MASTER 2 DATA SCIENCE

ECOLE POLYTECHNIQUE

ALTEGRAD

Youssef Sidhom, Clément Destouesse, Oumouhani Elvilaly

15 janvier 2026

Abstract

This report details our methodology for the Molecular Graph Captioning challenge, a multi-modal task requiring the translation of 2D atomic structures into human-readable tex. We evolved from a retrieval-based GCN baseline to a Dual-Tower Graph architecture. Our main contributions were integrating chemical-native embeddings (ChEmbed) and Contrastive Learning with InfoNCE loss, thanks to which we significantly enhanced the alignment between the graph and text modalities. Our final model achieved a public leaderboard score MRR of 0.614.

Contents

1 Introduction

The goal of this project is to bridge the gap between structured graph data and unstructured natural language, specifically known as Molecular Graph Captioning. This task is essential for translating complex 2D structures into summaries for scientific literature and assisting in AI-driven drug discovery.
We were given two dataset on which to train our model, one training with roughly 31000 samples, and one validation with 1000 samples. Our solution searches on the training set the textual description which best matches the embedding of the test data. We mainly focused on improving the retrieval performance and not on generating tasks.

2 Dataset analysis

2.1 Graph structure

The dataset consists of approximately 33,000 molecule-text pairs. Each molecule is represented as a `torch_geometric.data.Data` object.

- **Node features (atoms):** Each atom contains 9 categorical indices, including atomic number, chirality, degree, formal charge, and hybridization.
- **Edge features (bonds):** Bonds are defined by type (single, double, triple, aromatic), stereochemistry, and conjugation.

2.2 Language modality

The captions describe the key functional groups and properties of the molecules. A significant challenge identified was the highly specialized chemical nomenclature, which general-purpose language models often fail to tokenize or embed accurately.

3 Methodology: retrieval

This section outlines the strategic approach and technical considerations employed to address the challenge. We detail the iterative refinements made to our dual-stream architecture, focusing on enhancing both structural and semantic information capture. Specifically, the following subsections describe the optimizations applied to:

- **Text Representation Learning:** Enhancements to semantic encoding and the integration of advanced language models.
- **Graph Representation Learning:** Improvements in topological feature extraction and node embedding techniques.

3.1 Text Representation Learning

n this section, we detail the various embedding models employed to enhance the representation of molecular descriptions. We explore how these specialized encoders capture semantic nuances to improve downstream performance.

3.1.1 SciBERT

he textual description of molecules is highly specific to chemistry. Thus, a generalist embedder as the one used in the baseline, BERT, may struggle with efficient representation. The text we dealt with is not representative of the training of the model, with specific word occurences being very limited. We then searched for another model which could provide us with better results. Our first choice was `SciBERT` (Scientific BERT), model trained on scientific papers more suitable for our end. This improved our leaderboard score to **MRR = 0.563**

3.1.2 Chembed

While SciBERT was a significant upgrade over general English models, it remains a broad scientific encoder. It understands science generally but lacks the specialized understanding of chemics, and the relationships between molecules and functional properties. To achieve higher semantic accuracy, we needed a more specified model to our use case. This led to the use of ChEmbed (`BASF-AI/ChEmbed-full`), a model specialized on chemical structures. Thanks to this improvement, our score went up to **MRR = 0.614**.

3.2 Graph Representation Learning

3.2.1 Limitations of the baseline

The initial baseline used a simple Graph Convolutional Network (GCN). This model is quite simplistic : it projects the given graph into a specific sized-space, without taking into account the specificities of the data (atom in that case). The produced embedding is then matched against the BERT-computed text embeddings to retrieve the closest meaning.
As noted, this architecture was blind to specific atom types and bond attributes, treating the molecule as a generic skeleton. It performed poorly, and our first step was to switch to another model to integrate the main graph features.

3.2.2 GINE implementation

We then transitioned to a **Graph Isomorphism Network with Edge features (GINE)**. This allowed the model to incorporate edge features (bond types) into the message-passing phase, ensuring that a $C = C$ double bond is processed differently than a $C - C$ single bond. Moreover, it takes into account the atom features. The model then distinguishes between different molecule efficiently.

3.2.3 Graph Transformer Component

While the GINE architecture successfully integrated chemical features, it remained constrained by the fundamental limitation of standard Message Passing Neural Networks (MPNNs): locality.

GINE relies on a k-hop message passing mechanism, where an atom's representation is updated only by aggregating information from its immediate neighbors. To capture information from distant atoms, the model requires stacking many layers, which often leads to the "over-smoothing" problem where node representations become indistinguishable. This locality is a significant bottleneck for molecular captioning.

