LELEM team: Molecular Graph Captioning

Kshitij Ambilduke **Nemanja Vujadinović** **Mohammed El Hassan Ayoubi**
ENS Paris-Saclay, Gif-sur-Yvette

Abstract

This study investigates molecular graph captioning, utilizing Large Language Models (LLMs) to generate natural language descriptions from graph-based molecular representations. To align the distinct semantic spaces of graph encoders and LLMs, we propose a two-stage training strategy: contrastive alignment followed by prefix-tuning. We employ a Graph Neural Network (GNN) encoder to capture atom- and bond-level information , integrating it with instruction-tuned LLaMa models. Our experiments reveal that, surprisingly, the 1B parameter model surpasses both the 3B and 8B variants in performance. Furthermore, we observe that post-generation re-ranking strategies—utilizing both graph-caption similarity and LLM-based evaluation—do not surpass the quality of raw captions generated by the best-performing model. All our code is available online on Github.[1]

1 Introduction

The success of Large Language Models (LLMs) across general and scientific language tasks has naturally translated to their application in biomedical and chemical domains, where they have shown promising performance Taylor et al. [2022]. Building on this, we study the use of LLMs for molecular graph captioning, a task that focuses on generating natural language descriptions of a molecule from its graph-based representation. The successful outcome of this task has the potential to support chemical research by improving molecular understanding and discovery.
Prior work that incorporates LLMs for molecular understanding typically uses 1D molecular representations, such as SMILES strings Weininger [1988]. Despite their effectiveness, such representations do not explicitly encode the 2D relational structure of molecules, needed for both human and LLMs comprehension of molecules. In this work, we utilize molecular representations obtained from a graph neural network (GNN) to condition language generation on both atom- and bond-level information.

2 Methodology

Consider G_θ as the graph encoder which takes as input x which is the Graph with node and edge features and returns a set of node embeddings $\bar{x} = (\bar{x}_1, \bar{x}_2, \cdots, \bar{x}_s)$ where $x_i \in \mathbb{R}^{G_{dim}}$ and s represents the total number of nodes in graph x. Following this, consider a Projector module, which projects each of these embeddings into the LLM dimension $x_G = P_\psi(\bar{x}) = (x_1, x_2, \cdots, x_s)$. These are the embeddings we use to prefix-tune a LLM f_ϕ to generate captions of the input graph. We describe the specific architectures of G_θ and P_ψ in Section 3.

2.1 Grounding graphs to text

In order to derive textual captions from molecules, it is necessary to align molecular representations with natural language. This is not natively supported, as LLMs operate in a semantic space that differs from graph-based molecular representations. To mitigate this gap, we propose a two-stage training strategy: (1) a contrastive alignment phase followed by (2) supervised fine-tuning of LLM (Fig. 1).

For implementing the contrastive alignment phase, we follow the approach proposed in Fei et al. [2025]. Given a molecular graph x, node and edge features are first processed by the graph encoder G_θ to produce a set of node-level embeddings $\bar{x}$. These embeddings are then passed through the projection module P_ψ to obtain graph representations in the language model's embedding space, i.e. $x_G = P_\psi(\bar{x})$. In parallel, the corresponding molecular caption is processed by the language model f_ϕ. We extract the textual representations from the 16th decoder block of LLaMa, and denote them as $x_{C;16}$. We use intermediate-layer representations here in order to capture rich contextualized semantics and therefore avoid the final-layer outputs, which may be overspecialized. During this stage, both G_θ and P_ψ are trainable, while f_ϕ remains frozen.

