Molecule Retrieval with Natural Language Queries

Yassine BENIGUEMIM
Ecole Polytechnique
yassine.beniguemim@polytechnique.edu

Abstract

This project explores the intersection of Natural Language Processing (NLP) and molecular retrieval, employing machine learning techniques to extract molecules from textual queries. The challenge involves discerning the correct molecule from a list of graphs without accompanying text or references. This demands the fusion of structured knowledge from text and chemical properties within molecular graphs. The main approach involves co-training a text encoder and a molecule encoder using contrastive learning with a sentence transformer as the text encoder and Dynamic Graph Attention Models as the base of the graph encoder along with node embedding from Mol2vec model. This entails simultaneously training two specialized encoders to handle textual data and molecular structures. Through contrastive learning, the model learns to map similar text-molecule pairs closer together and dissimilar pairs apart. Our presented approach achieves a notable label ranking average precision score of 0.8064% for the test set in the public leaderboard, which surpasses the baseline model of 0.3480%. This highlights the efficacy of our model in seamlessly integrating textual and structural information, showcasing the potential of contrastive learning in addressing interdisciplinary challenges at the convergence of NLP and molecular retrieval.

1. Introduction

The synergy between NLP and molecular retrieval represents a challenging frontier in information retrieval, particularly in the context of retrieving molecules represented as graphs through natural language queries. In this study, we address this intricate task, extending beyond existing paradigms to handle scenarios where explicit textual information about molecules is absent.

The landscape of related works reveals notable advancements in molecular retrieval, with a focus on leveraging structured data and chemical properties. The main paper that tackles a challenge similar to what we study is called Text2Mol [3], which focuses on retrieving molecules using natural language descriptions as queries. The research involves creating a paired dataset of molecules and their corresponding text descriptions to learn a common semantic embedding space for retrieval. The study also presents a cross-modal attention-based model for explainability and reranking, as well as an ensemble approach that significantly improves their results. The research utilizes a BERT-based text encoder and an MLP-GCN model for molecule retrieval based on language descriptions. The proposed approach achieves promising results in cross-modal molecule retrieval with natural language queries. One year after, in 2022, a molecular multimodal foundation model, named MoMu [6], pre-trained on the paired multi-modal data consisting of molecule graphs and their weakly-related biochemical descriptions retrieved from publicly available Scientific Citation Index (SCI) papers. The pre-trained MoMu exhibits strong generalization abilities in a wide range of downstream tasks, including cross-modal molecule retrieval, molecule caption, zero-shot molecule generation, and molecular property prediction. A year after, GIMLET [8] proposes a nature language instruction-based graph zero-shot learning for molecule tasks. The proposed approach extends large language models to handle graph and text data by applying the transformer mechanism with generalized position embedding and decoupled attention.

Indeed, the particularity of our project is that we are confronting the challenge of molecular retrieval without textual references. This necessitates the integration of both structured knowledge from text and inherent chemical properties within molecular graphs. Our approach is more closer to Text2Mol but with some specificities that we will discuss in this report.

2. Methodology overview

In the context presented earlier, the objective revolves around constructing a model that harnesses two primary constituents, namely text and graph encoders. Each component within our model possesses a distinct architecture, the details of which will be discussed in the subsequent section.

The principal model synergizes both components

through a contrastive loss mechanism, mandating the projection of embeddings from both models into a shared space. The optimization of this loss entails ensuring that the textual information about any specific graph is significantly proximal compared to the textual information of dissimilar graphs. Next, we will see each component in details.

2.1. Text encoding model

This segment undertakes the task of training a model capable of mapping each textual description of the test data, containing 3301 texts. The training is executed on a dataset comprising 26408 samples in the training set and 3301 in the validation set. To have more visibility of the textual information provided, here is an example of the textual description of a particular molecule in the training dataset.

> Phosphorous acid is a phosphorus oxoacid. It is a conjugate acid of a dihydrogenphosphite. It is a tautomer of a phosphonic acid.

In this context, the baseline employs the distill-Bert model, a streamlined variant of the Bert model with reduced parameters. In our approach, we opt for a more simple and cost-efficient architecture (without fine-tuning), which is the pertinent iteration of Bert specifically designed for embedding sentences while preserving comprehensive semantic meaning, which is a crucial factor in our scenario. The chosen text embedding architecture is grounded in the model "sentence-transformers/all-MiniLM-L6-v2," derived from the paper "Sentence-Bert: Sentence Embeddings using Siamese BERT-Networks" [5] and implemented through the Hugging Face Transformers library.

