ALTEGRAD 2025-2026

Molecular Graph Captioning: Graph-to-Text Generation

Nazim Fadli, Zié Coulibaly, Mathis Lecoq

January 15, 2026

Abstract

This report presents our work on the ALTEGRAD Kaggle challenge on *Molecular Graph Captioning*, where the goal is to generate natural-language descriptions from molecular graphs. We first reproduce an embedding-based retrieval baseline, then introduce two improved retrieval models based on edge-aware graph encoders, contrastive graph–text alignment, and matching-based re-ranking. We further explore generative approaches, including a Retrieval-Augmented Generation (RAG) pipeline with large language models and an end-to-end graph-to-text model based on MolT5. We evaluate all methods using validation metrics and Kaggle leaderboard scores, observing consistent improvements over the baseline. Our best retrieval-based model achieves a Kaggle score of 0.61, outperforming both RAG (0.57) and end-to-end generation (0.58), highlighting the effectiveness of contrastive retrieval with chemical feature modeling.

1 Introduction

Molecular graph captioning is the multimodal task of translating structured molecular graphs into coherent natural language descriptions. Situated at the intersection of Graph Representation Learning [4] and NLP, the objective is to capture key structural and functional characteristics (e.g., charge, functional groups) from the graph topology.

In this report, we investigate three distinct solution paradigms:

1. **Retrieval-Based:** Identifies the best-matching caption from a pre-existing training bank using shared embedding spaces.
2. **Retrieval-Augmented Generation (RAG):** A hybrid approach that prompts a Large Language Model (LLM) with retrieved context and explicit graph summaries to combine factual grounding with generative flexibility.
3. **Generative (MolT5):** Utilizes a specialized encoder-decoder architecture to synthesize novel descriptions directly from graph features, enabling *de novo* captioning.

2 Dataset and Problem Setting

The dataset consists of molecular graphs stored as `torch_geometric.data.Data` objects. The goal is to learn a mapping $f : G \to S$ from an input graph G to a natural language description S. The graph components are:

- **Node Features ($\mathbf{x} \in \mathbb{N}^{N\times 9}$):** Nine categorical indices per atom, including atomic number, chirality, formal charge, hybridization, and aromaticity flags.
- **Edge Indices (`edge_index` $\in \mathbb{N}^{2\times E}$):** Connectivity defined by directed edges (bidirectional for undirected bonds).
- **Edge Attributes (`edge_attr` $\in \mathbb{N}^{E\times 3}$):** Three categorical features per bond: bond type, stereochemistry, and conjugation status.
- **Identifier (`id`):** Unique key for submission alignment.

3 Baseline Pipeline: Embedding-Based Retrieval

3.1 Model

Two encoders map inputs into $\mathbb{R}^{768}$.

Text Encoder. Captions are represented using frozen `bert-base-uncased` embeddings. For caption y_i, the text embedding is

$$v^i_{text} = \text{BERT}(y_i)_{\texttt{[CLS]}} \in \mathbb{R}^{768}. \tag{1}$$

Graph Encoder. A 3-layer GCN encodes molecular graphs using topology only: all nodes are initialized with a shared learnable vector, and atom/bond features are ignored. Node embeddings are globally pooled and projected to obtain

$$v^i_{graph} \in \mathbb{R}^{768}. \tag{2}$$

3.2 Training and Inference

The graph encoder is trained using MSE to align v^i_{graph} with its paired v^i_{text}. At inference, the caption whose embedding maximizes cosine similarity with the query graph embedding is retrieved from the training set.

3.3 Experimental Results

Kaggle score (Baseline) = 0.489.

4 Proposed Solution 1: Improving Retrieval via Contrastive Graph–Text Alignment with Matching-based Re-ranking

To address the baseline limitations, we propose an improved retrieval framework that incorporates chemical features and explicitly learns to discriminate correct and incorrect graph–text pairs via contrastive and matching-based objectives.

4.1 Architectural Overview

Text Embedding. Each caption y_i is encoded using a pretrained BERT model, producing a fixed text embedding

$$v^i_{text} \in \mathbb{R}^{768}, \tag{3}$$

which is precomputed and treated as a frozen target during training.

Graph Embedding (GINE). Molecules are represented as attributed graphs and encoded using a graph neural network MolGNN composed of three **GINEConv** layers. Unlike the baseline, the encoder incorporates nine atom features and three bond features. Categorical features are embedded and summed to form initial node and edge representations. Node updates follow:

$$h_i^{(l+1)} = \text{MLP}^{(l)}\left((1+\epsilon)h_i^{(l)} + \sum_{j\in\mathcal{N}(i)} \text{ReLU}(h_j^{(l)} + e_{ij})\right) \tag{4}$$

Final node representations are aggregated using global add pooling and projected to obtain

$$v^i_{graph} \in \mathbb{R}^{768}. \tag{5}$$

Both v^i_{graph} and v^i_{text} are ℓ_2-normalized for cosine similarity.