[1] https://github.com/vujadinovicn/molecular-graph-captioning/tree/main

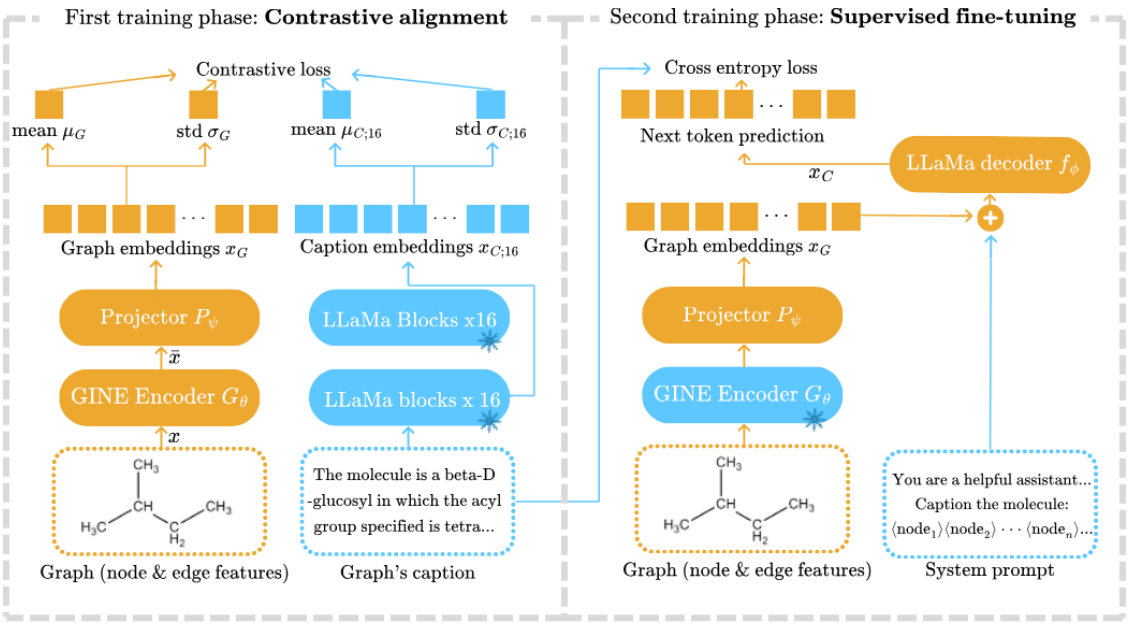


Figure 1: Two-stage training strategy overview.

To train this stage, we incorporate an InfoNCE loss van den Oord et al. [2018] on the graph and caption embeddings x_G and $x_{C;16}$. For each sample, we construct two pooled channels from both modalities: (a) mean-pooled vectors and (b) standard-deviation-pooled vectors. To compute InfoNCE objective, we first compute the similarity scores for both pairs of pooled vectors:

$$Z_{\text{mean}} = \frac{\mu_G \mu_{C;16}^\top}{\tau}, \quad Z_{\text{std}} = \frac{\sigma_G \sigma_{C;16}^\top}{\tau}, \tag{1}$$

where $\mu_G = \text{meanPool}(x_G)$, $\mu_{C;16} = \text{meanPool}(x_{C;16})$, $\sigma_G = \text{stdPool}(x_G)$ and $\sigma_G = \text{stdPool}(x_{C;16})$ are pooled vectors, and τ is a temperature hyperparameter. The final contrastive alignment loss is then defined as

$$\mathcal{L}_{\text{contrastive}} = \lambda_1 \mathcal{L}_{\text{InfoNCE}}(Z_{\text{mean}}) + \lambda_2 \mathcal{L}_{\text{InfoNCE}}(Z_{\text{std}}), \tag{2}$$

$$\mathcal{L}_{\text{InfoNCE}}(Z) = \frac{1}{2}\big(\text{CrossEntropy}(Z) + \text{CrossEntropy}(Z^\top)\big). \tag{3}$$

The importance of this training stage is illustrated in Fig. 2. Before contrastive alignment, the similarity matrix is noisy and lacks clear structure, with high similarity values distributed across unrelated graph–text pairs. After alignment, the highest similarity values are concentrated along the diagonal, showing that the model has learned to associate corresponding molecular graphs and textual captions. This implicitly reduces the risk of representational mismatch in the second stage.

2.2 Prefix-tuning LLM using graph embeddings

We tried several types of training for tuning the LLM and the graph encoder. These variants differ mainly along 2 axes. (a) Instruction tuning vs Supervised fine-tuning of LLM (b) Pretraining the graph model (Section 2.1) and freezing it vs pretraining the graph modela and tuning it vs training the graph model from scratch. Note that the difference between IT and SFT is that in the former, we calculate loss only on the LLM output whereas for the latter we calculate loss on the whole sequence. While parameter efficient finetuning is also an interesting way of tuning the LLM, in initial few runs, we observed that using LoRA adapters resulted in worse performance than tuning all the parameters of the LLM, hence we excluded experimenting with LoRA adapters in this project.