The model architecture encompasses a singular linear layer, configured with a hidden size of 256. Notably, the MiniLM-L6-v2 model adopts a transformer-based architecture, pre-trained on an extensive corpus of diverse textual data. Its distinctive attribute lies in its ability to capture intricate contextual relationships within sentences, facilitating the extraction of nuanced semantic information.

In the overarching context, Sentence-BERT (SBERT) emerges as a comprehensive framework designed for the computation of sentence embeddings, leveraging the BERT model's capabilities. Primarily intended for diverse downstream tasks, SBERT achieves computational efficiency through the incorporation of Siamese Networks [5]. This architectural paradigm entails the integration of two or more identical subnetworks, as depicted in Figure 1, utilized for generating feature vectors for each input and subsequent comparative analysis.

The fundamental core of the model involves the provision of three input items, where two are intentionally similar (anchor and positive samples), while the third remains unrelated (a negative example). The model's objective centers on learning to discern the similarity between items by

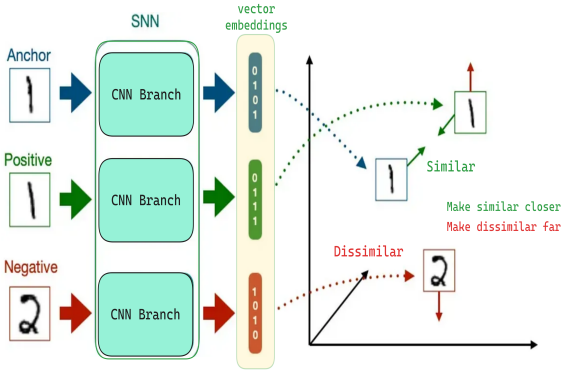

Figure 1. Siamese network design involves the provision of three input items, where two are intentionally similar (anchor and positive samples), while the third remains unrelated (a negative example)

minimizing the distance between similar elements and concurrently increasing the distance between dissimilar ones, facilitated by the Triplet loss function.

Significantly, mean pooling constitutes an integral facet of our architecture, functioning as a mechanism for aggregating token embeddings and generating a fixed-size representation. This means pooling, synergized with an attention mask, adeptly handles variable-length sequences, thereby endowing the model with adaptability across a spectrum of textual data.

During the forward pass, the input text undergoes processing through the MiniLM-L6-v2 model, where mean pooling is applied to derive a fixed-size representation. Subsequently, this representation transforms a linear layer and layer normalization, followed by rectified linear unit (ReLU) activation. The resultant output is then normalized using L2 normalization, yielding a compact and semantically enriched representation of the input text.

2.2. Graph encoding model

In our comparative study, the established baseline employed the conventional Graph Convolutional Network (GCN) as a reference model with node embedding computed with Mol2vec, which is an unsupervised machine learning approach for learning vector representations of molecular substructures. Subsequently, our study undertook a benchmarking analysis of two primary architectures: GraphSAGE [4] and Graph Attention Model (GAT) [7]. GraphSAGE, or Graph Sample and Aggregation, operates by sampling and aggregating information from a node's local neighborhood, generating representative embeddings for each node. This methodology, while effective, in some cases exhibits limitations in capturing intricate relationships within the graph due to its reliance on simplified aggregation techniques. In contrast, the Graph Attention Model

(GAT) surpasses these limitations by introducing a more sophisticated attention mechanism. GAT enables nodes to selectively attend to specific parts of their neighborhood during the aggregation process, thereby capturing nuanced structural features and intricate relationships within the graph. Our comprehensive evaluation revealed that GAT consistently outperformed GraphSAGE in terms of encoding graph structures, demonstrating its superiority in capturing and leveraging fine-grained details. Consequently, our best-performing approach in the graph encoding component of our model was achieved through the integration of a better version of the Graph Attention Model, which we will explain later in Section 2.2.2.

Unlike traditional GCN, which relies on a simple aggregation of neighborhood information, GAT introduces a more sophisticated attention mechanism. This mechanism allows nodes to selectively attend to different parts of their neighborhood during the aggregation process, effectively assigning distinct importance weights to neighboring nodes. This adaptability enables the model to focus on relevant structural features and capture intricate relationships within the graph. The utilization of attention mechanisms in GAT enhances the model's capacity to discern fine-grained details. Our experimental results demonstrate the efficiency of the GAT-based approach, showcasing its superiority over the baseline GCN and the GraphSAGE model in capturing and leveraging intricate graph structures for enhanced performance in molecular retrieval using natural queries.

