2 across all five structural regression domains. These results indicate that the shared graph language captures continuous structural properties reliably even when synthetic graph-family labels remain difficult to separate.

Generation fidelity on the shared corpora. Table E.5 complements the V.U.N. results in Table 1 with distributional fidelity. V.U.N. and MMD exhibit different preferences on WN18RR. Its permissive validity predicate rewards diverse decoded graphs, whereas MMD penalizes departures from the reference structural distribution. Reporting both measures therefore distinguishes syntactic and sample-level quality from distributional agreement.

[D.1] https://github.com/pytorch/pytorch
[D.2] https://github.com/pyg-team/pytorch_geometric

Table E.1: Accuracy (%, mean±std over five runs) of node classification under Uni-Seen and Uni-Unseen. Best results are presented in **bold**, and runner-ups are underlined.

Method	Cora	CiteSeer	PubMed	WikiCS	Amz-Comp	Amz-Photo	Arxiv
Uni-Seen (all domains seen in pre-training)							
GCN [11]	26.72 ±0.33	24.84 ±1.05	36.00 ±1.01	37.18 ±0.28	44.86 ±0.80	53.80 ±0.50	30.00 ±0.32
GAT [22]	29.97 ±0.62	23.03 ±0.94	36.07 ±1.04	36.42 ±1.01	44.14 ±0.87	53.72 ±0.27	35.65 ±0.92
SAMGPT [32]	52.27 ±4.44	45.99 ±4.67	68.13 ±4.26	44.09 ±6.31	62.03 ±4.35	73.28 ±6.63	35.44 ±2.49
MDGPT [31]	36.23 ±4.76	42.76 ±3.68	62.47 ±6.41	36.28 ±7.13	56.98 ±15.66	68.45 ±6.07	33.28 ±3.96
MDGFM [24]	45.23 ±5.68	40.60 ±8.25	63.23 ±5.70	37.34 ±5.39	58.48 ±7.78	64.80 ±6.65	27.90 ±1.81
GCOPE [34]	54.43 ±2.69	53.10 ±2.36	64.67 ±3.09	47.10 ±2.71	69.69 ±1.78	74.73 ±4.12	50.20 ±1.27
GFT [25]	29.33 ±0.00	17.98 ±1.50	36.80 ±1.06	22.18 ±0.00	37.29 ±0.00	24.32 ±0.00	14.17 ±1.96
UniGraph2 [5]	60.17 ±0.91	**59.89 ±2.45**	**75.96 ±0.93**	56.61 ±3.28	63.09 ±8.99	72.76 ±6.16	51.27 ±3.14
BRIDGE [33]	55.17 ±1.31	57.64 ±1.84	69.57 ±1.65	55.32 ±1.70	72.03 ±2.02	77.74 ±1.87	52.97 ±1.59
Comp.Only	63.47 ±1.76	54.68 ±1.42	71.40 ±1.68	69.01 ±1.46	80.34 ±1.41	88.99 ±1.29	50.67 ±1.19
Naive	**64.87 ±1.69**	57.78 ±0.94	73.73 ±1.27	**71.60 ±1.11**	**81.29 ±2.62**	89.73 ±0.79	**53.50 ±0.33**
UniCG	64.47 ±0.62	55.69 ±1.25	72.07 ±1.82	69.62 ±1.43	80.00 ±2.50	**90.61 ±1.03**	52.60 ±1.33
Uni-Unseen (target domain held out)							
GCN [11]	26.67 ±0.37	24.45 ±1.27	35.83 ±0.81	37.13 ±0.31	45.02 ±0.78	53.92 ±0.48	30.04 ±0.28
GAT [22]	29.97 ±0.62	23.06 ±0.91	36.05 ±1.08	36.50 ±1.06	44.01 ±0.84	53.62 ±0.27	35.42 ±1.02
SAMGPT [32]	50.44 ±2.67	43.66 ±3.70	68.88 ±3.82	41.71 ±5.77	60.11 ±4.11	73.37 ±5.07	34.98 ±2.43
MDGPT [31]	41.78 ±4.63	45.96 ±3.86	62.20 ±7.48	39.11 ±5.82	56.69 ±12.88	62.84 ±7.55	31.23 ±3.79
MDGFM [24]	44.33 ±6.88	43.50 ±4.28	63.05 ±6.07	34.06 ±2.24	62.90 ±8.27	69.46 ±4.91	29.86 ±1.43
GCOPE [34]	45.00 ±3.40	43.10 ±1.82	49.83 ±2.81	36.86 ±1.28	46.97 ±4.79	52.50 ±4.01	30.17 ±1.50
GFT [25]	29.33 ±0.00	19.86 ±1.35	36.80 ±1.06	22.18 ±0.00	37.29 ±0.00	24.32 ±0.00	13.13 ±1.39
UniGraph2 [5]	53.47 ±5.09	49.76 ±3.79	59.40 ±3.17	42.59 ±3.30	48.17 ±4.67	54.86 ±6.77	31.43 ±1.10
BRIDGE [33]	50.03 ±1.86	47.75 ±1.90	54.73 ±2.56	41.52 ±4.61	55.29 ±2.49	63.61 ±4.89	32.91 ±1.86
Comp.Only	63.67 ±0.76	54.68 ±1.76	69.87 ±2.42	67.78 ±1.07	**81.22 ±1.80**	89.66 ±2.32	51.43 ±1.82
Naive	**63.87 ±0.75**	**56.64 ±2.25**	**72.93 ±2.16**	**71.13 ±1.34**	80.20 ±1.91	**90.54 ±1.07**	**51.50 ±1.32**
UniCG	62.67 ±1.73	56.23 ±1.20	71.00 ±1.17	69.90 ±1.63	80.07 ±1.01	88.72 ±0.46	50.13 ±0.88

Table E.2: ROC-AUC (%, mean±std over five runs) of graph classification and link prediction under Uni-Seen and Uni-Unseen. Best results are presented in **bold**, and runner-ups are underlined.

