Altegrad Project-Molecular Graph Captioning

Ilan BACRY Jin YING Théo BASSERAS

January 2026

1 Introduction

This project was carried out as part of the ALTEGRAD course of the MVA. It takes the form of a competition aiming to maximize the performance of a model that generates textual descriptions of molecules from their atomic graph structure. Throughout the project, our team explored a wide range of approaches, from classical representation alignment methods to more ambitious graph–language model architectures. Performance fluctuated during our experiments which reflected the challenges of aligning structural and linguistic latent spaces, but we eventually reached a best score of 0.62666 on the public leaderboard. The full implementation is available in our GitHub repository at `https://github.com/luckyman94/Altegrad-project-MVA`. The name of our team is BabaYin.

2 Baseline: Retrieval-Based Approach

As a starting point, we consider the retrieval-based baseline provided with the competition, which achieves a score of approximately **0.44** on the public leaderboard. This approach formulates the task as an embedding alignment problem between molecular graphs and their textual descriptions, without performing explicit text generation. During inference, captions are retrieved via nearest-neighbor search in the text embedding space. The overall pipeline is illustrated in Figure 1. .

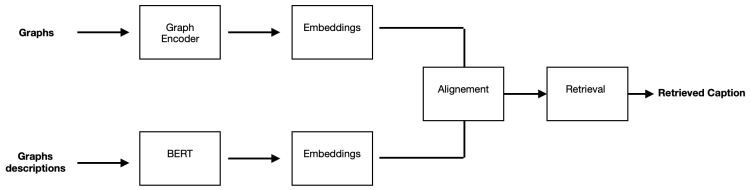

Figure 1: Pipeline of the baseline approach

3 Enhancing the Retrieval-Based Model

3.1 Improving the Graph Encoder

3.1.1 GINE Encoder

While the retrieval-based baseline provides a reasonable starting point, its graph encoder lacks chemical expressiveness. The baseline initializes all atoms with identical node features and relies on a simple GCN that ignores bond types, leading to a loss of important chemical and structural information.

To address these limitations, we improve the graph encoder while keeping the retrieval framework unchanged. We replace the basic GCN with a chemically informed GINEConv architecture [1], enriching node representations via a multi-dimensional AtomEncoder and incorporating bond and stereochemical information

through an EdgeEncoder. This chemically aware message-passing scheme enables better discrimination of molecular structures and improves retrieval performance to approximately **0.54** on the public leaderboard.

3.1.2 GPS Encoder

While GINEConv effectively captures local chemical interactions, it is limited to neighborhood-level message passing and struggles to model long-range dependencies in molecular graphs. However, many molecular properties depend on global structural patterns that cannot be captured by local interactions alone.

To overcome this limitation, we adopt a GPS-based architecture [2] that combines chemically informed message passing with global attention mechanisms. Each GPS layer integrates GINEConv with multi-head self-attention, enabling joint modeling of local chemical features and long-range structural relationships. This hybrid representation significantly improves the expressiveness of graph embeddings and yields our second best retrieval performance, reaching a score of **0.62395** on the public leaderboard.

3.2 Changing the Description Encoder

In addition to improving the graph encoder, we explored several alternatives for encoding textual descriptions. We experimented with multiple pretrained language models, including BERT, MiniLM, E5 ,in order to assess their impact on retrieval performance.

Among all tested approaches, the E5 model consistently provided the most effective text representations. E5 is a sentence embedding model introduced by Wang et al. [3], specifically trained for semantic similarity and retrieval tasks, making it particularly well suited for nearest-neighbor caption retrieval. Replacing the original text encoder with E5 led to a significant performance gain, resulting in our best overall score of **0.62666** on the public leaderboard.

3.3 Adding New Features

We explored the addition of global molecular features to improve model generalization, including counts of C, N, O and P atoms and the aromaticity ratio, combined with a GPS-based encoder. However, these hand-crafted features introduced noise and did not improve retrieval performance. Consequently, this direction was not pursued further.

3.4 Improving Retrieval

Beyond improving the encoders, we explored several strategies to further enhance the retrieval process itself. In particular, we investigated variations of the nearest-neighbor search as well as more fine-grained retrieval schemes based on the internal structure of textual descriptions.

3.4.1 k-Nearest Neighbors

As a first step, we extended the retrieval procedure from single nearest-neighbor retrieval to a k-nearest neighbors (kNN) strategy, aiming to improve robustness by considering a larger set of semantically similar captions. However, selecting or aggregating information from multiple neighbors did not lead to consistent performance gains and improvements remained marginal compared to single nearest-neighbor retrieval. Using this approach with $k = 5$ and the GPS-based graph encoder resulted in a score of **0.62394**. We have conducted few experiments. The results are summarized in Table 1

3.4.2 Structure-Aware Retrieval

We observed that many molecular descriptions follow a recurring structure, typically starting with an introductory sentence (e.g., *"The molecule is..."*), followed by structural details and functional or biological roles. Motivated by this pattern, we explored sentence-level retrieval strategies instead of retrieving entire descriptions.

