Molecular Graph Captioning via Hybrid MolT5 Architecture

Youssef ADOUIRI ALAOUI

M2 Data Science

Institut Polytechnique de Paris

2025-2026

Outline

1. Introduction
2. Baseline Analysis
3. Proposed Architecture
4. Inference Strategy
5. Results
6. Conclusion
7. END
8. Annexes

The Challenge: Graph-to-Text Generation

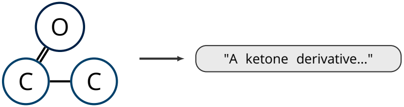
Objective: Translate a 2D Molecular Graph $G = (V, E)$ into a natural language description S.

Applications:

- AI-driven Drug Discovery.
- Augmenting chemical databases (PubChem).
- Summarizing scientific literature.

The Core Difficulty

Isomorphism vs. Semantics: Two molecules can share 90% of the same atoms but differ by a single bond order (Stereochemistry).



Exemple : *The molecule is an amino cyclitol glycoside that is 2-deoxystreptamine in which the pro-R hydroxy group is substituted by a 6-amino-6-deoxy-alpha-D-glucosyl residue. It has a role as an antimicrobial agent. It derives from a 2-deoxystreptamine. It is a conjugate acid of a 2'-deamino-2'-hydroxyneamine(3+)*

The Baseline: Rigid Retrieval (GCN-BERT)

The provided baseline utilizes a **Retrieval-Only** approach.

How it Works

1. **Training:** Aligns GCN graph embeddings with frozen BERT text embeddings (MSE Loss).
2. **Inference:** Projects test graph → embedding space → Retrieves Nearest Neighbor from Training Set.

Why We Can Do Better

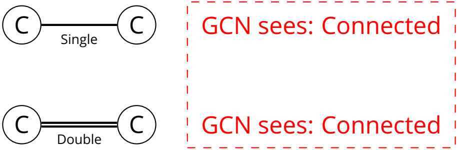
The baseline suffers from the **Closed-World Assumption**:

- **No Generative Capability:** If a test molecule is unique (novel structure), the baseline *must* retrieve an incorrect description from the training set.
- **Weak Encoder:** The baseline GCN ignores edge features (bond types), failing to distinguish isomers in the embedding space.

The Baseline: GCN

Disadvantage

1. **Oversmoothing (GCN):** GCNs average neighbor features, blurring the distinction between specific functional groups in large rings.
2. **Edge Ignorance:** Standard GCNs ignore edge attributes.

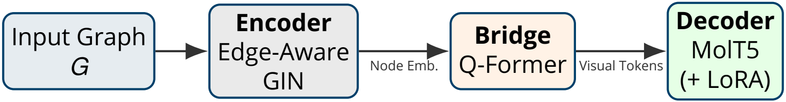


GCN sees: Connected

GCN sees: Connected

Solution: The Hybrid MolT5 Architecture

I propose a three-stage pipeline designed to solve the geometric and semantic bottlenecks.



- **Encoder (Graph Isomorphism Network):** Satisfies Weisfeiler-Lehman test (Isomorphism). Explicitly embeds bond types.
- **Bridge (Q-Former):** Compresses variable graph sizes into fixed "Visual Tokens" via Cross-Attention.
- **Decoder (MolT5):** Pre-trained on ChEBI-20. Knows IUPAC nomenclature natively.

Detail: Encoder & Decoder Choices

1. Edge-Aware GIN

Standard GCN sums neighbors. But GIN sums neighbors + **Edge Features**.

$$h_v^{(k)} = \text{MLP}^{(k)} \left((1 + \epsilon^{(k)}) \cdot h_v^{(k-1)} + \sum_{u \in \mathcal{N}(v)} (h_u^{(k-1)} + \mathbf{e}_{uv}) \right)$$

Result: Can distinguish isomers.

2. MolT5 + LoRA

MolT5: Tokenizer treats "methyl" as a concept, not letters.
LoRA: Low-Rank Adaptation.

$$W_{new} = W + B \cdot A$$

Allows fine-tuning 220M parameters on personal computer in ≈12 hours.

-> (Nvidia GTX 1650ti ***Cuda*** Enabled)

The Bridge: Q-Former Formulation

The Dimension Mismatch Problem:

- **Input (H_G):** Graph node features from GIN. Dimensions: **N** × **300**.
- **Target (Z):** LLM (MolT5) expects fixed-size token embeddings. Dimensions: **32** × **768**.

Algorithm: Cross-Attention Compression

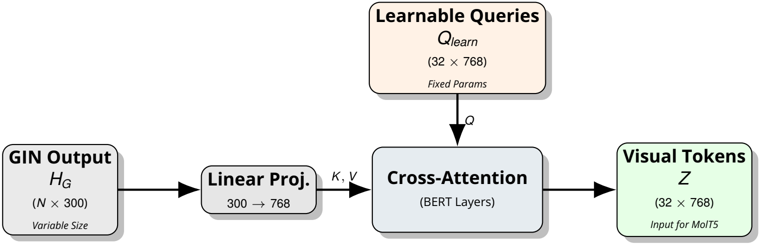
We introduce **32 Learnable Queries** $Q_{learn} \in \mathbb{R}^{32\times768}$. The Q-Former projects the graph features and compresses them via Cross-Attention:

$$Z = \text{Softmax}\left(\frac{Q_{learn} \cdot (H_G W_K)^T}{\sqrt{d}}\right)(H_G W_V)$$

- **Queries (Q):** Fixed learnable parameters (32×768).
- **Keys/Values (K, V):** The atom features H_G ($N \times 300$) projected to 768 by matrices W_K, W_V.
- **Output (Z):** Fixed "Visual Tokens" (32×768) ready for the Decoder.