Task	Graph Classification			Link Prediction	
Method	BACE	BBBP	HIV	WN18RR	FB15k-237
Uni-Seen (all domains seen in pre-training)					
GCN [11]	52.79 ±0.08	62.12 ±0.04	55.15 ±0.26	62.99 ±0.30	56.91 ±0.30
GAT [22]	52.31 ±0.10	61.87 ±0.11	54.89 ±0.22	63.71 ±0.15	58.69 ±0.97
SAMGPT [32]	59.86 ±1.79	70.24 ±0.99	54.60 ±6.95	60.29 ±1.86	58.82 ±0.42
MDGPT [31]	57.11 ±3.45	70.02 ±0.45	54.68 ±13.18	62.53 ±1.79	59.15 ±0.98
MDGFM [24]	51.93 ±2.64	66.43 ±3.48	58.56 ±9.97	53.48 ±5.47	51.99 ±5.49
GCOPE [34]	61.62 ±0.35	**73.35 ±0.39**	59.24 ±1.65	71.68 ±0.77	65.57 ±0.42
GFT [25]	52.02 ±3.95	65.78 ±1.18	57.95 ±7.19	60.58 ±4.87	44.81 ±2.95
UniGraph2 [5]	60.89 ±0.73	70.99 ±1.58	65.19 ±2.40	67.71 ±2.64	63.49 ±2.09
BRIDGE [33]	61.08 ±0.84	72.16 ±0.56	65.49 ±1.97	72.91 ±0.81	65.63 ±1.01
Comp.Only	62.73 ±1.24	67.29 ±1.80	71.37 ±2.62	92.95 ±0.85	85.38 ±0.55
Naive	61.64 ±1.29	67.54 ±0.49	71.28 ±1.40	**97.37 ±0.27**	**93.49 ±0.28**
UNICG	**65.13 ±2.59**	67.87 ±1.93	**71.97 ±2.61**	95.66 ±0.70	88.61 ±1.57
Uni-Unseen (target domain held out)					
GCN [11]	52.80 ±0.09	62.11 ±0.05	55.09 ±0.23	62.97 ±0.27	56.92 ±0.29
GAT [22]	52.27 ±0.09	61.87 ±0.10	54.90 ±0.22	63.71 ±0.17	58.68 ±0.92
SAMGPT [32]	59.53 ±1.62	68.57 ±2.15	51.03 ±9.04	60.17 ±2.15	58.57 ±0.14
MDGPT [31]	57.57 ±3.52	69.81 ±0.93	59.27 ±10.52	61.49 ±2.60	54.75 ±5.27
MDGFM [24]	53.99 ±4.13	63.77 ±6.76	54.12 ±6.14	53.30 ±4.41	52.84 ±4.75
GCOPE [34]	60.91 ±0.53	**73.02 ±0.86**	61.83 ±1.69	71.59 ±0.29	64.88 ±1.08
GFT [25]	52.66 ±3.15	58.85 ±12.71	55.57 ±7.58	55.84 ±7.95	49.13 ±5.27
UniGraph2 [5]	61.19 ±0.45	70.67 ±0.63	59.36 ±5.34	68.38 ±2.85	61.11 ±1.30
BRIDGE [33]	59.97 ±1.64	71.35 ±0.74	63.94 ±2.27	73.06 ±0.87	63.53 ±0.82
Comp.Only	**62.30 ±3.69**	64.30 ±3.28	69.04 ±3.08	92.69 ±0.61	84.23 ±0.37
Naive	60.74 ±2.60	70.91 ±1.83	72.51 ±1.55	**97.22 ±0.31**	**91.24 ±0.63**
UNICG	61.03 ±3.87	68.72 ±2.03	**74.81 ±1.88**	93.36 ±1.29	85.92 ±3.12

Table E.3: Accuracy (%, mean±std over five runs) of synthetic graph and node classification under Uni-Seen and Uni-Unseen. Best results are presented in **bold**, and runner-ups are underlined.

Uni-Seen (all domains seen in pre-training)

Method	GraphFam	CitSynth
GCN [11]	35.75 ±2.07	43.11 ±1.23
GAT [22]	37.12 ±7.21	38.19 ±0.65
SAMGPT [32]	43.37 ±4.16	76.07 ±3.96
MDGPT [31]	42.12 ±1.46	60.00 ±11.76
MDGFM [24]	44.88 ±2.07	58.36 ±4.93
GCOPE [34]	38.25 ±13.16	56.89 ±2.22
GFT [25]	40.25 ±0.75	64.59 ±0.00
UniGraph2 [5]	50.00 ±4.79	65.08 ±6.36
BRIDGE [33]	**51.88 ±2.85**	55.41 ±4.70
Comp.Only	40.00 ±1.43	88.85 ±1.91
Naive	38.75 ±2.40	**93.11 ±2.82**
UNICG	41.00 ±1.84	91.14 ±1.67

Uni-Unseen (target domain held out)

Method	GraphFam	CitSynth
GCN [11]	35.75 ±2.07	43.11 ±1.23
GAT [22]	37.87 ±7.40	38.19 ±0.65
SAMGPT [32]	43.12 ±2.20	73.77 ±3.44
MDGPT [31]	42.25 ±1.88	65.25 ±18.24
MDGFM [24]	48.13 ±4.71	68.85 ±6.56
GCOPE [34]	43.50 ±7.17	57.54 ±5.90
GFT [25]	40.25 ±0.75	64.59 ±0.00
UniGraph2 [5]	**51.13 ±2.22**	64.92 ±0.66
BRIDGE [33]	49.25 ±2.58	57.87 ±4.02
Comp.Only	43.62 ±3.32	**90.49 ±1.61**
Naive	40.62 ±3.33	88.03 ±5.04
UNICG	43.62 ±2.83	86.10 ±7.29

Table E.4: R^2 (mean±std over five runs) of graph-property regression under Uni-Seen and Uni-Unseen. Best results are presented in **bold**, and runner-ups are underlined.

Method	Tree	ER	BA	SBM	Planar
Uni-Seen (all domains seen in pre-training)					
GCN [11]	0.457 ±0.024	0.565 ±0.003	0.555 ±0.002	0.408 ±0.003	0.380 ±0.098
GAT [22]	0.440 ±0.017	0.571 ±0.003	0.561 ±0.002	0.418 ±0.003	0.400 ±0.047
SAMGPT [32]	0.374 ±0.005	0.234 ±0.119	0.247 ±0.117	0.085 ±0.066	-0.078 ±0.029
MDGPT [31]	0.372 ±0.002	0.184 ±0.093	0.195 ±0.089	0.061 ±0.036	-0.078 ±0.025
MDGFM [24]	0.370 ±0.003	0.113 ±0.041	0.144 ±0.065	0.027 ±0.024	-0.101 ±0.002
GCOPE [34]	0.393 ±0.005	0.495 ±0.038	0.547 ±0.017	0.335 ±0.037	0.114 ±0.067
GFT [25]	0.368 ±0.006	-0.004 ±0.001	-0.006 ±0.001	-0.014 ±0.005	-0.117 ±0.011
UniGraph2 [5]	0.397 ±0.027	0.299 ±0.171	0.357 ±0.212	0.175 ±0.126	-0.036 ±0.070
BRIDGE [33]	0.395 ±0.006	0.448 ±0.047	0.498 ±0.041	0.264 ±0.056	0.030 ±0.070
Comp.Only	**0.998 ±0.001**	0.858 ±0.012	0.922 ±0.004	0.708 ±0.015	0.826 ±0.023
Naive	0.997 ±0.001	0.850 ±0.012	0.924 ±0.002	0.667 ±0.009	0.822 ±0.002
UNICG	**0.998 ±0.000**	**0.865 ±0.009**	**0.934 ±0.006**	**0.711 ±0.019**	**0.830 ±0.010**
Uni-Unseen (target domain held out)					
GCN [11]	0.456 ±0.022	0.566 ±0.004	0.556 ±0.002	0.407 ±0.003	0.385 ±0.088
GAT [22]	0.438 ±0.018	0.570 ±0.003	0.561 ±0.002	0.418 ±0.003	0.392 ±0.044
SAMGPT [32]	0.371 ±0.005	0.244 ±0.121	0.239 ±0.114	0.076 ±0.040	-0.072 ±0.029
MDGPT [31]	0.373 ±0.003	0.153 ±0.067	0.163 ±0.059	0.050 ±0.042	-0.076 ±0.027
MDGFM [24]	0.368 ±0.003	0.129 ±0.057	0.191 ±0.103	0.018 ±0.011	-0.098 ±0.005
GCOPE [34]	0.391 ±0.010	0.488 ±0.038	0.556 ±0.006	0.362 ±0.031	0.142 ±0.040
GFT [25]	0.367 ±0.005	-0.004 ±0.000	-0.006 ±0.000	-0.012 ±0.003	-0.116 ±0.009
UniGraph2 [5]	0.382 ±0.015	0.377 ±0.136	0.331 ±0.148	0.196 ±0.128	-0.090 ±0.036
BRIDGE [33]	0.399 ±0.010	0.430 ±0.041	0.537 ±0.034	0.300 ±0.046	0.047 ±0.063
Comp.Only	**0.999 ±0.000**	0.862 ±0.008	**0.936 ±0.006**	0.698 ±0.036	0.828 ±0.010
Naive	0.995 ±0.001	0.847 ±0.010	0.920 ±0.003	0.649 ±0.019	0.814 ±0.008
UNICG	**0.999 ±0.000**	**0.865 ±0.008**	0.930 ±0.005	**0.718 ±0.026**	**0.829 ±0.017**