4.2 Training Strategy: Dual Objectives

Training optimizes a composite loss combining contrastive alignment and matching supervision [3]:

$$\mathcal{L} = \mathcal{L}_{\text{GTC}} + \mathcal{L}_{\text{GTM}}. \tag{6}$$

4.2.1 Graph–Text Contrastive (GTC) Loss

We replace MSE with a symmetric InfoNCE objective. Given a batch of graph and text embeddings $\{v^i_{graph}\}$ and $\{v^i_{text}\}$, we compute similarities

$$S_{ij} = \alpha \langle v^i_{graph}, v^j_{text} \rangle, \tag{7}$$

where $\alpha = \exp(s)$ is a learned temperature. The contrastive loss is:

$$\mathcal{L}_{\text{GTC}} = -\frac{1}{2N} \sum_{i=1}^{N} \left(\log \frac{e^{S_{ii}}}{\sum_j e^{S_{ij}}} + \log \frac{e^{S_{ii}}}{\sum_j e^{S_{ji}}} \right). \tag{8}$$

4.2.2 Graph–Text Matching (GTM) Loss

To improve fine-grained discrimination, we train a lightweight MLP matching head that predicts whether a concatenated pair $[v^i_{graph}; v^j_{text}]$ is a true match. Hard negatives are selected within each batch by choosing the non-matching text embedding with highest cosine similarity to v^i_{graph}.

4.3 Inference and Results

Two-stage Retrieval. At inference, a test graph is embedded as v^{test}_{graph} and used to retrieve a Top-K shortlist ($K = 10$) based on cosine similarity to training text embeddings. The GTM head then re-ranks these candidates, and the caption with the highest matching score is selected.

Performance. The best checkpoint is selected using validation Recall@1. This method substantially improves over the baseline:

$$\textbf{Kaggle score (V1)} = \textbf{0.61}.$$

4.4 Alternative Architecture: GATv2 with Virtual Nodes

We additionally evaluated a more expressive encoder based on **GATv2**, which computes dynamic attention weights for each edge and can, in principle, focus on chemically salient substructures.

Result. Despite higher complexity, this variant achieved similar retrieval performance (R@1 ≈ 0.58) and a Kaggle score of 0.60. Given the marginal gains, we retain the simpler GINE-based encoder.

5 Proposed Solution 2: Retrieval-Augmented Generation (RAG) with LLMs

Pure retrieval models are constrained to reproducing captions seen during training. To enable novel yet grounded descriptions, we propose a Retrieval-Augmented Generation (RAG) pipeline that leverages the reasoning capabilities of Large Language Models (LLMs). The key idea is to condition an LLM on explicit molecular properties and retrieved similar captions, allowing it to synthesize factually accurate and stylistically consistent descriptions.

5.1 Prompt Engineering Strategy

To interface molecular graphs with a text-only LLM, we design a structured prompt composed of three components: (1) a deterministic textual summary of explicit graph properties (e.g., atom counts, charge, aromaticity), (2) the top-k retrieved captions from the GINE-based retriever as few-shot examples, and (3) a system instruction prompting the model to act as a chemistry expert. Details of the prompt template are provided in Section A.

5.2 Implementation Details: Mistral-7B and Quantization

We adopt **Mistral-7B-Instruct-v0.2** as the generative backbone, balancing expressivity and efficiency. To reduce memory footprint, we apply **4-bit Normal Float (NF4) quantization** via the `bitsandbytes` library.

5.2.1 Inference Configuration

Text generation uses nucleus sampling with conservative settings to reduce hallucinations:

- **Temperature:** 0.1
- **Top-p:** 0.9
- **Max New Tokens:** 512, selected based on training-set length statistics

5.3 Experimental Results

Kaggle score (RAG) = 0.57.

5.4 Note on Alternative Generative Approaches

Before adopting RAG, we experimented with fine-tuning a smaller decoder-only model (**GPT-2**) using **LoRA**. This approach yielded only marginal improvements over the retrieval baseline (Kaggle score ≈ 0.49). Due to computational constraints preventing fine-tuning of larger models, we discarded this strategy in favor of the inference-only RAG pipeline described above.

6 Proposed Solution 3: End-to-End Generation with MolT5

While RAG relies on existing captions, our second approach aims to solve the task via true *de novo* generation. We trained a dedicated encoder-decoder architecture, **MolT5**, to translate molecular graphs directly into natural language, we were inspired by [1] and [4].

6.1 Architecture: GATv2 with Virtual Node

For end-to-end generation, we employ a graph encoder based on **GATv2** augmented with a **Virtual Node** to capture both local chemical interactions and global molecular context.

6.1.1 Encoder: Graph Attention Network (GATv2)

Each molecule is represented as an attributed graph with nine atom-level and three bond-level categorical features. Each feature field is mapped to a learnable embedding, and embeddings within a node or edge are summed to form the initial representations.

We use multi-head **GATv2** layers with dynamic, content-dependent attention computed over node and bond features:

$$\alpha_{ij} = \text{softmax}_j \left(\mathbf{a}^\top \text{LeakyReLU} \left(\mathbf{W}[h_i \| h_j \| e_{ij}] \right) \right). \tag{9}$$

Residual connections, layer normalization, and GELU activations are applied to stabilize training.