The Bridge: Data Flow & Dimensions

The Q-Former acts as a "Translator" from the Graph modality ($d = 300$) to the Text modality ($d = 768$).



Why this matters:

- **Bridge:** Projects the 300-dim chemistry embeddings to the 768-dim language space.
- **Compression:** Regardless of molecule size ($N = 10$ or $N = 100$), the LLM always receives exactly **32 tokens**.

Empirical Motivation: Dataset Overlap Analysis

To determine the optimal inference strategy, we analyzed the structural similarity between the Test set ($N_{test} = 1,000$) and the Training set ($N_{train} = 31,008$)

Methodology: Extracted dense graph embeddings using the trained **Edge-Aware GIN** encoder. Computed Cosine Similarity Matrix (Test × Train).

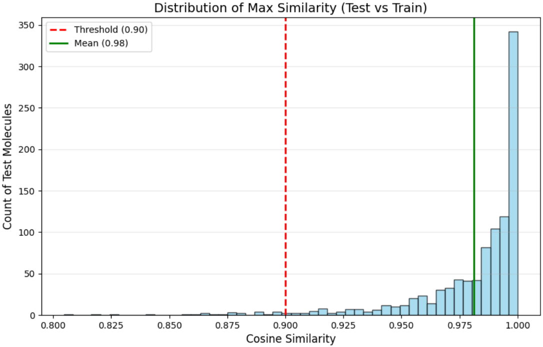


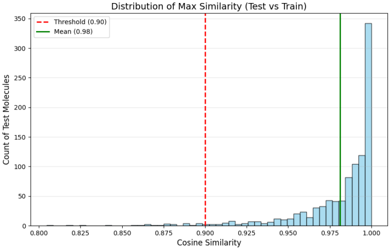
Figure 1: Distribution of Max Similarity Scores. The red line denotes our retrieval threshold ($\tau = 0.9$).

Empirical Motivation: Dataset Overlap Analysis

To determine the optimal inference strategy, we analyzed the structural similarity between the Test set ($N_{test} = 1,000$) and the Training set ($N_{train} = 31,008$)

Quantitative Results:

- **Average Max Similarity: 0.9811**
- **Matches** > 0.90**: (97.6%)**
- **Matches** > 0.99**: (51.9%)**

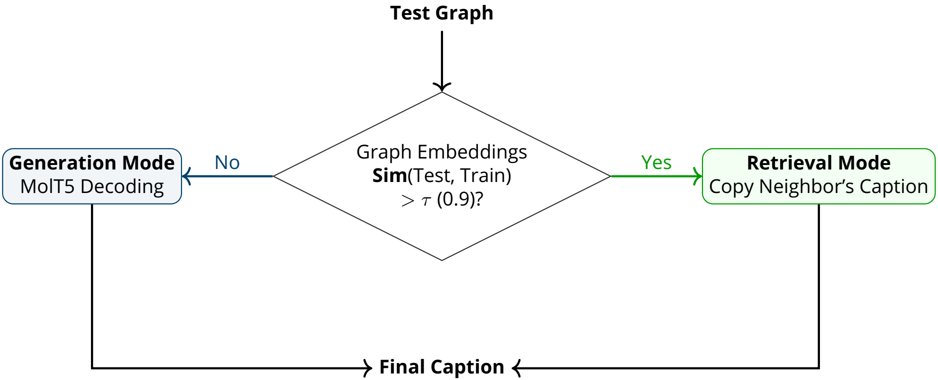


Observation

Over **50%** of the Test set is practically similar (> 0.99) to training examples.
Conclusion: A retrieval-based approach is statistically guaranteed to outperform pure generation for the majority of the test set.

The Hybrid Retrieval-Generation

Observation: The Test set chemically overlaps with the Training set.



This strategy handles common structures via retrieval (high accuracy) and novel structures via generation (MolT5 robustness).

Experimental Results

By fixing the baseline's inability to generate, and upgrading the encoder to handle bond types, we achieve a significant boost in Score for this Dataset and BLEU-4 and BERTScore on validation set.

Model Architecture	**Score**	**BLEU-4**	**BERTScore**
Baseline (GCN Retrieval)	0.492	0.1693	0.8732
EdgeAwareGIN + MolT5 (Pure Gen)	-	0.2526	0.9038
Hybrid MolT5 (Gen + Retrieval)	**0.532**	**0.4301**	**0.9307**

Table 1: Leaderboard Performance Comparison

Thank You for Your Attention!