Table E.5: Generation MMD (mean±std over five runs, lower is better) on the Uni-Seen corpora in Table 1. Best results are presented in **bold**, and runner-ups are underlined.

Method	Cora	PubMed	Arxiv	BACE	BBBP	WN18RR	FB15k-237
DiGress [23]	0.222 ±0.044	0.126 ±0.047	0.258 ±0.049	0.440 ±0.068	0.428 ±0.062	0.061 ±0.031	0.174 ±0.040
DeFoG [15]	**0.079 ±0.023**	0.129 ±0.042	**0.072 ±0.020**	0.434 ±0.073	0.289 ±0.043	0.069 ±0.032	0.088 ±0.045
Gen.Only	0.083 ±0.008	**0.111 ±0.004**	0.130 ±0.014	**0.079 ±0.005**	**0.061 ±0.006**	0.013 ±0.003	**0.029 ±0.004**
Naive	0.140 ±0.005	0.201 ±0.010	0.177 ±0.013	0.228 ±0.018	0.207 ±0.012	0.055 ±0.007	0.115 ±0.010
UNICG	0.113 ±0.003	0.167 ±0.008	0.146 ±0.009	0.089 ±0.011	0.068 ±0.006	**0.011 ±0.004**	0.038 ±0.004

Table F.1: Accuracy (%, mean±std) of **1-shot node classification** under Uni-Unseen. Best results are presented in **bold**, and runner-ups are underlined.

Method	Cora	CiteSeer	PubMed	WikiCS	Amz-Comp	Amz-Photo	Arxiv
GCN [11]	22.47 ±8.17	22.19 ±5.52	32.93 ±5.23	24.13 ±4.56	37.12 ±8.96	35.68 ±7.51	16.27 ±5.05
GAT [22]	29.80 ±9.08	21.51 ±5.37	39.66 ±4.15	25.36 ±3.84	37.73 ±6.67	35.34 ±8.39	15.93 ±2.19
SAMGPT [32]	**48.47 ±4.86**	**39.43 ±5.04**	55.20 ±7.06	35.19 ±6.89	44.71 ±10.56	41.83 ±3.95	11.47 ±3.30
MDGPT [31]	46.33 ±7.63	35.99 ±4.03	50.47 ±2.97	37.37 ±8.92	**54.68 ±12.95**	50.00 ±6.30	11.80 ±3.38
MDGFM [24]	44.47 ±5.24	38.82 ±4.44	56.87 ±5.34	35.53 ±2.69	42.57 ±4.12	56.15 ±4.80	11.17 ±2.34
GCOPE [34]	32.13 ±4.86	21.55 ±2.63	47.53 ±3.95	22.77 ±6.45	35.59 ±15.51	42.36 ±4.65	14.60 ±6.34
GFT [25]	35.87 ±7.50	38.12 ±2.27	31.20 ±4.01	31.26 ±7.23	33.46 ±2.88	40.48 ±3.59	11.50 ±0.64
UniGraph2 [5]	31.87 ±2.78	30.61 ±4.29	44.93 ±2.40	39.08 ±2.50	46.82 ±10.27	65.68 ±3.24	20.97 ±5.86
BRIDGE [33]	36.40 ±5.33	33.16 ±3.38	**59.60 ±1.39**	41.60 ±5.16	47.49 ±13.21	60.27 ±3.05	20.17 ±6.93
Comp.Only	39.47 ±6.53	39.09 ±4.35	55.33 ±6.94	**45.43 ±5.45**	46.03 ±6.31	64.26 ±1.98	20.93 ±0.90
Naive	37.33 ±6.07	37.95 ±2.05	55.20 ±2.42	44.51 ±5.81	48.58 ±7.06	65.05 ±8.34	18.17 ±2.49
UniCG	34.54 ±2.77	37.21 ±8.18	52.80 ±5.95	45.12 ±2.07	49.08 ±7.34	**66.35 ±9.96**	**21.20 ±2.02**

E.2 Full-Data Adaptation to Unseen Domains

Transfer across real graph domains. Holding the target domain out of pre-training reduces the advantage of multi-domain methods most visibly on node classification. Nevertheless, UniCG remains close to its matched comprehension variants and retains strong graph-level and link-level prediction. It achieves the best result on HIV and ranks second on both knowledge graphs, demonstrating that a jointly learned representation can be adapted to unseen molecular and relational domains with full target supervision.

Transfer across synthetic graph domains. Structural regression transfers consistently, with UniCG ranking first on four domains and second on BA. Synthetic classification is less stable, especially on CitSynth, where both joint variants trail *Comp.Only*. The contrast identifies synthetic family classification as the clearest boundary of the current cross-domain representation rather than a general failure of full-data adaptation.

Interaction between domain exposure and objective sharing. The relative ordering of *Comp.Only*, *Naive*, and UniCG changes across domains, showing that domain shift and objective interference are distinct effects. The shared expert often favors comprehension, while the task-decoupled experts preserve a stronger balance between comprehension and generation. Complete Uni-Unseen generation results are reported in Appendix G.

F Additional Results: Few-Shot Comprehension

Few-shot adaptation evaluates transfer to an *unseen* domain under limited target supervision. As defined in Appendix C.3, every method adapts from the same support set containing k labelled examples per class, sampled by a shared random-number generator. Generation remains unsupervised and independent of the label budget, so this axis isolates the transfer of graph comprehension. Each cell reports the mean±std over episodes for UniCG and over seeds for the baselines. The held-out test split is identical to the full-data experiments in Appendix E.

F.1 One-Shot Adaptation to Unseen Domains

Prediction across real graph domains. With one labelled example per class, performance varies substantially across domains and the standard deviations are correspondingly larger. UniCG remains competitive across node, graph, and link prediction, with particularly strong results on Amz-Photo, Arxiv, HIV, and FB15k-237. These results show that the unified representation provides a useful starting point even under the most restrictive support set.

Synthetic classification and structural regression. Synthetic class labels remain sensitive to the selected support examples, while structural regression is markedly more stable. UniCG ranks first on three of the five regression domains and second on the remaining two, indicating that global structural statistics transfer more reliably than fine-grained graph-family decisions in the one-shot regime.

Table F.2: ROC-AUC (%, mean±std) of **1-shot graph classification and link prediction** under Uni-Unseen. Best results are presented in **bold**, and runner-ups are underlined.