While instruction-tuning the LLM, we use a system prompt which includes a few random in-context examples in the form of captions from the training set. For the node embeddings, we use a special token `<|reserved_special_token|>` and extend the model embedding matrix to take this into account. After gathering the embeddings for all of the input sequence, we just replace the embeddings corresponding to this special token with the corresponding node embeddings.

Training Prompt Template

```
<|system|>
You are a helpful assistant that captions molecules based on their structure.  Provide
concise and informative captions.  Following are a few examples of captions:
## Example 1:  The molecule is a 4-O-[(E)-2-methyl-2-butenoyl]ascaroside...
## Example 2:  The molecule is an alkanesulfonic acid...
<|user|>
Caption the following molecule:  ⟨node_1⟩⟨node_2⟩...⟨node_N⟩
<|assistant|>
The molecule is a derivative of benzoic acid found in...
```

3 Implementation details

Graph model G_θ We implement a graph neural network encoder to process molecular graphs into structured embeddings that capture both atom- and bond-level information. Our encoder consists of two main components: an `AtomBondEncoder` for feature embedding and a `GINEEncoder` based on the Graph Isomorphism Network with Edge features (GINE) for message passing.

The `AtomBondEncoder` module encodes discrete atom and bond features into continuous vector representations. Given a set of categorical feature channels (e.g., atom type, formal charge, chirality, bond type, stereochemistry), we use separate embedding tables for each channel and sum the resulting embeddings:

$$\text{AtomBondEncoder}(x) = \sum_{i=1}^{C} E_i(x_i),$$

where C is the number of feature channels, E_i is the embedding layer for the i-th channel, and x_i is the corresponding categorical input. This design allows the model to learn distributed representations for each atomic and bond attribute while maintaining parameter efficiency.

The `GINEEncoder` processes the embedded node and edge features through a multi-layer GINE architecture **?**. Each layer performs neighborhood aggregation with edge feature conditioning:

$$h_v^{(l)} = \text{MLP}^{(l)}\left(h_v^{(l-1)} + \sum_{u \in \mathcal{N}(v)} \text{ReLU}\left(h_u^{(l-1)} + e_{uv}\right)\right),$$

where $h_v^{(l)}$ is the representation of node v at layer l, e_{uv} is the embedding of the bond between nodes u and v, $\mathcal{N}(v)$ denotes the neighbors of v, and $\text{MLP}^{(l)}$ is a two-layer perceptron with a hidden dimension twice the output dimension. We use batch normalization after each convolutional layer and apply dropout for regularization.

Our implementation uses $L = 5$ GINE layers with a hidden dimension of 256, followed by global mean pooling across nodes to obtain a graph-level representation. The final output is projected to a target dimension (e.g., 512) via a linear layer, with optional L2 normalization. This encoder Liu et al. [2024] achieves state-of-the-art performance on molecular graph understanding tasks and works well on our dataset as validated by our experimental results (Section 5). Other details can be found in the Appendix A.

Projector P_ψ The projection module consists of two fully connected layers, with a GELU activation function applied between them, and a layer normalization used at the output.

LLM f_ϕ We tried several base LLMs all differing in sizes. We chose to work only with instruction tuned LLMs since a few initial experiments helped up realize that without using an instruction tuned model, it is difficult to get structured output from the LLM such that it can be parsed at the end. To this end, we try the following LLMs as our base models: `meta-llama/Llama-3.2-1B-Instruct`, `meta-llama/Llama-3.2-3B-Instruct` and finally `meta-llama/Llama-3-8B-Instruct`.

Other parameters For computing the contrastive loss (Eq. 2), we use a temperature of $\tau = 0.15$ and set $\lambda_1 = \lambda_2 = 0.5$. For the optimizer, we use Adam Kingma and Ba [2014]. During the contrastive alignment stage, both G_θ and P_ψ are trained with a learning rate of $2e-4$, while in the fine-tuning stage, we train them with a learning rate of $1e-3$, and train f_ϕ with a value of $5e-5$. Moreover, in the contrastive learning stage, we train our model on 25 epochs using a batch size of 256, and in the fine-tuning stage, we train for 5 epochs with a batch size of either 4 or 8, depending on the choice of f_ϕ. For decoding, we emperically found that using beam search with a beam size of 4 was the best compromise between performance and inference speed, hence, we use the same.