Specifically, we performed separate kNN retrievals on different parts of the description, either by aligning corresponding sentences or by grouping them into semantic categories such as *Introduction*, *Structure*, *Role* and *Taxonomy*. However, this approach proved difficult to apply consistently, as many descriptions do not strictly follow this structure. As a result, segmentation errors led to noisy retrieval and limited performance gains.

Despite these improvements, retrieval-based approaches remain fundamentally limited by their reliance on existing textual descriptions. To overcome this limitation and enable true text generation, we next explore generative graph-to-text models.

3.5 Improving the loss function

We initially experimented with a Mean Squared Error (MSE) loss for aligning graph and text embeddings. Although the training loss quickly decreased to values on the order of 10^{-4} within the first epoch, this behavior did not translate into improved validation performance.

We attribute this rapid convergence to the point-wise regression nature of MSE. Our objective is to learn approximately isometric (bijective) mappings from graphs to the latent space and back.

Motivated by this objective, we replaced the MSE objective with a cross-entropy–based contrastive loss, namely the InfoNCE loss [4]. This loss is better aligned with the retrieval objective, as it enforces consistency between graph-to-text and text-to-graph mappings by comparing similar samples to one another within each batch.

Formally, given a batch of N molecular graph embeddings v and text embeddings t, the graph-to-text loss $\mathcal{L}_{v\to t}$ is defined as:

$$\mathcal{L}_{v\to t} = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{\exp(v_i^\top t_i/\tau)}{\sum_{k=1}^{N}\exp(v_i^\top t_k/\tau)}, \tag{1}$$

where τ denotes a temperature hyperparameter. The final objective is obtained by symmetrizing the loss:

$$\mathcal{L} = \frac{1}{2}\left(\mathcal{L}_{v\to t} + \mathcal{L}_{t\to v}\right). \tag{2}$$

Although switching to InfoNCE [5] aligns better with the retrieval objective, the performance did not improve significantly in practice. We hypothesize that this limitation is due to the *easy negative* phenomenon [6], where randomly sampled negatives in small batches are often chemically dissimilar enough to be trivially separated, preventing the model from learning fine-grained structural features.

4 Generative Approaches

4.1 Graph-to-Text Generation

4.1.1 Baseline Training Setup

We next explored a generative approach that directly translates molecular graphs into textual descriptions. The overall architecture of this approach is illustrated in Figure 2. We initially relied on GPT-2 as the generative backbone.

Molecular graphs are encoded using a GPS-based graph encoder and projected into a conditioning representation that is concatenated with a fixed textual prompt and passed to GPT-2. The model generates the description auto-regressively, while GPT-2 remains frozen and only the graph encoder and projection module are trained. We used a cross-entropy-loss between the generated and the ground truth descriptions. We could not use BLEU to optimize our model since BLEU is non-differentiable.

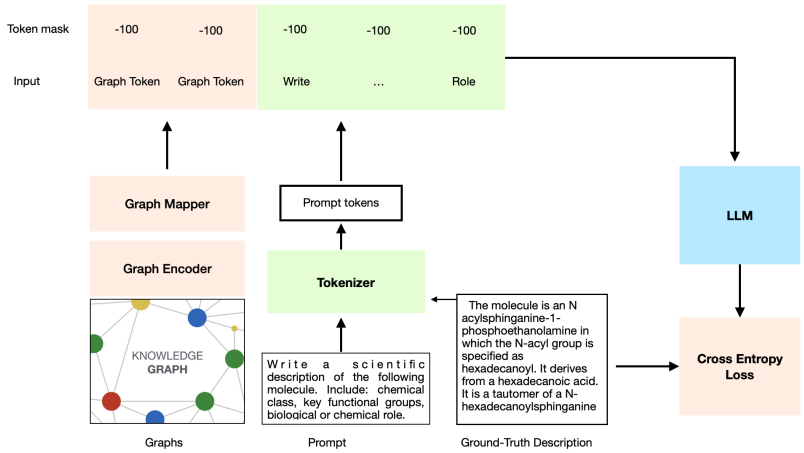

Figure 2: Graph-to-text generation pipeline based on graph-conditioned decoder-only language modeling.

We introduce the notion of *soft tokens*, which are continuous vectors used to condition the language model during generation, without corresponding to discrete vocabulary tokens. This is an hyperparameter that we optimized a lot. Generally our best results are with 4 *soft tokens*.