Task	**Graph Classification**			**Link Prediction**	
Method	BACE	BBBP	HIV	WN18RR	FB15k-237
GCN [11]	55.80 ±5.59	68.33 ±1.09	56.48 ±7.17	52.32 ±9.38	55.09 ±6.29
GAT [22]	54.84 ±6.25	68.14 ±0.59	55.70 ±7.12	53.63 ±16.04	54.66 ±5.88
SAMGPT [32]	54.29 ±7.16	63.94 ±8.36	54.95 ±5.57	52.95 ±9.48	55.26 ±6.68
MDGPT [31]	54.07 ±7.56	63.64 ±9.39	56.71 ±5.97	53.87 ±10.21	55.46 ±6.55
MDGFM [24]	55.65 ±4.44	67.48 ±1.94	52.62 ±7.23	52.81 ±7.80	54.10 ±7.91
GCOPE [34]	55.90 ±1.76	63.15 ±8.20	51.10 ±9.37	52.25 ±8.93	55.31 ±7.19
GFT [25]	50.16 ±8.11	50.18 ±0.23	51.56 ±7.94	48.87 ±3.73	50.16 ±0.41
UniGraph2 [5]	54.02 ±2.72	65.06 ±4.96	57.71 ±2.82	**54.65 ±8.67**	56.64 ±8.13
BRIDGE [33]	56.41 ±2.29	68.29 ±2.64	55.25 ±5.80	52.29 ±8.29	55.43 ±6.93
Comp.Only	**58.25 ±8.24**	**68.71 ±4.81**	53.41 ±12.72	51.67 ±8.90	55.00 ±4.12
Naive	56.04 ±8.24	65.30 ±2.04	58.96 ±13.55	51.90 ±8.44	57.20 ±2.96
UNICG	57.03 ±4.81	65.99 ±2.20	**63.24 ±13.09**	54.04 ±8.25	**59.93 ±5.95**

Table F.3: Accuracy (%, mean±std) of **1-shot synthetic graph and node classification** under Uni-Unseen. Best results are presented in **bold**, and runner-ups are underlined.

Method	GraphFam	CitSynth
GCN [11]	32.87 ±6.94	47.71 ±6.51
GAT [22]	36.25 ±8.22	39.18 ±7.11
SAMGPT [32]	38.50 ±8.09	60.66 ±8.86
MDGPT [31]	38.50 ±7.56	52.79 ±8.70
MDGFM [24]	45.25 ±5.39	58.20 ±9.76
GCOPE [34]	40.00 ±0.79	54.10 ±6.67
GFT [25]	40.25 ±0.75	63.28 ±5.99
UniGraph2 [5]	45.25 ±6.38	63.44 ±7.52
BRIDGE [33]	42.75 ±4.52	68.03 ±11.47
Comp.Only	41.87 ±2.88	**70.98 ±13.07**
Naive	**45.75 ±1.99**	68.53 ±3.04
UNICG	42.13 ±1.16	67.30 ±8.96

Table F.4: R^2 (mean±std) of **1-shot structure-property regression** under Uni-Unseen. Best results are presented in **bold**, and runner-ups are underlined.

Method	Tree	ER	BA	SBM	Planar
GCN [11]	0.356 ±0.111	0.440 ±0.124	0.294 ±0.154	0.285 ±0.130	0.102 ±0.070
GAT [22]	0.348 ±0.111	0.442 ±0.112	0.313 ±0.155	0.292 ±0.131	0.099 ±0.060
SAMGPT [32]	0.311 ±0.117	0.051 ±0.041	-0.039 ±0.048	-0.075 ±0.071	-0.092 ±0.125
MDGPT [31]	0.313 ±0.116	0.021 ±0.018	-0.057 ±0.101	-0.093 ±0.069	-0.086 ±0.099
MDGFM [24]	0.306 ±0.124	0.003 ±0.026	-0.067 ±0.056	-0.093 ±0.046	-0.092 ±0.111
GCOPE [34]	0.317 ±0.115	0.189 ±0.084	0.165 ±0.067	0.051 ±0.101	-0.050 ±0.099
GFT [25]	0.306 ±0.122	-0.024 ±0.018	-0.120 ±0.067	-0.106 ±0.040	-0.102 ±0.123
UniGraph2 [5]	0.308 ±0.121	0.011 ±0.025	-0.049 ±0.072	-0.081 ±0.034	-0.095 ±0.122
BRIDGE [33]	0.329 ±0.117	0.133 ±0.046	0.170 ±0.109	0.012 ±0.085	-0.067 ±0.095
Comp.Only	**0.993 ±0.003**	**0.653 ±0.064**	0.656 ±0.177	0.529 ±0.078	0.697 ±0.068
Naive	0.947 ±0.016	0.551 ±0.108	0.559 ±0.118	0.514 ±0.054	0.772 ±0.026
UNICG	0.984 ±0.006	0.621 ±0.122	**0.737 ±0.074**	**0.560 ±0.068**	**0.798 ±0.029**

Table F.5: Accuracy (%, mean±std) of **5-shot node classification** under Uni-Unseen. Best results are presented in **bold**, and runner-ups are underlined.

Method	Cora	CiteSeer	PubMed	WikiCS	Amz-Comp	Amz-Photo	Arxiv
GCN [11]	28.33 ±2.00	25.32 ±5.09	35.20 ±3.47	25.63 ±4.61	32.98 ±5.18	39.73 ±5.07	12.93 ±3.53
GAT [22]	26.67 ±2.62	25.93 ±6.34	31.93 ±4.78	27.00 ±4.40	32.58 ±5.65	40.74 ±5.11	15.73 ±2.33
SAMGPT [32]	43.80 ±4.33	38.05 ±4.33	55.47 ±6.51	35.02 ±13.88	45.69 ±5.38	48.85 ±7.85	11.00 ±5.69
MDGPT [31]	41.47 ±4.16	38.65 ±3.69	50.53 ±4.27	36.32 ±7.74	51.12 ±7.96	53.98 ±8.96	13.10 ±2.37
MDGFM [24]	43.53 ±2.75	38.79 ±7.34	50.67 ±8.43	36.11 ±7.67	49.36 ±6.38	56.83 ±4.10	13.37 ±5.10
GCOPE [34]	30.27 ±5.50	29.29 ±4.93	40.83 ±5.71	23.45 ±5.21	36.31 ±8.82	45.81 ±7.17	18.08 ±6.47
GFT [25]	35.50 ±8.35	32.66 ±6.27	36.00 ±1.33	34.06 ±6.07	31.32 ±13.20	44.19 ±6.23	14.46 ±4.41
UniGraph2 [5]	38.00 ±6.01	32.09 ±3.37	45.00 ±1.91	34.27 ±5.69	45.12 ±9.28	61.08 ±5.49	22.04 ±5.43
BRIDGE [33]	37.66 ±2.73	37.88 ±7.79	52.67 ±3.85	46.86 ±4.19	48.31 ±8.06	64.19 ±6.42	21.57 ±6.64
Comp.Only	**44.67 ±4.21**	**41.95 ±6.39**	**61.73 ±6.86**	**48.74 ±2.05**	**62.85 ±9.27**	**74.59 ±7.58**	21.40 ±4.59
Naive	43.53 ±1.09	37.24 ±4.92	56.40 ±3.73	43.69 ±3.73	61.29 ±6.87	72.50 ±4.76	22.47 ±3.15
UniCG	44.20 ±0.98	39.66 ±4.79	56.73 ±4.17	46.96 ±1.66	54.51 ±6.30	71.49 ±3.51	**24.47 ±1.38**

Table F.6: ROC-AUC (%, mean±std) of **5-shot graph classification and link prediction** under Uni-Unseen. Best results are presented in **bold**, and runner-ups are underlined.

