Molecule Retrieval with Natural Language Queries

Julián Alvarez de Giorgi
Institut Polytechnique de Paris
Palaiseau, France
julian.alvarez@telecom-paris.fr

Mohamed Khalil Braham
Institut Polytechnique de Paris
Palaiseau, France
khalil.braham@telecom-paris.fr

Barthélémy Dang-Nhu
ENS Paris-Saclay
Saclay, France
barthelemy.dang-nhu@ens-paris-saclay.fr

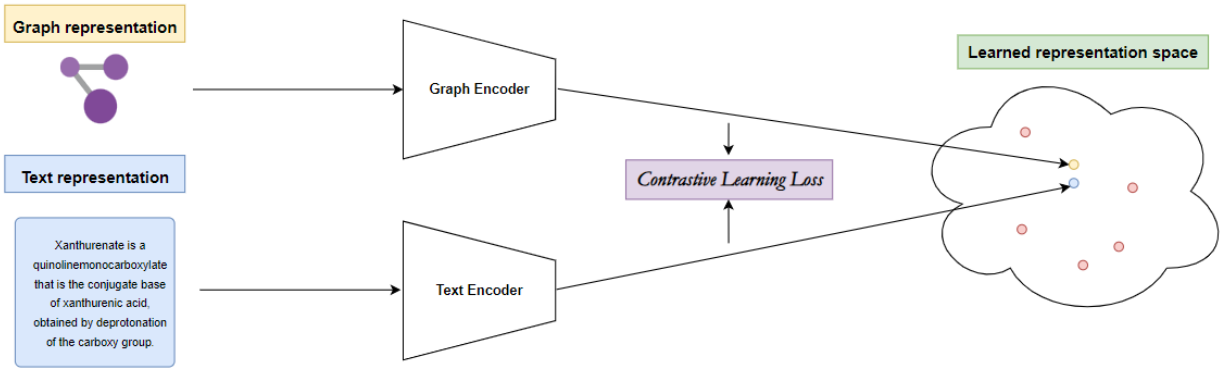

Figure 1. Conceptual diagram of the model structure.

Abstract

This project was carried out as a final project-challenge of the ALTEGRAD (Advance Learning for Text and Graph Data) course of the MVA Master of the ENS Paris-Saclay. The objective of this work was to use the machine-learning machinery to retrieve molecules, represented as graphs, from natural language queries. For the challenge, a list of molecules and a text query were given for this purpose. Two encoders were implemented, a graph encoder based on a GCN (Graph Convolution Network) and GAT (Graph Attention Networks), and a sci-BERT encoder, as the text encoder. Then, a contrastive learning approach was taken to learn a mutual representation space mapping together the text and molecule representations.

CCS Concepts: • **Computing methodologies** → **Neural networks**; *Spectral methods.*

Keywords: Graph Neural Networks; Geometrical Deep Learning; NLP; Contrastive Learning, BART

1 Introduction

The richness and expressiveness of natural language offer a different perspective that goes beyond the structural details of molecules. While molecular structures provide a fundamental basis, the inclusion of natural language descriptions enriches the understanding of molecular behavior, properties, and interactions. Understanding the interplay between molecular structures and their textual representations is a task that only experts on the subject can do. Recent advances in Natural Language Processing (NLP) have opened the way for the development of various multimodal models that integrate natural language with images [6], audio [10] and as in this work, with graph molecules representation [7]. Furthermore, several improvements from [7] were carried, as [5]. In this work we have worked in this direction, applying deep learning to shape the molecule representation space from its natural language description and its graph structure using contrastive learning. The report starts presenting some related work in section 2, then in section 3 the model is presented, section 4 explain the data we train on, further in section 5 the training of the model is explained, and finally in section 6 some conclusions are drawn.