We report three representative generative experiments illustrating the impact of optimization choices (see Appendix 2). We initially used identical learning rates for the encoder and mapper, then introduced a smaller learning rate for the encoder to allow slower adaptation. Despite these adjustments, generative approaches remained significantly less effective than retrieval-based methods.

4.1.2 Alternative Training Strategies

We explored alternative generative training strategies by freezing the graph encoder and the llm and by training only the mapper. However, we observed that this architectural rigidity limited cross-modal alignment. Hence we chose to unfreeze the LLM. We used a non-linear architecture for the mapper. Our experiments were conducted with GPT-2 and BioGPT model, fine-tuned using LoRA [7] to reduce memory usage. Batch sizes ranging from 8 to 64 were evaluated, with 32 providing the best trade-off between training time and performance. The corresponding results are reported in Table 3. While fine-tuning the language model improved performance compared to a frozen setup, results remained below retrieval-based approaches. BioT5 [8] consistently outperformed GPT-2, highlighting the benefit of domain-specific pretraining.

Despite these adjustments, generative approaches remained less effective than retrieval-based methods. While we theoretically expected better performance from generative modeling, the results were constrained by:

- Limited Dataset Size: The available dataset is relatively small for training high-capacity generative models, making it difficult for the model to generalize complex chemical syntax without overfitting.
- Insufficient Training: Due to GPU resource limitations and time constraints, the large models we tested were not fully converged, resulting in underfitting.

4.1.3 Retrieval-Augmented Generation

We also explored a Retrieval-Augmented Generation (RAG) strategy [9], implemented using FAISS (Facebook AI Similarity Search) [10] for nearest-neighbor retrieval in the shared embedding space. In practice, this approach proved challenging to implement and stabilize, which resulted in overall poor performance. Moreover, the additional retrieval step substantially increased computational cost, requiring the use of a smaller language model.

4.2 Generative Modeling via Latent Space

4.2.1 Overview

This approach decomposes graph-to-text generation into simpler steps:

- learn a latent textual representation via text autoencoding.
- predict this latent representation from molecular graphs.
- generate the final description by decoding from the predicted latent.

4.2.2 Latent Space Learning

We begin by learning a meaningful latent representation of molecular descriptions independently of the molecular graphs. This latent space is constructed using an iterative and diagnostic autoencoding approach, starting from simple baselines and progressively refining the latent design based on reconstruction quality. Early experiments showed that low-dimensional continuous pooling severely degrades token-level information, whereas prefix-based representations better preserve sequence structure. Increasing the prefix capacity and partially unfreezing the text encoder further improved conditioning.

To overcome the blurring effects inherent to continuous latents, we ultimately adopted a discrete latent space based on a VQ-style formulation with a learnable codebook, EMA updates, commitment loss and straight-through gradients. With careful optimization and greedy decoding during evaluation, this approach achieved near-perfect text reconstruction performance, with BLEU scores close to **0.99** (see Table 4).

4.2.3 Graph-to-Latent Mapping

Once the latent space was fixed, we attempted to learn a mapping from molecular graphs to this latent representation (see Table 5). To stabilize learning, the task was decomposed into simpler supervised components. In particular, we trained classical models to predict high-level molecular categories (e.g., nucleotide, carbohydrate), achieving precision scores around **0.87**–**0.90** depending on the class.

In addition, we re-explored the use of data augmentation and explicit symbolic features in this setting, despite having already evaluated similar ideas in earlier experiments. In this context, we incorporated atom count features (C, O, N, P, S), functional group indicators defined using SMARTS patterns and SMILES representations. SMARTS is a rule-based chemical pattern language used to specify and detect functional substructures within molecules, while SMILES provides a linear string encoding of molecular graphs. These symbolic signals were concatenated and used as conditioning inputs to predict the latent representation learned during the autoencoding stage.

4.2.4 Limitations of the Learned Latent Manifold

Despite low training loss and good auxiliary performance, the learned latent space was not suitable for text generation. Small perturbations caused large semantic inconsistencies and BLEU scores remained below **0.34**, suggesting a lack of smoothness and robustness in the learned chemical manifold.

5 Future Work

A natural extension of this work would be to combine retrieval-based methods with generation. In particular, retrieved molecular descriptions could be used to condition a generative model pretrained on biomedical data, such as BioT5. Due to time constraints, this retrieval-augmented generative approach was not explored in the present project but represents a promising direction for future investigation.

References

[1] Keyulu Xu et al. "How Powerful are Graph Neural Networks?" In: *ICLR* (2019).

[2] Ladislav Rampášek et al. "Recipe for a General, Powerful, Scalable Graph Transformer". In: *NeurIPS* (2022).

[3] Liang Wang et al. "Text Embeddings by Weakly-Supervised Contrastive Pre-training". In: *arXiv preprint arXiv:2212.03533* (2022).