Task	Graph Classification			Link Prediction	
Method	BACE	BBBP	HIV	WN18RR	FB15k-237
GCN [11]	48.51 ±1.71	55.29 ±7.47	49.44 ±7.54	59.76 ±0.79	57.14 ±7.71
GAT [22]	48.07 ±3.09	54.42 ±7.01	49.77 ±6.08	58.09 ±1.32	55.30 ±6.67
SAMGPT [32]	55.96 ±2.65	62.40 ±9.67	51.22 ±5.56	60.49 ±0.69	53.62 ±9.27
MDGPT [31]	57.09 ±2.07	62.11 ±9.90	50.46 ±8.00	59.65 ±1.27	56.72 ±6.69
MDGFM [24]	55.72 ±4.68	64.92 ±7.07	42.33 ±8.52	55.58 ±2.94	51.94 ±7.65
GCOPE [34]	55.96 ±2.44	64.37 ±1.42	47.26 ±10.41	59.26 ±1.03	56.89 ±7.59
GFT [25]	49.53 ±1.15	49.76 ±1.28	53.11 ±5.14	51.34 ±1.56	50.01 ±0.03
UniGraph2 [5]	55.17 ±2.42	63.49 ±7.77	52.33 ±7.09	65.03 ±3.56	**61.15 ±1.65**
BRIDGE [33]	53.93 ±1.97	**68.71 ±2.00**	52.12 ±6.66	58.81 ±0.90	57.90 ±8.56
Comp.Only	60.42 ±9.03	68.64 ±2.72	61.91 ±9.88	63.51 ±3.89	51.68 ±3.05
Naive	56.66 ±4.95	60.41 ±4.29	**64.30 ±6.23**	**65.60 ±2.97**	58.99 ±1.90
UniCG	**61.13 ±3.10**	64.94 ±6.41	63.48 ±8.27	63.33 ±2.14	60.55 ±2.30

F.2 Five-Shot Adaptation to Unseen Domains

Prediction across real graph domains. Expanding the support set to five examples per class reduces variance and improves the balance across node, graph, and link prediction. The complete tables extend the representative summary in Table 2 to every evaluation domain. UniCG remains among the leading methods across the three prediction levels, showing that the joint representation adapts effectively without requiring abundant target labels.

Synthetic classification and structural regression. The additional supervision improves CitSynth and preserves the strong regression trend. UniCG achieves the best result on four of the five regression domains and remains second on Tree. The contrast with synthetic classification confirms that the effect of the label budget depends on the target property rather than graph size alone.

Table F.7: Accuracy (%, mean±std) of **5-shot synthetic graph and node classification** under Uni-Unseen. Best results are presented in **bold**, and runner-ups are underlined.

Method	GraphFam	CitSynth
GCN [11]	35.13 ±2.18	47.05 ±2.32
GAT [22]	33.88 ±2.55	41.15 ±3.53
SAMGPT [32]	40.62 ±3.58	60.49 ±3.74
MDGPT [31]	39.25 ±3.05	61.15 ±3.38
MDGFM [24]	37.00 ±3.90	63.93 ±12.78
GCOPE [34]	37.50 ±4.35	67.05 ±9.25
GFT [25]	40.62 ±0.00	62.62 ±5.62
UniGraph2 [5]	37.96 ±3.60	64.43 ±5.86
BRIDGE [33]	46.25 ±8.58	67.54 ±7.11
Comp.Only	**47.12 ±2.70**	**74.59 ±9.81**
Naive	46.87 ±1.90	72.65 ±6.99
UNICG	46.25 ±1.31	73.82 ±7.29

Table F.8: R^2 (mean±std) of **5-shot structure-property regression** under Uni-Unseen. Best results are presented in **bold**, and runner-ups are underlined.

Method	Tree	ER	BA	SBM	Planar
GCN [11]	0.378 ±0.076	0.531 ±0.019	0.527 ±0.037	0.345 ±0.057	0.131 ±0.076
GAT [22]	0.371 ±0.067	0.532 ±0.019	0.532 ±0.038	0.362 ±0.059	0.138 ±0.063
SAMGPT [32]	0.332 ±0.074	0.066 ±0.053	0.051 ±0.018	-0.007 ±0.044	-0.035 ±0.043
MDGPT [31]	0.334 ±0.074	0.022 ±0.019	0.018 ±0.046	-0.020 ±0.043	-0.042 ±0.055
MDGFM [24]	0.321 ±0.093	0.014 ±0.019	0.027 ±0.037	-0.021 ±0.043	-0.043 ±0.049
GCOPE [34]	0.336 ±0.079	0.295 ±0.073	0.341 ±0.083	0.125 ±0.050	0.027 ±0.060
GFT [25]	0.325 ±0.087	-0.023 ±0.013	-0.027 ±0.036	-0.042 ±0.045	-0.053 ±0.055
UniGraph2 [5]	0.334 ±0.079	0.067 ±0.039	0.060 ±0.058	-0.019 ±0.046	-0.045 ±0.055
BRIDGE [33]	0.349 ±0.080	0.189 ±0.054	0.298 ±0.084	0.087 ±0.045	-0.011 ±0.048
Comp.Only	**0.999 ±0.001**	0.808 ±0.016	0.899 ±0.008	0.552 ±0.039	0.796 ±0.030
Naive	0.995 ±0.003	0.772 ±0.019	0.879 ±0.007	0.622 ±0.042	0.839 ±0.009
UNICG	0.996 ±0.003	**0.812 ±0.012**	**0.917 ±0.011**	**0.667 ±0.045**	**0.841 ±0.010**

F.3 TEN-SHOT ADAPTATION TO UNSEEN DOMAINS

Prediction across real graph domains. At ten shots, UNICG continues to improve on several node classification domains and remains competitive on molecular and knowledge graphs. The ranking is still domain dependent, which indicates that additional labels reduce estimation variance without eliminating the inductive biases of each pre-training method.

Synthetic classification and structural regression. UNICG obtains the strongest CitSynth result and leads on four structural regression domains. Across the three label budgets, structural regression exhibits the most consistent gains, while graph-family classification remains the most variable task. This task-dependent pattern explains why no single comprehension method dominates every dataset despite the broader benefit of increased supervision.

Table F.9: Accuracy (%, mean±std) of **10-shot node classification** under Uni-Unseen. Best results are presented in **bold**, and runner-ups are underlined.