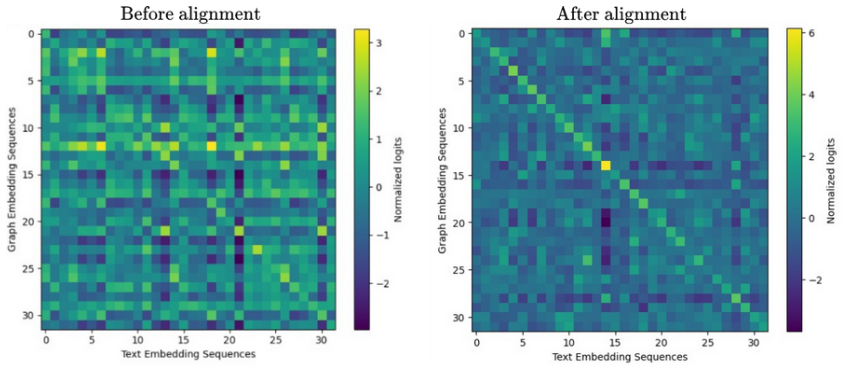


Figure 2: Similarity matrix before and after contrastive alignment on graph-caption pairs.

4 Re-ranking

After getting several candidate generations from all the models we trained, we tried re-ranking the generated captions corresponding to each molecule. We tried two approaches for re-ranking, one relying primarily on the contrastive multimodal alignment we described in Section 2.1 and other relying primarily on the linguistics and the node features provided in the dataset. We describe both of these approaches below.

4.1 Re-ranking using graph-caption similarity

For graph-caption similarity re-ranking, we select the graph encoder from the model that achieved the best performance on the test set and use it as a reference model. For each sample, we compute a graph embedding using this encoder and obtain its caption embeddings for each candidate model following the procedure described in Section 2.1. A similarity score between the molecular graph and each candidate caption is then computed using the similarity formulation defined in Eq. 1. The top-ranked caption, according to this score, is selected as the final prediction for each molecule.

4.2 Re-ranking using LLM

For re-ranking using a LLM, we use a stronger backbone LLM (`Llama-3-8B-Instruct`) and give it the node features as input in the system prompt along with a few strict instructions regarding re-ranking the given list of captions. Also following the training, we add a few in-context examples of captions from the training set. The exact prompt used for this part can be found in Appendix C

5 Results and Discussion

Dataset analysis The dataset contains multiple features that can be considered categorical or discrete in nature. A key challenge lies in the distribution of these features. Although the dataset is relatively small in terms of the number of graphs (32k graphs), it is large at the structural level, containing a total of 994,928 nodes and 2,069,580 edges.

The distributions of several features are highly imbalanced. In many cases, the feature values span a wide range, but the majority of observations are concentrated in a narrow region. For example, The atomic number as shown in Figure 3a, the values range from 0 to 119; however, the distribution is heavily skewed toward lower values. While larger values do exist, they are significantly less frequent and appear compressed when compared to the dense concentration of smaller values. This pattern is observed across many features and is particularly pronounced for the formal charge (Figure 3c), which is centered around a small set of dominant values; for the bond type feature (Figure 3b), where the distribution exhibits two distinct clusters; and for the number of radical electrons (Figure 3d), which is equal to zero in the vast majority of cases, with nonzero values occurring only rarely.

Evaluation We evaluate model performance using a BLEU score, BERTScore F1 and composite metric that combines both of these. We choose both these metrics to cover both syntactics and semantics of the generated captions. For calculating the Composite score, BERTScore is rescaled to a range of $[0, 100]$ and a mean of BLEU score and this re-scaled BERTScore is used as composite score. We hypothesize that this metric equally weights lexical accuracy and semantic fidelity. As an internal indication of how good our trained model is, we evaluate all our models on the validation set and report the score we get for all of the metrics mentioned above.