2 Related Work

2.1 SciBERT.

SciBERT [2] is a pre-trained BERT-based model [4] that was trained on a large multi-domain corpus of scientific publications, making use of the full text of the papers, not just the abstracts, making up a total of 1.14 million scientific papers in the domains of biomedicine (82%) and computer

science (12%), making it better suited for processing texts that describe molecular properties. Four different versions of SciBERT were released, as in BERT, they release the case and uncase versions (differentiating uppercase and lowercase letters, or not), and also the variants BASEVOCAB or SCIVOCAB, where the BASEVOCAB versions were fine-tunned from the original BERT training, while the SCIBERT where trained from scratch. We have used the uncase SCIVOCAB version.

They construct a new WordPiece vocabulary on the scientific corpus using the SentencePiece1 library, called SCIVOCAB. The vocabulary size was set to 30K to match the size of BASEVOCAB (original BERT vocab). The resulting token overlap between BASEVOCAB and SCIVOCAB is 42%, which means a significant difference between the frequently used words in the scientific literature and the general literature.

2.2 Graph Attention Networks.

GAT (Graph Attention Network) architecture was presented in [9], they adopted the self-attention idea from transformers [8] to graph convolutional networks (GCN). Let $\mathbf{h} = (h_1, h_2, ..., h_N) \ / \ h_i \in \mathbb{R}^F$ be the input features of each node, and $\mathbf{h'} = (h'_1, h'_2, ..., h'_N) \ / \ h'_i \in \mathbb{R}^{F'}$. In the general GCN framework, a GCN layer can be described as:

$$h'_i = \sigma\left(\sum_{v_j \in \mathcal{N}(v_i)} \alpha_{ij} \mathbf{W} h_j\right) \tag{1}$$

Where:

- v_i is the node i.
- $\mathbf{W}$ is a learnable matrix with size $F' \times F$.
- α_{ij} specifies the weighting factor (importance) of node j's features to node i.
- σ is a non-linearity.

GATs compute the weighting factors making use of the node features, first, the attention coefficients are computed:

$$e_{ij} = a\left(\mathbf{W}h_i, \mathbf{W}h_j\right) = LeakyReLu\left(\mathbf{a}\left[\mathbf{W}h_i || \mathbf{W}h_j\right]\right) \tag{2}$$

Then, $a(.)$ is called the attention mechanism and is a single feed-forward network with $2 \times F'$ parameters (vector $\mathbf{a}$, and $LeakyReLu$ activation function. Finally, the weighting factor is computed as follows:

$$\alpha_{ij} = softmax_j(e_{ij}) = \frac{exp(e_{ij})}{\sum_{k \in \mathcal{N}(v_i)} exp(e_{ik})} \tag{3}$$

Furthermore, as in Transformers, we can have multi-head attention mechanism, where for a K-head attention we have:

$$h_j = ||_{k=1}^{K} \sigma\left(\sum_{j \in \mathcal{N}(v_i)} \alpha_{ij}^k \mathbf{W}^k h_j\right) \tag{4}$$

Later, [3] proposed an improvement on this attention mechanism, modifying the order of internal operations in GAT and introduced GATv2, which has a strictly more expressive attention mechanism. Building what they define as a *dynamic* graph attention network. The changes are the following:

- GAT [9]: $e_{ij} = LeakyReLu\left(\mathbf{a}\left[\mathbf{W}h_i || \mathbf{W}h_j\right]\right)$
- GATv2 [3]: $e_{ij} = \mathbf{a}\ LeakyReLu\left(\mathbf{W}\left[h_i || h_j\right]\right)$

2.3 Contrastive learning.

Contrastive learning is a type of self-supervised learning technique, which its primary goal is to teach a model to understand the similarities and differences between different examples in a dataset. This technique has been successfully used to enhance the model's ability to understand and represent the complex relationships between different modalities. In [7], they have proven its effectiveness for learning joint representations of molecular graphs and and natural language descriptions.

3 Model

3.1 Baseline Model

The baseline model for the challenge, employs DistilBERT as the text encoder and GCN as the graph encoder. To obtain the corresponding graph, cosine similarities between each text description embedding and all molecule embeddings are computed. The model is trained for 5 epochs, minimizing the contrastive loss between the graph and text representations.