To delve into greater detail, let us elucidate the operational intricacies of the Graph Attention Model (GAT).

2.2.1 Graph Attention Networks (GAT)

To understand graph attention networks. Let's revisit the node-wise update rule of GCNs. As we can see, we have this coefficient $\frac{1}{\sqrt{|\mathcal{N}_i||\mathcal{N}_j|}}$ which is multiplied in our projection of the node features. The coefficient is derived from the degree matrix of the graph and is heavily dependent on the structure of the graph. Intuitively, it represents how important the node's j features are for node i.

$$h_i^{(l)} = \sigma\left(\sum_{j\in\mathcal{N}_i} \frac{1}{\sqrt{|\mathcal{N}_i||\mathcal{N}_j|}} W h_j\right)$$

The main idea behind GAT is to compute that coefficient implicitly rather than explicitly as GCNs do. That way we can use more information besides the graph structure to determine each node's "importance" through the consideration of the coefficient to be a learnable attention mechanism.

The paper behind GAT suggests that the coefficient, denoted as a_{ij}, should be computed based on node features, which are then passed into an attention function. Finally, the softmax function is applied in the attention weights a_{ij} to result in a probability distribution. Mathematically we have:

$$a_{ij} = \text{attention}(h_i, h_j)$$

$$a_{ij} = \frac{\exp(a_{ij})}{\sum_{k\in\mathcal{N}_i}\exp(a_{ik})}$$

Visually this can be seen on the left side in figure 3

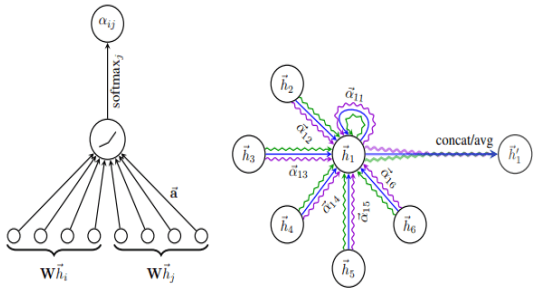

Figure 2. GAT: novel neural network architectures that operate on graph-structured data, leveraging masked self-attentional layers to address the shortcomings of prior methods based on graph convolutions or their approximations.

2.2.2 Dynamic Graph Attention Variant (GATv2)

The GATv2 operator from the "How Attentive are Graph Attention Networks?" paper [2], which fixes the static attention problem of the standard `GATConv` layer. The main philosophy of this new variant is that the linear layers in the standard GAT are applied right after each other thus the ranking of attended nodes is unconditioned on the query node. In contrast, in `GATv2`, every node can attend to any other node.

$$x_i' = \alpha_{i,i}\odot x_i + \sum_{j\in\mathcal{N}(i)} \alpha_{i,j}\odot x_j,$$

where the attention coefficients $\alpha_{i,j}$ are computed as

$$\alpha_{i,j} = \frac{\exp\left(a^\top \text{LeakyReLU}(\theta(\mathbf{W}x_i + \mathbf{W}x_j))\right)}{\sum_{k\in\mathcal{N}(i)\cup\{i\}}\exp(a^\top \text{LeakyReLU}(\theta(\mathbf{W}x_i + \mathbf{W}x_k)))}.$$

If the graph has multi-dimensional edge features $e_{i,j}$, the attention coefficients $\alpha_{i,j}$ are computed as

$$\alpha_{i,j} = \frac{\exp\left(a^\top \text{LeakyReLU}(\theta(\mathbf{W}x_i + \mathbf{W}x_j + \mathbf{W}_e e_{i,j}))\right)}{\sum_{k\in\mathcal{N}(i)\cup\{i\}}\exp(a^\top \text{LeakyReLU}(\theta(\mathbf{W}x_i + \mathbf{W}x_k + \mathbf{W}_e e_{i,k})))}.$$

For more details, in our research, we employed a graph encoder with an architecture comprised of multiple layers, including GATv2Conv layers for graph attention mechanisms and linear layers for subsequent transformations. The

ReLU activation function is applied throughout the convolutional layers to introduce non-linearity, facilitating the extraction of complex relationships within the graph data. Additionally, Layer Normalization is utilized to enhance the stability and convergence of the model. Our graph encoder concludes with a readout layer that computes the global mean over the graph representations, followed by a multilayer perceptron (MLP) to map the aggregated features to the desired output dimension. This design aims to effectively capture and distill the intricate molecular structures, enabling the model to learn meaningful representations. For more details, see our code part.