Method	Cora	CiteSeer	PubMed	WikiCS	Amz-Comp	Amz-Photo	Arxiv
GCN [11]	26.33 ±4.26	21.92 ±1.84	37.87 ±3.08	23.34 ±1.74	35.56 ±1.64	36.22 ±2.86	10.67 ±1.26
GAT [22]	22.53 ±2.93	23.00 ±2.17	36.26 ±2.17	24.58 ±1.92	35.90 ±4.05	47.36 ±2.36	11.97 ±2.13
SAMGPT [32]	47.67 ±5.29	32.83 ±4.28	56.20 ±7.04	37.88 ±17.14	41.73 ±7.32	42.44 ±7.72	17.50 ±3.41
MDGPT [31]	47.87 ±6.18	32.42 ±2.25	52.67 ±4.10	38.50 ±7.83	53.49 ±7.05	59.66 ±5.58	18.50 ±4.06
MDGFM [24]	48.13 ±1.20	30.47 ±4.63	54.80 ±5.92	36.72 ±7.25	49.76 ±5.35	51.28 ±4.19	18.20 ±3.78
GCOPE [34]	33.11 ±2.38	29.46 ±2.19	47.73 ±3.09	27.75 ±4.83	34.64 ±4.60	40.41 ±3.48	17.50 ±4.38
GFT [25]	38.41 ±6.34	31.28 ±6.23	33.67 ±3.45	37.51 ±0.00	39.32 ±15.42	43.35 ±7.22	11.00 ±0.29
UniGraph2 [5]	41.22 ±4.12	35.86 ±2.45	47.45 ±1.29	32.43 ±4.45	41.49 ±8.47	64.26 ±4.90	26.83 ±4.71
BRIDGE [33]	42.27 ±7.61	38.03 ±2.60	57.33 ±2.49	46.38 ±5.00	49.42 ±9.83	60.07 ±5.34	25.03 ±1.71
Comp.Only	54.00 ±2.73	47.07 ±2.25	**66.13 ±1.77**	**55.36 ±2.74**	**64.54 ±7.21**	**79.93 ±3.87**	24.90 ±2.99
Naive	48.40 ±3.86	43.57 ±3.06	60.87 ±2.33	53.65 ±4.98	62.37 ±6.66	78.31 ±3.02	26.00 ±3.31
UniCG	**55.27 ±2.49**	**48.17 ±4.01**	65.27 ±1.31	53.17 ±4.24	62.88 ±8.81	78.04 ±3.25	**26.87 ±1.06**

Table F.10: ROC-AUC (%, mean±std) of **10-shot graph classification and link prediction** under Uni-Unseen. Best results are presented in **bold**, and runner-ups are underlined.

Task	Graph Classification			Link Prediction	
Method	BACE	BBBP	HIV	WN18RR	FB15k-237
GCN [11]	55.03 ±1.56	57.62 ±5.98	56.19 ±7.56	59.99 ±0.65	60.74 ±1.90
GAT [22]	55.89 ±1.43	56.96 ±5.52	56.92 ±6.45	58.69 ±0.34	60.05 ±2.47
SAMGPT [32]	56.56 ±3.18	62.95 ±8.92	55.33 ±5.76	60.94 ±1.03	55.50 ±6.44
MDGPT [31]	59.54 ±1.30	62.10 ±10.15	54.35 ±7.53	60.00 ±0.75	59.30 ±0.68
MDGFM [24]	58.78 ±1.85	66.65 ±6.41	52.84 ±10.76	56.94 ±3.52	55.27 ±6.19
GCOPE [34]	57.41 ±1.82	69.17 ±0.76	55.80 ±4.77	59.79 ±0.89	60.35 ±1.87
GFT [25]	50.00 ±0.00	57.87 ±0.00	49.36 ±3.97	49.70 ±0.53	50.23 ±0.46
UniGraph2 [5]	54.61 ±1.91	66.62 ±3.80	57.02 ±3.37	62.11 ±4.73	59.29 ±3.46
BRIDGE [33]	55.99 ±2.50	**70.50 ±0.00**	57.14 ±4.93	59.05 ±0.55	61.04 ±2.58
Comp.Only	**63.97 ±5.20**	65.80 ±4.97	64.61 ±6.32	64.82 ±3.29	52.81 ±3.10
Naive	59.16 ±3.88	60.76 ±3.12	**67.36 ±4.20**	**69.69 ±2.36**	**63.73 ±3.63**
UniCG	61.00 ±4.33	60.57 ±8.13	63.08 ±7.71	68.17 ±2.49	61.17 ±4.83

Table F.11: Accuracy (%, mean±std) of **10-shot synthetic graph and node classification** under Uni-Unseen. Best results are presented in **bold**, and runner-ups are underlined.

Method	GraphFam	CitSynth
GCN [11]	38.37 ±2.49	49.00 ±3.34
GAT [22]	38.75 ±4.82	43.77 ±1.80
SAMGPT [32]	38.50 ±0.85	61.15 ±5.83
MDGPT [31]	39.50 ±3.10	66.56 ±7.08
MDGFM [24]	46.38 ±3.96	61.80 ±5.72
GCOPE [34]	44.13 ±6.13	45.90 ±10.72
GFT [25]	40.62 ±0.00	66.23 ±1.64
UniGraph2 [5]	22.62 ±5.55	72.78 ±8.49
BRIDGE [33]	26.88 ±6.94	65.90 ±8.16
Comp.Only	**50.75 ±2.99**	78.36 ±4.80
Naive	49.12 ±1.96	74.26 ±3.59
UniCG	48.87 ±2.53	**79.51 ±5.59**

Table F.12: R^2 (mean±std) of **10-shot structure-property regression** under Uni-Unseen. Best results are presented in **bold**, and runner-ups are underlined.

Method	Tree	ER	BA	SBM	Planar
GCN [11]	0.407 ±0.036	0.541 ±0.010	0.542 ±0.008	0.354 ±0.028	0.114 ±0.082
GAT [22]	0.400 ±0.029	0.544 ±0.011	0.546 ±0.009	0.367 ±0.036	0.125 ±0.073
SAMGPT [32]	0.365 ±0.029	0.089 ±0.059	0.082 ±0.046	0.001 ±0.029	-0.055 ±0.043
MDGPT [31]	0.366 ±0.029	0.041 ±0.008	0.039 ±0.020	-0.012 ±0.030	-0.065 ±0.050
MDGFM [24]	0.362 ±0.035	0.035 ±0.020	0.057 ±0.029	-0.019 ±0.034	-0.073 ±0.050
GCOPE [34]	0.375 ±0.025	0.347 ±0.024	0.402 ±0.052	0.179 ±0.040	0.041 ±0.037
GFT [25]	0.344 ±0.041	-0.003 ±0.000	-0.012 ±0.009	-0.041 ±0.027	-0.080 ±0.049
UniGraph2 [5]	0.368 ±0.033	0.178 ±0.075	0.124 ±0.097	0.031 ±0.052	-0.057 ±0.049
BRIDGE [33]	0.384 ±0.023	0.216 ±0.031	0.296 ±0.061	0.106 ±0.038	-0.034 ±0.039
Comp.Only	**0.999 ±0.000**	0.819 ±0.018	0.905 ±0.011	0.629 ±0.016	0.813 ±0.028
Naive	0.996 ±0.001	0.800 ±0.008	0.907 ±0.006	0.643 ±0.035	0.829 ±0.018
UniCG	0.998 ±0.000	**0.828 ±0.007**	**0.931 ±0.004**	**0.708 ±0.021**	**0.847 ±0.003**

Table G.1: Performance of unconditional graph generation under Uni-Seen and Uni-Unseen, measured by V.U.N. (%, higher is better) and MMD (lower is better). Best results are presented in **bold**, and runner-ups are underlined.