To overcome this, we replaced the GINE backbone with a Graph Transformer, which utilizes a global self-attention mechanism. Unlike MPNNs, this allows every atom to attend to every other atom in the molecule in a single layer, regardless of the topological distance between them. By capturing these long-range dependencies and global structural patterns, the Graph Transformer generates a holistic molecular embedding that is far more semantically aligned with the high-level descriptions found in the text captions.

3.2.4 Dual-Tower Graph Transformer

The "Dual-Tower" (or Two-Tower) architecture is the state-of-the-art framework that enabled our final performance breakthrough. It is designed to solve the fundamental problem of modal alignment: taking two completely different types of data—a molecular graph and a text description—and mapping them into a shared vector space where they can be directly compared.In previous phases, we treated the text embeddings as static "ground truth" targets (using pretrained BERT/SciBERT). The Dual-Tower approach, however, trains both the graph and text encoders simultaneously, allowing them to adapt to each other.

1. **Architecture Components:** As the name implies, the model consists of two parallel neural networks ("towers") that process the two modalities independently:
   - **The Graph Transformer (Tower A)**: The molecular graph (atoms, bonds, and features).Function: It processes the structural information using the Graph Transformer (as detailed in Section 2.1.3). This network learns to "read" the molecule's geometry and chemistry.It ouputs a single vector embedding, $v_{graph} \in \mathbb{R}^d$, that summarizes the entire molecule.
   - **The Text Encoder (Tower B):** It uses the ChEmbed model (initialized with chemistry-specific weights) to process the text. Crucially, we add a trainable Adapter Layer (MLP) on top of ChEmbed. This adapter projects the general chemical text embedding into the specific alignment space required by our task and outputs a single vector embedding, $v_{text} \in \mathbb{R}^d$, that summarizes the description.
2. **The Shared Latent Space:** The core mechanism of this architecture is the Shared Latent Space. The two towers project their respective inputs into the same high-dimensional vector space (e.g., 256 or 768 dimensions). If a specific graph matches a specific text description, their vectors (v_{graph} and v_{text}) should be very close together (high cosine similarity).

4 Optimization Strategies and Methodological Refinements

4.1 Loss improvements

4.1.1 Contrastive loss

One of the drawbacks of the ealier approach also was the use of MSE loss when retrieving vectors from the embedding space. During the training phase, the MSE nudges the model to produce and retrieve embedding as close as possible to the training ones, sometimes at the expense of the true relationship between the vector. In opposite, the contrastive loss allows more liberty to the model, as the focus in mainly on the distance in the embedding space rather than vector components :

$$MSE(\mathbf{u},\mathbf{v}) = \frac{1}{n}\sum_{i=1}^{n}(u_i - v_i)^2$$

$$\mathcal{L}(Y,\mathbf{u},\mathbf{v}) = (1-Y)\frac{1}{2}(D_w)^2 + (Y)\frac{1}{2}\max(0, m - D_w)^2, \text{with } D_w = \sqrt{\sum_{i=1}^{n}(u_i - v_i)^2}$$

where D_w represents the euclidean distance between two vectors. The point of this change was to be more efficient in the context of a retrieval task, rather than pure regresion.Thanks to those modifications, we reached the leaderboard score of **MRR = 0.523**

4.1.2 Hard Negative Mining

Standard contrastive learning usually selects negative pairs randomly from the dataset. In this setup, the "negative" molecule is often completely different from the "target" molecule (e.g., a small ring vs. a long chain). This makes the task too easy for the model. Because the model can easily distinguish these random pairs, it does not learn detailed chemical features. To fix this, we implemented a *Hard Negative Mining* strategy. This approach forces the model to distinguish the target molecule from "hard negatives"—molecules that look very similar but have different text descriptions. This compels the model to focus on fine-grained structural details rather than just general shapes.

To find these hard negatives, we first need to measure how similar two molecules are. We use **Morgan Fingerprints**, which are binary vectors that represent the chemical substructures of a molecule. We calculate the similarity between two molecules, G_i and G_j, using the **Tanimoto coefficient**:

$$S(G_i, G_j) = \frac{|f_i \cap f_j|}{|f_i \cup f_j|} \tag{1}$$

where f_i and f_j are the fingerprints of the two molecules. During a preprocessing step, we search the entire dataset to find the top-k most similar molecules for every target molecule. These "nearest neighbors" become our pool of hard negatives.

We modified the training process to use these hard negatives. Instead of creating batches with random pairs, we ensure that for every target molecule G_i, the batch also contains a hard negative G_n selected from its nearest neighbors.

This changes the behavior of the loss function. The loss now includes a specific term for the hard negative in the denominator:

$$\mathcal{L} = -\log \frac{\exp(\text{sim}(v_{target}, v_{correct})/\tau)}{Z_{random} + \exp(\text{sim}(v_{target}, v_{hard\_negative})/\tau)} \tag{2}$$

By including this term, the loss function penalizes the model heavily if it thinks the target molecule is similar to the hard negative's description. To lower this loss, the model must learn to project the target and the hard negative into different parts of the vector space, effectively "pushing" them apart despite their structural similarity. Thanks to this new approach we improved our score up to **MRR = 0.622**.