3. Main experiments

3.1. Evaluation

The performance of our models is assessed using the label ranking average precision score:

$$\text{LRAP}(y, \hat{y}) = \frac{1}{n_{\text{samples}}} \sum_{i=0}^{n_{\text{samples}}-1} \frac{1}{||y_i||_0} \sum_{j:y_{ij}} \frac{|L_{ij}|}{\text{rank}_{ij}}$$

Where $L_{ij} = \{k : y_{ik} = 1, \hat{y}ik \geq \hat{y}ij\}$, rank$ij = |\{k : \hat{y}ik > \hat{y}_{ij}\}|$, and $|\cdot|$ computes the cardinality of the set. $||\cdot||_0$ is the ℓ_0 norm.

Label ranking average precision (LRAP) is the average over each ground truth label assigned to each sample, of the ratio of true vs. total labels with lower scores. The goal is to give a better rank to the label associated to each sample. As there is exactly one relevant label per sample, label ranking average precision is equivalent to the mean reciprocal rank. Where MRR assesses how well the model ranks the correct descriptions for each molecule in the list of queries. Indeed, for each molecule-query pair, the reciprocal rank is computed based on the position at which the correct description is ranked. The reciprocal rank is the inverse of this position. At the end, MRR then calculates the average of these reciprocal ranks across all molecule-query pairs. Technically, a higher MRR indicates that, on average, the correct description for each molecule is ranked closer to the top of the list of queries. In contrast, a lower MRR suggests that, on average, the correct descriptions are ranked further down the list, indicating a less effective association between molecules and their descriptions.

3.2. Loss

The Contrastive Loss is calculated using a Cross-Entropy (CE) loss function. For each instance, the loss evaluates the alignment of the text and graph embeddings, considering the ground truth labels. Positive samples, where text and graph embeddings should be close, contribute to minimizing the loss, while negative samples, indicating dissimilarity, contribute to increasing the loss. The Contrastive Loss encourages the model to learn embeddings that effectively capture semantic relationships between textual and graphical representations, thereby improving the overall performance of the embedding model.

3.3. Main results

In our model training phase, we employed the Adam optimizer with a learning rate of 3e-5, a weight decay of 0.01, and betas set to (0.9, 0.999). The training process spanned 60 epochs, utilizing a batch size of 32. The embedding space dimensions were set to 256 for both text and graph embeddings. In the graph encoder, node embeddings are generated using Mol2vec with the size of 300, with an additional 512 hidden layers for the MLP of 3 layers coming after three dynamic convolutional attention layers, along with 600 channels for graph hidden layers. For the text encoder, on top of a sentence-transformer layer we have one linear layer followed by linear normalization. The architecture of the model is built using the Pytorch framework, and the model training was executed on a GPU P100, completed in 6 hours to get the accuracy of 0.8064 in the public leaderboard of the Kaggle competition.

	Time	Epochs	Last_val_loss	Test_loss
GCN	3h	40	0.45	0.4497
GraphSAGE	3h	40	0.33	0.5507
GAT	3h	40	0.20	0.6704
Gatv2	3h	40	0.14	0.8064

Table 1. Some main experiments over the main architectures tested, runned in a machine of GPU P100 with the same conditions as before and each time an MLP of 3 layers is added after the convolutional layers and 1 linear layer after the sentence-transformer layer

4. Other experiments

As previously mentioned, node embeddings were initially generated using Mol2vec. An alternative exploration involved experimenting with node embeddings generated through GraphSAGE. This process entailed creating embeddings for nodes across 102,981 graphs, resulting in the storage of 3,138 node vectors. Each graph was individually processed, and the embeddings were saved in a file. The challenge emerged due to the vector length of 300, translating to 1 gigabyte of storage for every 500 processed graphs. The intended approach aimed to compute vector embeddings for all nodes within each graph, and as multiple graphs shared common nodes, the objective was to average the embeddings for each shared node across the graphs. However, difficulties arose concerning file optimization and embedding computation. The existing code lacked parallelization, and running the model sequentially over 102,981

graphs was estimated to require approximately 28 hours on a 10-core machine. Although parallel computation challenges were addressed with multithreading processing, storing final embeddings before averaging would demand a substantial amount of terabytes. Consequently, this experiment, following the outlined logic, proved unfeasible with the current technique. The code employed for this experiment is provided in the code section.