This baseline model reaches an MRR score of 0.348. Its integration of both text and graph representations sets the foundation for our subsequent enhancements and refinements. Our iterative approach builds upon this framework, incorporating additional techniques, fine-tuning parameters, or exploring alternative architectures to further elevate the model's performance and effectiveness in accurately retrieving molecules from natural language queries.

3.2 Our Model

Our model consists of two encoders, a graph encoder, and a text encoder, where with contrastive learning we have shape a learned representation space mapping the text representation embeddings and the graph representation embeddings, making the representations wich correspond to the same molecule close together and the differents far away from them. This is conceptually illustrated in Fig.1.

3.2.1 Text Encoder. As a text encoder, we have used a BERT architecture, that is, a multi-layer bidirectional Transformer encoder, presented in [8]. We've used the BERTbase release version, which consists of 12 transformer layers with a hidden dimension of 768 and 12 attention heads, making a total of 110M of parameters.

We have used the weights of the uncased Sci-BERT model, described in 2.

3.2.2 Graph Encoder. In pursuit of optimizing performance and achieving robust results, we employed an Ensemble Model. The ensemble approach capitalizes on the diversity and complementary strengths of the three basic individual models, leveraging their distinctive features to enhance overall predictive accuracy.

We compute the mean of the output representation of 3 different graph encoders, two GATs, one based on GAT and another on GATv2, and a third which is a GCN architecture. the 3 encoders follow the same structure, but the convolutional layers change, Fig. 2.

$$z_G = \frac{1}{3}(z_G^{GAT} + z_G^{GATv2} + z_G^{GCN}) \tag{5}$$

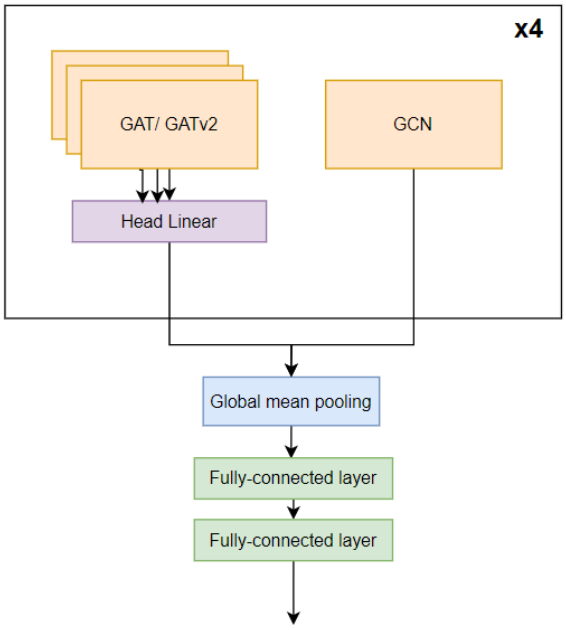

Figure 2. Structure of the 3 graph encoders used in our model, based on GATconv, GATv2conv, and GCN layer.

3.2.3 GAT. This GAT encoder is based on the GAT described in section 2, on the first version of it. It has 4 GAT convolutional layers with concatenation aggregation over the multi-head layers, each GAT convolutional layer is then followed by a fully-connected layer of size $[ghd \times n\_heads, ghd]$ to retrieve the graph hidden feature dimension (ghd), and its ReLu activation function. A mean aggregation is done on the node features to find the graph representation, and finally, 2 fully-connected layers take the embedding from ghd to a hidden dimension and then to the final output dimension.

The input feature size is of 300, then the hidden dimension of each convolutional layer is 1024, the hidden dimension of the two fully connected layer is 300, and the output dimension is 1024. Each GAT convolutional layer has 10 attention heads.

3.2.4 GATv2. This GATv2 is similar to the one explained above, the only difference is that we've implemented the improved version of the GAT convolutional layer, also explained in section 2. All the parameters are the same as the GAT encoder.

3.2.5 GCN. This encoder follows the same structure described before but with a simple GCN convolutional layer, hence we don't have the layer aggregation, all the different representations from the different heads. The dimension of the spaces, a hidden dimension between convolutional layers, and the dimension of the last 2 fully connected layers are the same as before.