4.2 Embedding improvements

4.2.1 Matryoshka Representation Learning

To improve retrieval efficiency and robustness, we implemented a Matryoshka Representation Learning strategy. Instead of training for a single fixed output dimension, this loss function forces the model to learn a hierarchical representation where the first dimensions (e.g., 64, 128, 256) are independently capable of accurate retrieval. This can avoid to have too much noise in the embedding and only focus on the most relevant components on the embedding when computing the retrieval to balance speed and accuracy. The results on the validation set are the following :

Dimension	64	128	**256**	512	768
MRR	0.5235	0.5748	**0.5996**	0.5867	0.5970

Table 1: Matryoshka performance across different embedding dimensions

Surprisingly (or not), the highest MRR score is achieved for an embedding of size 256, which is lower than the actual default embedding size we used, the one from SciBERT (768 dimensions). It suggests that the intrinsic dimensionality of the chemical-text space is lower than the the one of SciBERT.

However, this didn't quite improved the overall score, as Matryoshka RL is design for efficiency rather than performance. It was still an interesting fact to note that bigger isn't always better when it comes to embedding.

4.3 Strategy improvements

4.3.1 Two-stage retrieval (reranking)

Our next step during the project was to improve the retrieval part, rather than focus on the model and embedding. The intuition was that embedding has been well explored (different models and strategies), and we had to switch to another part of the project to keep improving the performances. We chose to focus on a two stage mechanism :

- **Retrieval** using the Matryoshka embedding (256 dimensions) to select the 20 best candidates per sample
- **Reranking** among the 20 candidates using a smaller GNN model with cross-attention mechanism to select the most pertinent embedding for our specific case

Unfortunately, this method only provided a very marginal improvement to **MRR = 0.567** on the leaderboard. Wde have continued applying this approach for each of the improvement on the embeddings (both on the graphs and on the descriptions) but it always yields a worst results. We do believe that the resons for this poor score is that our second step top-k cross-emebbedding wasn't very powerful and hence didn't help our model.

This would either suggest that this wasn't efficient and that we should try another method, or that contrary to our estimation, we still had work to do on the embedding.

5 Methodology: Generative model

After observing a performance plateau with purely retrieval-based methods, we transitioned to a Retrieval-Augmented Generation (RAG) framework anchored by MolT5-small. To bridge structural and textual modalities, we employ a pre-trained Dual-Tower model to retrieve descriptions of the top-k similar molecules. These descriptions serve as semantic "hints" encoded by the MolT5 encoder, while the target molecule is processed in parallel by a 3-layer TransformerConv graph encoder. These features are concatenated and passed to the MolT5-small decoder to synthesize the final caption.

The model achieved a **BLEU-4 score = 0.33** on both local validation and the public leaderboard. While this demonstrates intelligible text generation, the alignment between graph features and the pre-trained latent space remains suboptimal. We hypothesize the generator relied too heavily on retrieved hints. However, strict temporal constraints prevented the extensive hyperparameter tuning or architectural refinements (such as complex cross-attention mechanisms) necessary to address this limitation.

6 Results

Model / Phase	Public score
GINE + Contrastive loss	0.523
SciBERT upgrade	0.563
Two-Stage rerank	0.567
Dual Tower Graph Transformer	0.602
ChEmbed upgrade	0.614
Hard Negative Mining upgrade	0.622
Generative RAG	0.33

Table 2: Evolution of performance across different tests

7 Conclusion and suggestions

In this work, we addressed the challenging multi-modal task of Molecular Graph Captioning, aiming to bridge the gap between structured geometric data and unstructured natural language. While our current approach established a functional baseline, our analysis identifies critical limitations in retrieval-based paradigms and outlines a concrete roadmap for achieving state-of-the-art performance through generative modeling.

- **Architectural Refinement (Cross-Modal Attention):** Replace static concatenation with Cross-Attention. This enables dynamic focus on chemical substructures (e.g., aromatic rings) during generation, ensuring precise alignment between graph topology and text.
- **Encoder Initialization (Transfer Learning):** Training on limited data ($N \approx 32k$) 1 is inefficient. Use pre-trained GNNs (e.g., GIN-Virtual) to leverage rich, pre-established chemical semantics immediately.
- **Objective Optimization:** Transition from Cross-Entropy to Self-Critical Sequence Training (SCST). This Reinforcement Learning technique directly maximizes non-differentiable metrics like BLEU-4 and BERTScore2.Model Scaling: Semantic reasoning requires scale. Upgrade from MolT5-small to MolT5-base or MolT5-large to better capture complex molecular nuances and improve fluency.