Altegrad - Data Challenge Report
Molecule Retrieval with Natural Language Queries

Louise Durand-Janin
ENS Paris-Saclay
louise.durand--janin@ens-paris-saclay.fr

Elsa Azoulay
ENS Paris-Saclay
elsa.azoulay@ens-paris-saclay.fr

Axel Nguyen–Kerbel
ENS Paris-Saclay & CentraleSupélec
axel.nguyen-kerbel@student-cs.fr

February 2024

The goal of this challenge is to retrieve the corresponding molecule given a text query. One way to achieve this is co-training a text encoder and a graph encoder using contrastive learning. Contrastive learning seeks to create a mapping where similar pairs of queries and molecules are represented by closely aligned latent embeddings. In other words, it aims to ensure that comparable entities share similar characteristics in their learned representations. The performances of the models are assessed using the label ranking average precision score (**LRAP**)

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

Each model assigns a score to each label for a given sample, and the labels are then ranked based on these scores. For each ground truth label associated with a sample, LRAP calculates a ratio. The ratio is the fraction of true labels among the labels with a lower score.

Note:
Due to challenges in obtaining access to GPUs for this project, which were evidently necessary, we faced limitations in testing all the architectures we originally intended. As a result, we will also elaborate on what we cound not run and provide the associated code.

1 Baseline

The baseline provided for this challenge uses a pretrained *DistillBert* encoder and a 3-layers Graph Convolutional Networks (GCN) as graph encoder. The baseline trained on 5 epochs achieved a LRAP score of 0.3480 on the test set.

We tried to modidify the number of epochs the baseline was trained on and we obtained the curves of Figure 1.

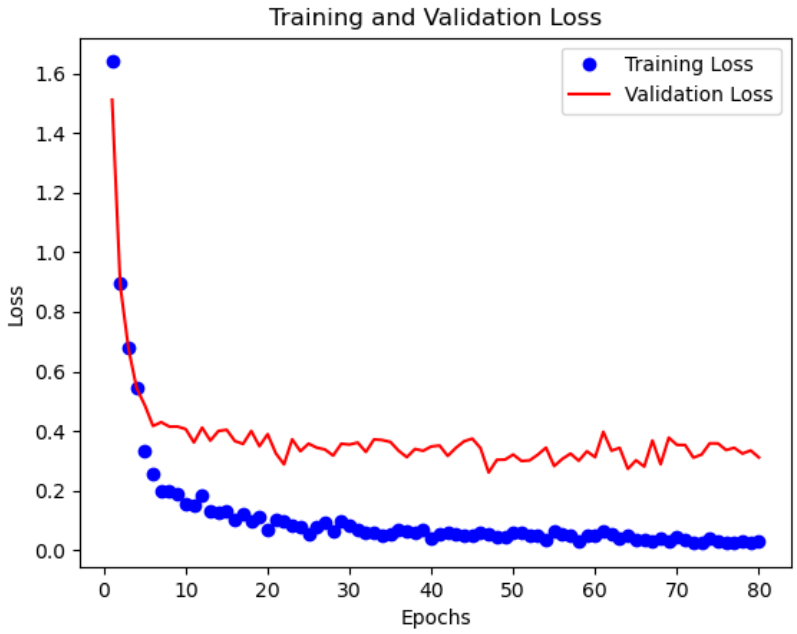

Figure 1: Training and validation losses for training of baseline model with 80 epochs.

As we can see, the losses decrease rapidly. This leads to a score of 0.5426 on the test set, which is an improvement compared to the score obtained by training on 5 epochs.

2 Graph Encoder using GATConv Layers

In Graph Convolutional Networks, every nodes are treated with equal importance. However, in the context of molecules, certain nodes may carry more significance than others to differentiate molecules. Thus, our very first idea was to enhance the graph encoder by replacing the classical GCN by Graph Attention Network (GATs). In GAT Neural Networks, every node updates its representation by attending to its neighbors using its own representation as the query. Each node is then represented using a weighted average of its neighbors. Our first architecture is inspired by the work of Hui Liu and Deng [2022]. It uses 2 GATConv layers with the first one having 10 attention heads for graph encoder. We used exponential linear unit activation functions and added dropout layers to prevent overfitting. We also complexified our network adding 2 Linear

layers at the end. We did not make any changes on text encoder at first to see how pur graph encoder would perform compared to the baseline. One epoch ran in approximately 11 minutes. In 50 epochs, the best validation loss that we got was 0.14 at epoch 30, while our mean training loss was around 0.04. We made predictions using the provided script with cosine similarity, we obtained a test score of 0.59.