Results Our initial experiments focused on the 1B model size to establish the optimal training configuration through an ablation study. We compared the effectiveness of Instruction Tuning (IT) against Supervised Fine-Tuning (SFT), with the results conclusively identifying Instruction Tuning as the superior approach. We also evaluated different graph model strategies, specifically examining the impact of using "contrast vs. no contrast" setups by varying whether the graph model was pre-trained or trained from scratch, and whether it was kept frozen or updated during training. Based on the composite performance across these trials, we determined that the configuration of using a pre-trained graph model and keeping it frozen (training only the projection module) was the most robust choice for subsequent experiments. Using this optimized "pre-trained and frozen" strategy, we scaled our evaluation to compare 1B, 3B, and 8B (PEFT) model sizes. The results highlighted a clear performance distinction, among the three choices, we observed that 1B model performed the best.

LLM Size	Graph model	Frozen G_θ	LLM IT/SFT	Eval Model	BLEU	BERTScore	Composite
1B	From scratch	✗	IT	RoBERTa-base ChemBERTa-ZINC	12.57	24.76 96.21	18.67 54.39
	Pre-trained	✔	IT	RoBERTa-base ChemBERTa-ZINC	14.3	24.54 96.43	19.42 55.37
	Pre-trained	✗	IT	RoBERTa-base ChemBERTa-ZINC	40.03	56.32 97.44	48.17 68.74
	Pre-trained	✔	SFT	RoBERTa-base ChemBERTa-ZINC	13.7	24.82 96.15	18.48 54.14
3B	Pre-trained	✔	IT	RoBERTa-base ChemBERTa-ZINC	39.85	56.47 97.46	48.16 68.66
8B PEFT	Pre-trained	✔	IT	RoBERTa-base ChemBERTa-ZINC	34.67	52.86 97.27	43.76 65.97

Table 1: Comparative analysis of all the training methods

Results after re-ranking As discessed in the section above, our best submission to the challenge was based on `meta-llama/Llama-3.2-1B-Instruct` as the LLM with a frozen Graph model and a trainable projector. This yeilded us a public score of 0.63748. However, for re-ranking, we chose all the models mentioned in Table 1. Interestingly, we observed that both type of re-ranking mentioned before resulted in worst score as compared to the raw-generations given by our best model. We got the public scores of 0.61167 and 0.62388 after using the methods mentioned in Section 4.1 and Section 4.2, respectively.

6 Conclusion

In this study, we developed a molecular graph captioning framework that aligns structural graph embeddings with the semantic space of Large Language Models through a two-stage contrastive and supervised training approach. Our extensive evaluation reveals that, surprisingly, the 1B parameter model consistently outperforms larger 3B and 8B variants, suggesting that smaller, specialized models can be highly effective for this cross-modal task. Furthermore, we demonstrated that post-generation re-ranking strategies, whether based on graph-caption similarity or LLM evaluation, failed to improve upon the raw captions. These findings indicate that a robustly aligned, frozen graph encoder combined with an instruction-tuned projector offers the most reliable performance, eliminating the need for complex post-processing. Although we choose LLaMA as our backbone LLM due to its general reliability, using a domain specific base LLM would likely outperform the current setup. Having access to such domain specific model also opens up the possibility of creating accurate synthetic captions which we found to be unreliable in the case of LLaMA.

References

Xiao Fei, Michail Chatzianastasis, Sarah Almeida Carneiro, Hadi Abdine, Lawrence P. Petalidis, and Michalis Vazirgiannis. Prot2text-v2: Protein function prediction with multimodal contrastive alignment. *CoRR*, abs/2505.11194, 2025. URL https://arxiv.org/abs/2505.11194.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014. URL https://api.semanticscholar.org/CorpusID:6628106.

Zhiyuan Liu, Sihang Li, Yanchen Luo, Hao Fei, Yixin Cao, Kenji Kawaguchi, Xiang Wang, and Tat-Seng Chua. Molca: Molecular graph-language modeling with cross-modal projector and uni-modal adapter, 2024. URL https://arxiv.org/abs/2310.12798.

Ross Taylor, Marcin Kardas, Guillem Cucurull, Thomas Scialom, Anthony Hartshorn, Elvis Saravia, Andrew Poulton, Viktor Kerkez, and Robert Stojnic. Galactica: A large language model for science. *CoRR*, abs/2211.09085, 2022. URL https://arxiv.org/abs/2211.09085.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *CoRR*, abs/1807.03748, 2018.