Method	Tree		ER		BA		SBM		Planar		Mean	
	V.U.N. ↑	MMD ↓	V.U.N. ↑	MMD ↓	V.U.N. ↑	MMD ↓	V.U.N. ↑	MMD ↓	V.U.N. ↑	MMD ↓	V.U.N. ↑	MMD ↓
Uni-Seen (all domains seen in pre-training)												
G2PT [2]	1.96	**0.002**	32.94	**0.011**	16.21	**0.013**	17.58	**0.010**	11.37	**0.020**	16.01	**0.011**
DiGress [23]	0.78	0.225	32.81	0.229	18.75	0.191	28.52	0.189	4.69	0.624	17.11	0.292
DeFoG [15]	0.00	0.369	84.38	0.048	61.33	0.123	62.11	0.050	1.95	0.509	41.95	0.220
Erdős–Rényi	0.00	0.364	**90.62**	0.050	**71.88**	0.161	**73.05**	0.065	10.55	0.687	**49.22**	0.266
Gen.Only	5.34	0.163	48.11	0.441	46.29	0.255	48.14	0.446	56.89	1.016	40.95	0.464
Naive	2.98	0.289	39.65	0.382	38.51	0.232	41.44	0.370	**59.90**	0.951	36.50	0.445
UNICG	**7.86**	0.147	37.53	0.542	34.91	0.333	37.13	0.563	59.12	1.115	35.31	0.540
Uni-Unseen (target domain held out)												
G2PT [2]	0.00	0.999	**76.56**	0.085	**79.69**	0.121	**76.17**	0.115	35.16	0.681	**53.52**	0.400
DiGress [23]	0.00	0.368	3.91	0.389	10.55	0.145	19.92	0.278	43.75	0.833	15.63	0.403
DeFoG [15]	0.00	0.859	53.52	0.105	60.16	**0.109**	65.62	0.116	24.22	0.700	40.70	0.378
Erdős–Rényi	0.00	0.926	64.84	**0.068**	75.39	0.144	72.66	0.102	25.78	0.690	47.73	0.386
Gen.Only	0.84	0.610	54.00	0.312	52.51	0.299	55.99	0.469	**44.83**	0.908	41.63	0.520
Naive	3.65	**0.025**	13.89	0.379	35.01	0.184	18.11	0.368	37.92	**0.364**	21.72	0.264
UNICG	**3.99**	0.039	11.46	0.437	32.28	0.148	28.74	**0.089**	17.02	0.411	18.70	**0.225**

G Additional Results: Graph Generation

Section 5.2 compares generation on corpora built for *comprehension*, as required to evaluate both capabilities over the same corpora. This appendix additionally evaluates standard generation benchmarks comprising five synthetic structure domains with exact structural predicates (Table G.1), de-novo molecules (Table G.2), and property-conditioned generation (Table G.3).

Dedicated generators remain stronger on the structure domains. Under Uni-Seen, UNICG reaches a mean V.U.N. of 35.3, compared with 42.0 for DeFoG. The Erdős–Rényi reference reaches 49.2 while obtaining poor MMD, showing that structural validity and diversity alone do not imply distributional fidelity. The Uni-Unseen block exhibits the same generation-side conflict after the target domain is excluded from pre-training and then used for unsupervised target-domain adaptation. Appendix C.3 specifies the exposure conditions.

Distributional fidelity can conceal memorization. G2PT attains near-zero MMD (0.002–0.020) and high raw validity (0.80–0.94), which would make it the strongest method under distributional metrics alone. However, its **novelty is only** 0.035**–**0.447, indicating substantial reproduction of its training graphs. UNICG achieves novelty between 0.39 and 1.00 on the same domains. This comparison motivates reporting V.U.N. alongside MMD and identifies a limitation of per-domain autoregressive generation.

Molecules and control. On QM9 the unified backbone generates chemically typed molecules at validity 25.6 and novelty 74.3, on par with its matched *Gen.Only* reference (26.5/79.9) but well below DiGress (63.9 validity). On ZINC250k, whose graphs are roughly three times longer, UNICG remains below the random reference, revealing the difficulty of long molecular sequences under the current configuration. Prepending a target-density token yields graphs in the requested bucket 71.3% (Uni-Seen) and 68.3% (Uni-Unseen) of the time against a random floor of 33.3, and the shared-adapter variant matches it, so the conflict that damages sample *quality* leaves controllability intact.

H Additional Results: Ablation and Sensitivity Analysis

H.1 A Frozen-Backbone Test of Expert Decoupling

Frozen-Backbone Diagnostic. Table H.1 freezes the jointly pre-trained backbone and continues specialization through the adapters. Under this diagnostic, UNICG does not outperform *Naive*. The result indicates that task-specific experts reduce direct interference but cannot remove interference

Table G.2: Performance (%) of de-novo molecule generation on QM9 and ZINC250k. Best results are presented in **bold**, and runner-ups are underlined.

Dataset	QM9					ZINC250k				
Method	**Validity**	**Unique.**	**Novelty**	**Atom-val.**	**Mol-val.**	**Validity**	**Unique.**	**Novelty**	**Atom-val.**	**Mol-val.**
Erdős–Rényi	28.70	75.33	85.01	80.39	28.64	5.78	88.93	**100.00**	77.23	5.76
DiGress [23]	63.92	70.21	73.57	92.66	63.90	11.42	98.95	**100.00**	81.31	11.38
DeFoG [15]	21.46	83.41	**86.59**	76.68	21.42	1.22	**100.00**	**100.00**	62.75	1.22
G2PT [2]	**89.32**	95.75	55.64	**98.87**	**91.17**	**44.32**	**100.00**	**100.00**	**96.54**	**47.62**
Gen.Only	26.53	99.76	79.91	93.11	59.60	2.00	**100.00**	**100.00**	87.11	13.64
Naive	25.60	**100.00**	72.69	92.23	58.17	0.00	0.00	0.00	85.32	2.27
UNICG	25.60	99.00	74.31	92.78	58.40	0.67	**100.00**	**100.00**	86.46	3.12

Table G.3: Target-bucket accuracy (%, mean±std) of controllable graph generation under Uni-Seen and Uni-Unseen. Best results are presented in **bold**, and runner-ups are underlined.

Method	**Uni-Seen ↑**	**Uni-Unseen ↑**
Gen.Only	72.13 ±1.62	**78.39 ±3.17**
Naive	**73.49 ±1.41**	69.77 ±10.05
UNICG	71.34 ±2.55	68.32 ±9.95

Table H.1: Generation V.U.N. (%, mean±std over five runs) of the frozen-backbone diagnostic. Best results are presented in **bold**, and runner-ups are underlined.

Method (frozen backbone)	**V.U.N. ↑**
Gen.Only	**42.5 ±18.0**
Naive	41.6 ±18.1
UNICG	37.5 ±17.3

already encoded in the shared backbone. This finding complements Table 3, where the backbone remains trainable during joint pre-training.

H.2 Sensitivity across Expert Capacity and Vocabulary Fidelity

Each point in the rank–scaling and vocabulary sweeps is obtained from a single run. We therefore interpret broad trends rather than small differences between neighboring configurations. For reference, the five-run standard deviations of UNICG in Table 1 are 0.62, 2.59, and 0.70 on Cora, BACE, and WN18RR, respectively.