3 Graph Encoder using GATv2Conv Layers

As highlighted by Brody et al. [2022], GATConv networks compute only a restricted "static" form of attention. It thus proposes a dynamic attention network called GATv2Conv. We wanted to investigate whether GATv2Conv layers could offer improved performance for our specific task. To explore this, we replicated the architecture used previously, replacing the GATConv layers with the newly introduced GATv2Conv layers. It appears to have quite similar results on 50 epochs. The best validation loss was 0.2 with one epoch running in around 10 minutes. At the end, we obtained a score of 0.55 on the test set.

4 Text Encoder using SciBert

Then, we aimed to focus on the Text Encoder. Our first thoughts were about the specifity of the NLP queries we were dealing with. *DistillBert*, while really performant and faster than *Bert* model, is trained on a general-domain corpus. We, thus wanted to try a pretrained model that is more specific to scientific texts that is *Scibert* introduced by Beltagy et al. [2019].While the core architecture is similar to BERT, SciBERT is pretrained on scientific texts only. We used the uncased prerained model that lowercase all characters when tokenizing. We trained this text encoder with the GATConv graph encoder. It appeared that while having a satisfying validation loss (around 0.2), the predictions were completely bad with a score of 0.0026. We think it might be due to an error in the code.

5 Hard Negative Mining

We noticed a large discrepancy between performances on the training set and the test set implying that the model might overfit the training set. We assumed that it was due to examples hard to classify because they were similar to other examples and were therefore misclassified in the sense that the graph embeddings were associated with the wrong text embeddings. To circumvent that, we thought it would be useful for the model during its contrastive task on the training set to have confusing pairs in the same batch in order to be able to distinguish between similar molecules.

This motivation was confirmed by the results we got when performing a kmeans clustering on the text embeddings from an already trained model on

50 epochs. We indeed saw that some clusters were huge with lots of similar embeddings and others were very small - even sometimes reduced to one example. This is very interesting because we want to make sure that the model is trained on those clusters with similar embeddings in the same batch to make sure the model is able to differentiate them. The problem is that with 26408 training examples the probability for two similar embeddings to appear in the same batch is very low. This is why we decided to use hard negative mining.

Hard negative mining simply consists here in making sure that similar text embeddings appear in the same batch in order to make the contrastive task harder. See Figure 2 to visualize the whole pipeline.

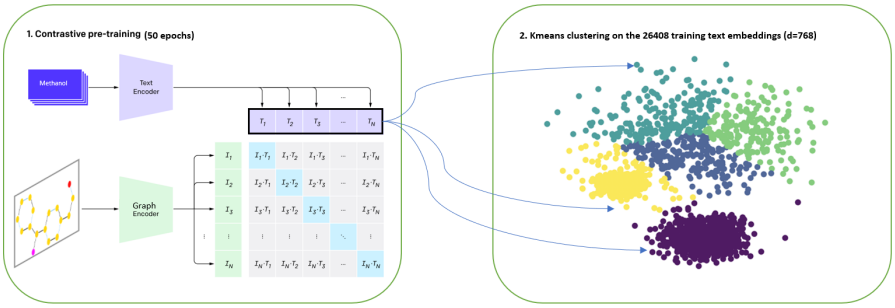

Figure 2: The trained text encoder is used to predict 26408 text embeddings on the training set which are then clustered in clusters of size greater than 32. The class labels are then used to make batches of the same class for further training.

After the clustering of the text embeddings we have a list of class labels. We use our labels list to make a sampler object which returns 32 random indices of samples from the same cluster. Finally, we can pass this sampler object to the training dataloader of PyTorch to make sure the training batch are made up of 32 samples with similar text embeddings and therefore similar descriptions. This makes the contrastive task harder and helps the model to perform better subsequently on the test set. Our best model trained this way went from 0.55 to 0.65 on the test set with a significant gain in terms of LRAP metric.