Annex A: Encoder Hyperparameters & Math

Module: `graph_encoder.py` (Class: `GNN`)

Hyperparameters:

- **Architecture:** GIN (Graph Isomorphism Network)
- **Layers:** 5 (`num_layer`)
- **Hidden Dimension:** 300 (`emb_dim`)
- **Dropout:** 0.5 (`drop_ratio`)
- **Readout:** "Last" (`JK` - Jumping Knowledge)

Edge-Aware Update Equation: (

$$h_v^{(k)} = \mathrm{MLP}^{(k)} \Bigg((1 + \epsilon^{(k)}) h_v^{(k-1)} + \sum_{u \in \mathcal{N}(v)} (h_u^{(k-1)}$$

) Where $\mathbf{e}_{uv}$ represents the bond feature embedding added *inside* the aggregation, allowing differentiation of single/double bonds.

Input Features (Embeddings):

- **Nodes (9):** Atomic Num, Chirality, Degree, Charge, Num Hs, Radical, Hybridization, Aromaticity, Ring.
- **Edges (3):** Bond Type, Stereo, Conjugation.

Annex B: Q-Former Bridge Configuration

Module: `bridge.py` (Class: `Qformer`)

The Q-Former bridges the modality gap between the Graph ($d = 300$) and the LLM ($d = 768$).

Configuration Logic

- **Base Model:** `bert-base-uncased` (initialized weights).
- **Hidden Size:** 768 (`hidden_size`).
- **Number of Queries:** 32 (`num_query_token`).
- **Cross-Attention Frequency:** 2 (`cross_attention_freq`).
  - *Note:* Cross-attention is applied at every 2^{nd} layer of the BERT encoder to fuse graph info.

Dimensionality Transformation:

$$\text{Input: } [N_{atoms}, 300] \xrightarrow{\text{Linear Proj.}} [N, 768] \xrightarrow{\text{Q-Former}} \text{Output: } [32, 768]$$

Annex C: MolT5 Decoder & LoRA Config

Module: `model.py` (Class: `Graph2Seq`)

Base Model:

- `laituan245/molt5-base`
- Pre-trained on ChEBI-20 (Chemical Entities of Biological Interest).
- Native understanding of SMILES and chemical nomenclature.

LoRA Fine-Tuning Params:

- **Rank (r):** 16
- **Alpha (α):** 32
- **Target Modules:** Query (q), Value (v).
- **Trainable Params:** $<$ 1% of total.

LoRA Update Equation

For a pre-trained weight matrix $W_0 \in \mathbb{R}^{d \times k}$, the update is constrained by low-rank decomposition matrices $B \in \mathbb{R}^{d \times r}$ and $A \in \mathbb{R}^{r \times k}$:

$$W = W_0 + \frac{\alpha}{r} BA$$

Annex D: Training Strategy

Experimental Setup

Parameter	Value
Batch Size	32
Epochs	12
Optimizer	AdamW
Learning Rate	2×10^{-4} (Encoder), 5×10^{-5} (Decoder)
Loss Function	Cross Entropy (Token Generation)
Hardware	Nvidia GTX 1650ti
Training Time	$\approx$ 12 Hours

Objective Function:

$$\mathcal{L} = -\sum_{t=1}^{T} \log P(y_t | y_{<t}, Z_{graph})$$

Where Z_{graph} are the 32 visual tokens from the Q-Former.

Annex E: Hybrid Inference Algorithm

Pseudo-code for Decision Logic

Algorithm 1: Hybrid Retrieval-Generation

```
Input: Test Graph $G_{test}$, Training Set $D_{train}$, Threshold $\tau = 0.9$
Output: Caption $S$
  1 Compute Embedding $v_{test} = \text{GIN}(G_{test})$
  2 Find Nearest Neighbor:
        $$(v_{best}, S_{best}) = \underset{(v_i, S_i) \in D_{train}}{\text{argmax}} \cos(v_{test}, v_i)$$
  3 Calculate Similarity Score: $sim = \cos(v_{test}, v_{best})$
  4 If $sim > \tau$:
        • Return $S_{best}$ (Retrieval Mode)
  5 Else:
        • Generate tokens $S_{gen} = \text{MolT5}(\text{QFormer}(v_{test}))$
        • Return $S_{gen}$ (Generation Mode)
```

Annex F: Evaluation Metrics Definitions

1. BLEU-4 (Bilingual Evaluation Understudy)

- Measures the overlap of N-grams (up to $N = 4$) between the generated text and the reference.
- **Why it matters:** Checks for correct chemical syntax (e.g., matching "2-methyl" exactly).

$$\text{BLEU} = \text{BP} \cdot \exp\left(\sum_{n=1}^{4} w_n \log p_n\right)$$

2. BERTScore (Semantic Similarity)

- Uses a pre-trained BERT model to compute cosine similarity between token embeddings of the candidate ($\hat{x}$) and reference (x).
- **Why it matters:** Captures meaning even if words differ (e.g., "toxic" $\approx$ "harmful").

$$R_{\text{BERT}} = \frac{1}{|x|} \sum_{x_i \in x} \max_{\hat{x}_j \in \hat{x}} \mathbf{x}_i^T \mathbf{x}_j$$