Comprehension Sensitivity to Adapter Capacity. Figure 2 evaluates the 4×4 grid $r \in \{8, 16, 32, 64\}$ and $\alpha \in \{16, 32, 64, 128\}$ under the Uni-Seen full-data setting of Table 1. Its three surfaces represent node classification on Cora, graph classification on BACE, and link prediction on WN18RR. Figure H.1 uses the same setting for PubMed, BBBP, and FB15k-237, respectively. The default configuration is (r=32, α=64). Cora and WN18RR remain within relatively narrow performance bands across most of the grid, while the molecular domains exhibit broader interior optima. PubMed favors the high-rank, high-scaling corner, whereas FB15k-237 peaks at an intermediate rank. No capacity pair dominates all six domains, showing that the benefit of task decoupling cannot be explained by a uniformly favorable adapter size.

Generation Sensitivity to Adapter Capacity. Figures H.2 and H.3 report V.U.N. for the same Uni-Seen full-data runs and the same $r \times \alpha$ grid. The panels in Figure H.2 correspond to Cora, BACE, and WN18RR, while those in Figure H.3 correspond to PubMed, BBBP, and FB15k-237. Generation varies more strongly than comprehension. Higher rank benefits Cora and PubMed over much of the grid, while BACE and BBBP favor moderate configurations and decline near some capacity extremes. The two knowledge graphs show different local optima, with WN18RR responding more strongly to scaling and FB15k-237 remaining high across most configurations. These task-specific surfaces support the use of a shared default rather than domain-wise hyperparameter selection.

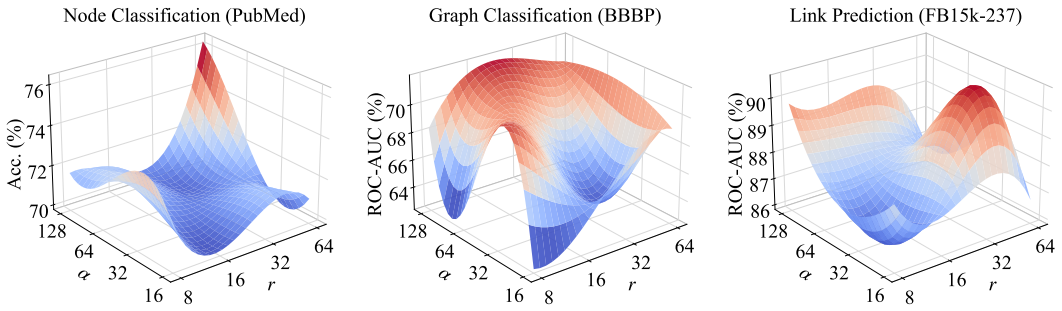


Figure H.1: Uni-Seen full-data comprehension over the $r \times \alpha$ grid. From left to right, the panels report node classification on `PubMed`, graph classification on `BBBP`, and link prediction on `FB15k-237`.

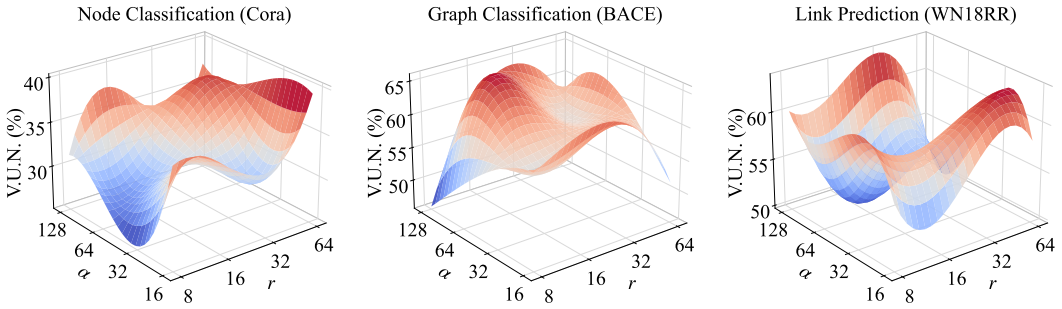


Figure H.2: Uni-Seen full-data generation V.U.N. over the $r \times \alpha$ grid. From left to right, the panels report `Cora`, `BACE`, and `WN18RR`.

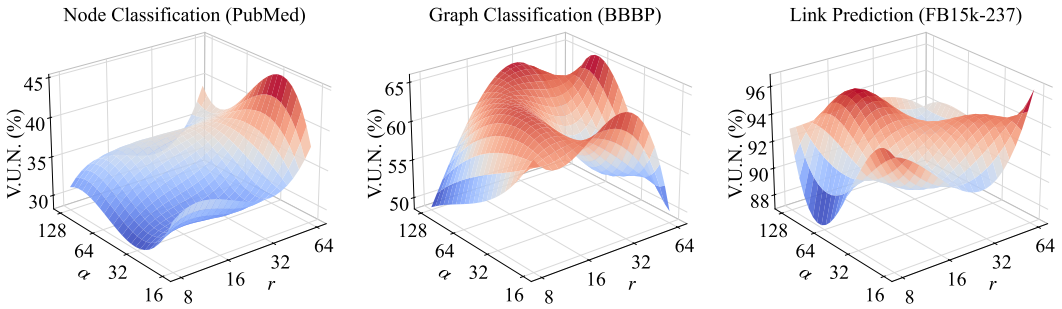


Figure H.3: Uni-Seen full-data generation V.U.N. over the $r \times \alpha$ grid. From left to right, the panels report `PubMed`, `BBBP`, and `FB15k-237`.

I Discussion and Limitations

A unified evaluation perspective. To our knowledge, this work provides the first systematic evaluation in which one shared autoregressive graph model performs direct comprehension and unconditional generation over the same multi-domain corpora. The evaluation spans node classification, graph classification, link prediction, graph-property regression, structural graph generation, knowledge-graph generation, and molecular generation. It also separates target-domain exposure from label availability through Uni-Seen, Uni-Unseen, and few-shot adaptation. This design moves beyond unifying task notation alone by testing whether both capabilities can coexist within the same representation, backbone, and adaptation pipeline.

Scope of the present study. The experiments focus on static graphs that can be serialized within the context length of the backbone. Temporal graphs, large-scale heterogeneous graphs, and generation with complex external constraints remain outside the current evaluation. Uni-Unseen is inductive with

respect to target examples but transductive with respect to the vocabulary schema, and its generation branch uses unlabeled target graphs during adaptation. These choices isolate representation transfer and joint optimization without conflating them with unseen symbol discovery or zero-shot domain identification. Extending the framework to unprovisioned schemas, longer graph sequences, and fully zero-shot generation would provide increasingly demanding tests of generalization.

Outlook. The results establish that graph comprehension and generation can be evaluated as complementary capabilities of one model rather than as disconnected problem families. We expect this perspective to support future research on larger graph foundation models, scalable graph tokenization, continual cross-domain adaptation, controllable graph synthesis, and generative reasoning over graph-structured knowledge. More broadly, the unified setting offers a common basis for studying when predictive supervision improves generation, when generative training strengthens graph representations, and how interference between the two objectives should be controlled.