We also found empirically that alternating between epochs with and without hard negative mining was further improving the performances, which was a good method since each epoch with hard mining took about 45 minutes.

6 Ensemble models

Upon closer examination of our prediction matrices, we observed a notable discrepancy among models with similar scores on the test set, they exhibited divergent rankings. This divergence in ranking implies that these models learned distinct patterns and features. In line with the insights presented in the paper

by Edwards et al. [2021] introducing the Text2Mol model, it was observed that the correct molecule often appeared among the highly ranked graphs. However, the molecules ranked above the correct one varied between models.

Motivated by this observation, our investigation focused on ensemble methods aimed at mitigating the disparities in ranking among models. We wanted to investivate on ensemble methods that would erase this difference in ranking while keeping the correct molecule at a high rank.

The ensemble method that we used, gets back the final embeddings of the GAT model trained on 50 epochs and on the GAT model with negative mining. It then computes cosine similarity matrices for each model. The final matrix takes the mean over cosine similarities for each pairs.

7 Perspectives

7.1 Data augmentation

Data augmentation is a technique commonly employed in machine learning to artificially increase the diversity of the training dataset, enhancing the model's robustness and generalization ability. If we had time to implement it, for text we would have employed techniques such as synonym replacement, random word insertion/deletion or word embedding-based approaches to diversify textual queries. The combined augmentation strategy aims to expose the model to a wide range of variations, preventing overfitting and facilitating better performance on unseen data which would lead to a better score on the test set.

7.2 Loss

When training a neural network, a loss function is essential as it quantifies the difference between the predicted output and the expected outpout, providing a measure of how well the model is performing. The baseline for this project employs contrastive loss, a technique particularly suited for tasks like text-graph retrieval, where embeddings of different modalities need to be compared. Contrastive loss encourages the model to minimize the distance between similar pairs (e.g., matching text and graph representations) and maximize the distance between dissimilar pairs, facilitating the creation of a meaningful joint embedding space.

Nonetheless, with additional time, it would have been interesting to try other loss functions. For instance, triplet loss could be tested, offering a more nuanced approach by considering three samples in each training instance. The InfoNCE (Noise-Contrastive Estimation) Loss, known for its effectiveness in self-supervised learning, could also be considered to enhance the model's ability to distinguish relevant information. Finally, a Joint Loss, combining classification and regression losses, could provide a comprehensive understanding of both semantic and structural aspects of the data.

Each of these alternatives gives the opportunity for more nuanced embeddings for the integrated text and molecular graph representations.

7.3 Similarity

Measuring similarity between embeddings is crucial for evaluating the effectiveness of the model in capturing meaningful relationships between different data modalities. The baseline similarity measure used in this project is cosine similarity, which calculates the cosine of the angle between two vectors and is well-suited for capturing directional relationships.

However, with more time, exploring alternative similarity metrics could offer a more nuanced understanding of the embeddings. Euclidean Distance provides a straightforward measure of geometric closeness, capturing the overall magnitude of differences between embeddings. Manhattan Distance, being the sum of absolute differences, is robust to outliers and may offer insights into specific feature deviations. The Correlation Coefficient measures linear relationships between embeddings, providing information about their linear dependency. Minkowski Distance, a generalization of both Euclidean and Manhattan distances, allows for tuning the distance metric based on the dataset characteristics.

Those alternative similarity metrics brings a unique perspective to the evaluation process, considering different aspects of the embedding space.

References

I. Beltagy, K. Lo, and A. Cohan. Scibert: A pretrained language model for scientific text. *EMNLP*, 2019.

S. Brody, U. Alon, and E. Yahav. How attentive are graph attention networks? *ICLR*, 2022.

C. Edwards, C. Zhai, and H. Ji. Text2mol: Cross-modal molecule retrieval with natural language queries. *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 2021.

X. L. Hui Liu, Yibiao Huang and L. Deng. Attention-wise masked graph contrastive learning for predicting molecular property. *Briefings in Bioinformatics*, pages 1–9, 2022.