[4] Ting Chen et al. "A simple framework for contrastive learning of visual representations". In: *ICML*. PMLR, 2020, pp. 1597–1607. URL: https://proceedings.mlr.press/v119/chen20a.html.

[5] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. "Representation Learning with Contrastive Predictive Coding". In: *arXiv preprint arXiv:1807.03748* (2018).

[6] Joshua Robinson et al. "Contrastive Learning with Hard Negative Samples". In: *International Conference on Learning Representations (ICLR)*. 2021.

[7] Edward J Hu et al. "LoRA: Low-Rank Adaptation of Large Language Models". In: *ICLR* (2022).

[8] Eric Lehman et al. "Do We Still Need Clinical Language Models?" In: *arXiv preprint arXiv:2302.08091* (2023).

[9] Patrick Lewis et al. "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks". In: *NeurIPS* (2020).

[10] Jeff Johnson, Matthijs Douze, and Hervé Jégou. "Billion-scale similarity search with GPUs". In: *IEEE Transactions on Big Data* (2019).

Annex

Graph Encoder	Text Encoder	LR	Epochs	BLEU
GPS ($k = 10$)	E5	1×10^{-4}	20	**0.621**
GPS ($k = 10$)	E5	3×10^{-4}	40	0.58
GPS ($k = 5$)	E5	3×10^{-4}	40	0.62666
GNN ($k = 10$)	E5	3×10^{-4}	40	0.512
GNN ($k = 10$)	BERT	3×10^{-4}	40	0.501

Table 1: Summary of retrieval-based results using different graph encoders, text encoders and optimization settings.

Epochs	Soft Tokens	LR Mapper	LR Encoder	BLEU
20	4	1e−4	1e−5	0.45642
11	4	1e−4	1e−5	**0.49727**
10	1	1e−4	–	0.43

Table 2: Results of generative graph-to-text experiments using GPT-2.

Model	Epochs	BLEU	batch size	learning rate
GPT2	30	0.454	32	1e-5
BioGPT	30	0.463	32	1e-5
BioT5	30	0.474	32	1e-5

Table 3: Results of generative graph-to-text experiments.

Experiment	**Latent / Bottleneck**	**Encoder frozen?**	**Key modifications / notes**	**BLEU**
Initial T5 AE (mean-pooled)	Continuous pooled z (D=256)	No	Simple encoder pooling → linear z; decode with T5 decoder	0.48
Latent prefix (64 tokens, frozen)	Prefix tokens (64 × 512)	Yes (fully frozen)	Use first encoder tokens (no pooling); map to learned prefix; frozen encoder to save memory	0.36
Latent prefix VAE (small KL)	Continuous VAE (z dim=256)	Yes (frozen)	VAE with KL weight (0.05); length control; attempted regularized latent	0.16
Colab-optimized prefix AE	Prefix (64), AMP	Yes (mostly frozen)	Mixed precision, Colab-friendly batch size; tuning of beams/repetition settings	0.35
Optimized prefix AE (higher capacity)	Prefix (increased to 192 tokens)	Partially (top-2 blocks unfrozen)	Stronger projection head, lower LR, greedy decoding during eval, smaller batch size for memory	0.63
Naïve discrete AE (argmax codebook)	Discrete codes (argmax, no VQ training)	No	Simple argmax over a learned codebook (non-differentiable); no commitment/EMA/STE → codebook collapse	0.20
Proper VQ / discrete-latent AE (final)	VQ-style discrete codes (codebook + EMA + commitment + STE), LATENT_TOKENS ≈ 128–192	No (train encoder & decoder)	Straight-through estimator, codebook EMA, commitment loss, lower LR, optimized codebook size, greedy decode for BLEU → final optimized training	**0.99**

Table 4: Summary of experiments (chronological). BLEU scores are the measured corpus BLEU on the validation set after the reported run.

Model / Variant	Decoder loss	BLEU	Verdict (one line)
Latent MSE	no	0.10	Expected low lexical fidelity (continuous MSE blurs tokens).
Masked latent	no	0.07	Worse — masking harmed reconstruction identity.
Chem summaries (slot augment)	no	0.10	Neutral; redundant summary info didn't help BLEU.
Latent + cosine loss	no	0.10	Minor change — cosine regularization insufficient.
Latent + decoder loss (joint)	yes	0.34	Best direction: adding decoder reconstruction term helps.
More epochs (same config)	no	+0.01	Useless — overfitting latent / plateau.
Text→Text AE (oracle)	—	0.99	Trivial perfect reconstruction (upper bound).
Graph→Latent (unfrozen decoder)	—	0.34	Strong semantic mapping (best non-oracle).

Table 5: Compact table of primary latent-learning experiments (reported decoder loss / BLEU).