6.1.2 Global Context via Virtual Node

To enable efficient global information exchange, we introduce a **Virtual Node**, a learnable global embedding added to all node representations at each layer. After every GATv2 layer, node embeddings are aggregated via global add pooling and used to update the virtual node through an MLP, allowing it to accumulate molecule-level context while influencing subsequent message passing.

The encoder outputs a sequence of node embeddings, which is preserved for generation.

6.2 Decoder and Alignment Strategy

We adopt **MolT5-base** as the decoder. Instead of collapsing the graph into a single vector, the full set of node embeddings is mapped to the T5 embedding space ($d = 768$) using a nonlinear MLP connector and provided as the encoder input sequence. This allows the decoder to attend directly to individual atoms during generation.

Training uses a composite objective:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{Gen}} + \lambda \mathcal{L}_{\text{Contrastive}}, \tag{10}$$

where $\mathcal{L}_{\text{Gen}}$ is the standard cross-entropy loss with teacher forcing. For contrastive alignment, node embeddings are mean-pooled into a global graph representation, while text representations are obtained by mean-pooling the MolT5 encoder states. These projected vectors are aligned using a CLIP-style contrastive loss with a learned temperature. We set $\lambda = 0.2$.

6.3 Implementation Details

We trained the model end-to-end on a single GPU using the following hyperparameters:

- **Model Dimensions:** GNN hidden dimension of 128 and T5 base decoder ($d = 768$).
- **Optimization:** AdamW optimizer with a learning rate of 1×10^{-4}, weight decay of 0.01, and a linear warmup of 500 steps.
- **Training Schedule:** Trained for 15 epochs with a batch size of 8.
- **Loss Balancing:** The contrastive loss weight λ was set to 0.2.
- **Inference:** Text generation was performed using Beam Search with a width of 5 beams.

6.4 Experimental Results

Kaggle score (MolT5) = 0.58.

Note on Training Improvements: Metric-Aware Optimization While our current approach relies on standard Teacher Forcing (Cross-Entropy), this method can be improved by explicitly modeling local correspondences as proposed by [2].

- **Cycle Consistency:** We could train a reverse *Text-to-Graph* adapter simultaneously. By enforcing that $G \to T \to G'$ reconstructs the original graph, we leverage the description not just as a target, but as a semantic bridge, ensuring the generated text preserves the structural fidelity of the molecule.

7 Generative Validation Methodology

For both generative approaches (MolT5 and RAG), we implemented a validation routine that we run to validate the results. We used **BLEU-4** and **BERTScore (F1)** using `distilbert-base-uncased` as an encoder.

References

[1] Zhiyuan Liu et al. *MolCA: Molecular Graph-Language Modeling with Cross-Modal Projector and Uni-Modal Adapter*. 2024. arXiv: 2310.12798 [cs.CL]. URL: https://arxiv.org/abs/2310.12798.

[2] Hyuntae Park, Yeachan Kim, and SangKeun Lee. *Bridging the Gap Between Molecule and Textual Descriptions via Substructure-aware Alignment*. 2025. arXiv: 2510.26157 [cs.LG]. URL: https://arxiv.org/abs/2510.26157.

[3] Alec Radford et al. "Learning Transferable Visual Models From Natural Language Supervision". In: *Proceedings of the 38th International Conference on Machine Learning*. Ed. by Marina Meila and Tong Zhang. Vol. 139. Proceedings of Machine Learning Research. PMLR, 18–24 Jul 2021, pp. 8748–8763. URL: https://proceedings.mlr.press/v139/radford21a.html.

[4] Benjamin Sanchez-Lengeling et al. "A Gentle Introduction to Graph Neural Networks". In: *Distill* (2021). https://distill.pub/2021/gnn-intro. DOI: 10.23915/distill.00033.

A Appendix: Prompt Engineering Template

To enable In-Context Learning for the RAG pipeline, we construct a structured prompt that combines a static expert persona, a rule-based summary of the graph's topology, and the retrieved textual context. Below is an example of a complete prompt fed to the LLM.

```
[System Instruction]
You are a chemistry expert.

[Graph Summary (Rule-Based)]
Target molecule graph summary:
- Number of atoms:  29
- Number of bonds:  31
- Contains aromatic atoms:  yes
- Contains rings:  yes
- Total formal charge:  145
- Number of heteroatoms:  5
- Contains phosphorus:  no
- Contains nitrogen:  yes

[Retrieved Context (from GINE Retrieval)]
Below are descriptions of molecules that are structurally and functionally similar to
the target molecule:
[1] The molecule is the chloride salt of ipratropium.  It is a quaternary ammonium
salt and a chloride salt.  It contains an ipratropium.

[2] The molecule is an organic chloride salt having nitro blue tetrazolium(2+) as the
counterion.  It contains a nitro blue tetrazolium(2+).

[Task Instruction]
Based on the graph summary and the retrieved examples above, write a concise and
accurate description of the target molecule.
Description:
```

Figure 1: Template of the RAG prompt. The System and Task instructions are static, while the Graph Summary and Retrieved Context are dynamically generated for each test molecule.