3.2.6 GATv2 with residuals connections and jumping knowledge. We have also implemented a graph encoder with the same structure as before, using GATv2 convolutions, but incorporating residual connections on the head linear layer. Also, we have tried incorporating jumping knowledge, presented in [11]. But due to computational power, we couldn't test it, but we think it's the next step to take.

4 Dataset

For training the model, we've used 26408 samples of molecules, with both representation, their graph representation, and their description.

A validation dataset with 3301 samples, with both, text description and graph representation.

For test proposes we've used 3301 samples, with both, text description and graph representation.

4.1 Data Augmentation

We've done 2 data augmentation on the graphs representation for each molecule, which is a common approach in this kind of task, and is recommended to help the contrastive learning goal, it gives more information about which types of graphs should be near each other, and which molecules are similar. Without data augmentations, we could end up with a learned space where each molecule is well-matched between graph and text representation, but there's no real order, and navigating along the space could not be coherent. The two more pertinent graph augmentations for molecule are node dropping and random walk subgraph because those augmentations doesn't alter much molecule properties[12].

4.1.1 Node Dropping. In the context of molecular graph data augmentation, this technique involves selectively retaining only a fraction, denoted as p, of randomly chosen nodes within each molecule. The impact of this process extends beyond individual nodes, as the edges connected to the dropped nodes are also removed. This approach introduces a

form of sparsity into the molecular graph, mimicking scenarios where certain molecular components might be missing or undergo structural changes. The choice of $p = 0.9$ signifies that, on average, 90% of the nodes within a molecule are retained during the augmentation process. However, due to time constraints, computational limitations, and restricted GPU access, the full integration of this augmentation into our training data was not feasible.

4.1.2 Random Walk Subgraph. This augmentation technique involves initiating the process by randomly selecting a node within a molecule. Subsequently, nodes are incrementally chosen from the neighborhood of the previously selected nodes, continuing until a proportion p of the entire graph is obtained. This approach reflects the exploration of molecular structures by iteratively traversing neighboring nodes, capturing local interactions and relationships. The parameter $p = 0.9$ indicates that the augmentation process aims to encompass 90%of the graph. Unfortunately, due to time constraints, computational limitations, and a lack of access to GPU resources, the incorporation of this augmentation into our training data was not realized.

4.1.3 Results. In summary, while these augmentation techniques hold promise for enhancing the robustness and generalization capabilities of our model, practical constraints prevented their full integration into the training pipeline. The chosen values for p in both techniques provide a balance between introducing diversity into the data and maintaining a sufficiently large dataset for training, albeit without compromising the model's performance.

5 Training

To train our model we have used contrastive learning, we have tried several losses:

- CE, Cross-Entropy loss:

$$\mathcal{L}_{CE} = -log\left(\frac{exp(sim(z_i, z_j)}{\sum_{k=1,k\neq i}^{N} exp(sim(z_i, z_k))}\right)$$

- NT-Xent, Loss Normalized temperature scaled Cross Entropy Loss:

$$\mathcal{L}_{NT\_xent} = -log\left(\frac{exp(sim(z_i, z_j)/\tau}{\sum_{k=1,k\neq i}^{N} exp(sim(z_i, z_k)/\tau)}\right)$$

- InfoNCE, Noise-Contrastive Estimation:

$$\mathcal{L}_{InfoNCE} = -log\left(\frac{exp(q * k_+)}{\sum_{i=0}^{K} exp(q * k_i)}\right)$$

Finally, we end up using a combination of CE loss and InfoNCE loss:

$$\mathcal{L} = \beta\mathcal{L}_{CE} + (1 - \beta)\mathcal{L}_{InfoNCE} \tag{6}$$

We use $\beta = 0.1$.

5.1 Model Selection

We performed a model selection task by choosing the architecture that demonstrated the highest Label Ranking Average Precision (LRAP) on the sampled validation set. This rigorous criterion ensures that the chosen model not only generalizes well on the validation data but also excels in preserving the relative ranking of labels.