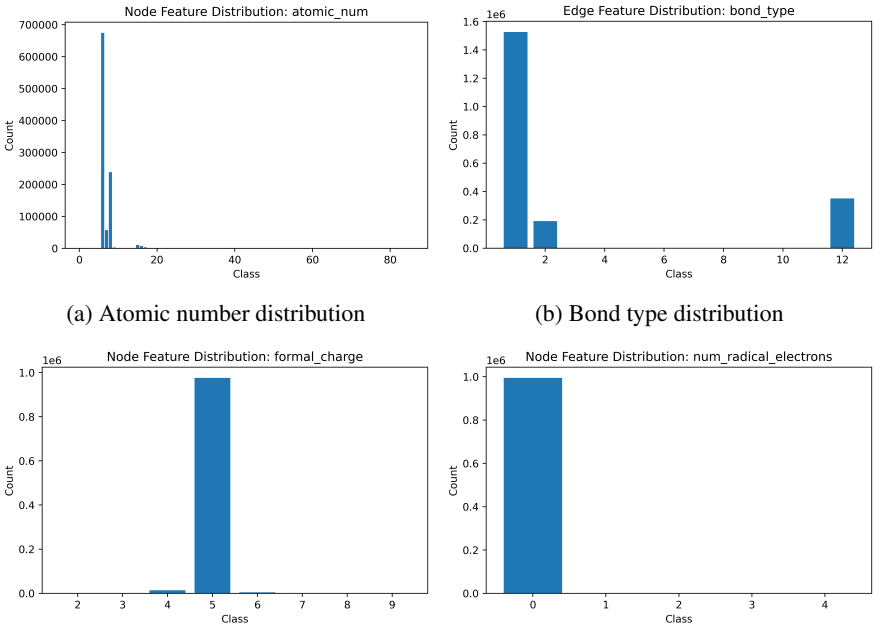
David Weininger. Smiles, a chemical language and information system. 1. introduction to methodology and encoding rules. *Journal of Chemical Information and Computer Sciences*, 28(1):31–36, 1988.

Appendix

A Architecture Details

- **Node features:** Atom type, degree, formal charge, chirality, hybridization, aromaticity, hydrogen count, radical electrons, etc.
- **Edge features:** Bond type, stereochemistry, conjugation, ring membership.
- **Layers:** 5 GINEConv layers with trainable epsilon parameters.
- **Activation:** ReLU between layers, GELU in projection modules.
- **Pooling:** Global mean pooling for graph-level features.
- **Normalization:** BatchNorm for node embeddings, LayerNorm in projection heads.

B Examples of some feature distributions



(a) Atomic number distribution

(b) Bond type distribution

(c) Formal charge distribution

(d) Number of radical electrons

Figure 3: Distributions of selected node and edge features in the dataset.

C LLM based re-ranking prompt template

Reranking Prompt Template

```
<|system|>
You are a precise chemical ontologist. You have been given several candidate captions for a
molecule. Your task is to select the ONE caption that best follows this specific scientific
format:
1. Structural Definition: Begins with "The molecule is..." (e.g., defines stereochemistry,
specific substitutions).
2. Functional Role: Describes biological or industrial roles (e.g., "It has a role as...").
3. Classification: Taxonomical membership (e.g., "It is a member of...", "It is a conjugate
acid of...").
4. Derivation: (Optional but preferred) Describes precursor molecules (e.g., "It derives
from...").
The best caption must be scientifically precise, grammatically rigid, and follow the pattern
of the examples below:
* "The molecule is the S- (more active) ..."
* "The molecule is a member of the class of ..."
Select the candidate that best matches this dense, multi-sentence encyclopedic style.
<|user|>
Here are the candidates:
Candidate 0: [Description 0 Here]
Candidate 1: [Description 1 Here]
Candidate 2: [Description 2 Here]
Evaluate the candidates based on the structural and functional detail pattern described
above. Return ONLY the index number of the best candidate (e.g., 0, 1, 2). Do not explain.
<|assistant|>
The best candidate index is
```