5. Conclusions

In summary, our study introduces an information retrieval pipeline tailored for molecular retrieval via natural language queries. Our approach involves the application of contrastive representation learning to both a Sentence-transformer text encoder and a GATv2-based molecule encoder, as evidenced by the mean reciprocal rank metric when compared to the baseline.

Our model achieves 0.8064 MRR (Mean Reciprocal Rank) in the public leaderboard of the kaggle challenge ALTEGRAD-2023, where the baseline is 0.3480 MRR.

While our current model displays promising outcomes, potential enhancements can be explored through refined encoder architectures. Incorporating models like Scibert [1], as demonstrated in [3], trained specifically on scientific texts, may improve text encoding performance. Our goal was to implement other architecture from what [3] implemented, along with getting good results. Additionally, investigating alternative graph encoder architectures, such as graph isomorphism networks or graph diffusion models, could contribute to further refinement.

Another avenue worth exploring involves leveraging query augmentation techniques. Augmenting text descriptions for molecule queries with external knowledge graphs from scientific papers, like OpenAlex, we believe it has the potential to enrich the training data, incorporating additional semantic tokens.

In conclusion, delving into the impact of diverse architectural choices on our model's performance could provide valuable insights for future developments in the field of molecular information retrieval.

6. Acknowledge

We would like to acknowledge Hugging Face for their contributions to the transformers library, and Pytroch-geometric for easy-to-use API for implementing graph architectures, which greatly facilitated our work on this project.

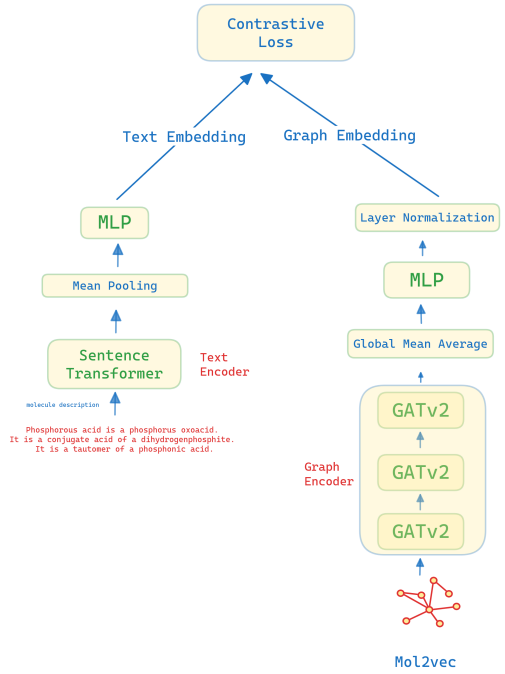

Figure 3. The architecture of molecule retrieval with natural language queries: a model of two encoders, text and graph encoders aligned with the contrastive loss for insuring pairs of text-graph are closer than other no associated text-graph pairs. Graph Attention Model handling node features coming from mol2vec model along with Sentence-Transformer model for handling molecule descriptions.

References

[1] Iz Beltagy, Kyle Lo, and Arman Cohan. Scibert: A pretrained language model for scientific text, 2019. 5

[2] Shaked Brody, Uri Alon, and Eran Yahav. How attentive are graph attention networks?, 2022. 3

[3] Carl Edwards, ChengXiang Zhai, and Heng Ji. Text2Mol: Cross-modal molecule retrieval with natural language queries. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 595–607, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. 1, 5

[4] William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs, 2018. 2

[5] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks, 2019. 2

[6] Bing Su, Dazhao Du, Zhao Yang, Yujie Zhou, Jiangmeng Li, Anyi Rao, Hao Sun, Zhiwu Lu, and Ji-Rong Wen. A molecular multimodal foundation model associating molecule graphs with natural language, 2022. 1

[7] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks, 2018. 2

[8] Haiteng Zhao, Shengchao Liu, Chang Ma, Hannan Xu, Jie Fu, Zhi-Hong Deng, Lingpeng Kong, and Qi Liu. Gimlet: A unified graph-text model for instruction-based molecule zero-shot learning, 2023. 1