5.2 Hyperparameter tunning

Seeking the best performance possible with the tools, data, and resources we had, we have done a hyperparameter search using Optuna [1] package, looking for the best number of epochs, batch size, graph learning rate, hidden dimension of the two last fully-connected layers of the graph encoders, and the text learning rate.

We have used the Adam optimizer with a reduced learning rate on the validation loss as a metric, on 'min' mode, with a factor of 0.7, patience of 3 epochs, and minimum learning rate of 1e-7. The final hyperparameters were:

- Epochs = 100.
- Batch size = 31.
- hidden dimension = 300.
- Graph learning rate = $5e-5$.
- Text learning rate = $5e-6$.
- Weight decay = 0.01.
- betas = (0.9, 0.999).

All the training was done on a machine with a 3.00GHz Intel Xeon Gold 6136 CPU, 187GB of RAM, and an NVIDIA Tesla V100 GPU with 16GB of RAM, taking 10 minutes per epoch approx, a total amount of $\sim$ 16.7 hours.

6 Conclusion

We were able to build a model capable of retrieving the molecule from the text description with a mean reciprocal rank of 0.8915. We conclude that this is a satisfactory result, but we would like to further work and test other architectures, and incorporate the augmentations as part of our training data. We also believe that this work is a good basis to build on, and we value the fact of having a direction to continue working to improve it in different aspects of the model, with ideas and different implementations that we have been mentioning throughout this report.

References

[1] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. 2019. Optuna: A Next-Generation Hyperparameter Optimization Framework. In *The 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2623–2631.

[2] Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. SciBERT: A Pretrained Language Model for Scientific Text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun

Wan (Eds.). Association for Computational Linguistics, Hong Kong, China, 3615–3620. https://doi.org/10.18653/v1/D19-1371

[3] Shaked Brody, Uri Alon, and Eran Yahav. 2021. How Attentive are Graph Attention Networks? *CoRR* abs/2105.14491 (2021). arXiv:2105.14491 https://arxiv.org/abs/2105.14491

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *CoRR* abs/1810.04805 (2018). arXiv:1810.04805 http://arxiv.org/abs/1810.04805

[5] Romain Lacombe, Andrew Gaut, Jeff He, David Lüdeke, and Kateryna Pistunova. 2023. Extracting Molecular Properties from Natural Language with Multimodal Contrastive Learning. arXiv:2307.12996 [cs.LG]

[6] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning Transferable Visual Models From Natural Language Supervision. *CoRR* abs/2103.00020 (2021). arXiv:2103.00020 https://arxiv.org/abs/2103.00020

[7] Bing Su, Dazhao Du, Zhao Yang, Yujie Zhou, Jiangmeng Li, Anyi Rao, Hao Sun, Zhiwu Lu, and Ji-Rong Wen. 2022. A Molecular Multimodal Foundation Model Associating Molecule Graphs with Natural Language. arXiv:2209.05481 [cs.LG]

[8] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. *CoRR* abs/1706.03762 (2017). arXiv:1706.03762 http://arxiv.org/abs/1706.03762

[9] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. arXiv:1710.10903 [stat.ML]

[10] Lukas Wolf, Greta Tuckute, Klemen Kotar, Eghbal Hosseini, Tamar Regev, Ethan Wilcox, and Alex Warstadt. 2023. Whis-BERT: Multimodal Text-Audio Language Modeling on 100M Words. arXiv:2312.02931 [cs.CL]

[11] Keyulu Xu, Chengtao Li, Yonglong Tian, Tomohiro Sonobe, Ken-ichi Kawarabayashi, and Stefanie Jegelka. 2018. Representation Learning on Graphs with Jumping Knowledge Networks. *CoRR* abs/1806.03536 (2018). arXiv:1806.03536 http://arxiv.org/abs/1806.03536

[12] Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. 2021. Graph Contrastive Learning with Augmentations. https://doi.org/10.48550/arXiv.2010.13902 arXiv:2010.13902 [cs].

A Online Resources

The source code is available at https://github.com/JulianAlvarezdeGiorgi/Molecule-Retrieval-with-Natural-Language-Queries.

Received 3 February 2024