Molecule Retrieval with Natural Language Queries

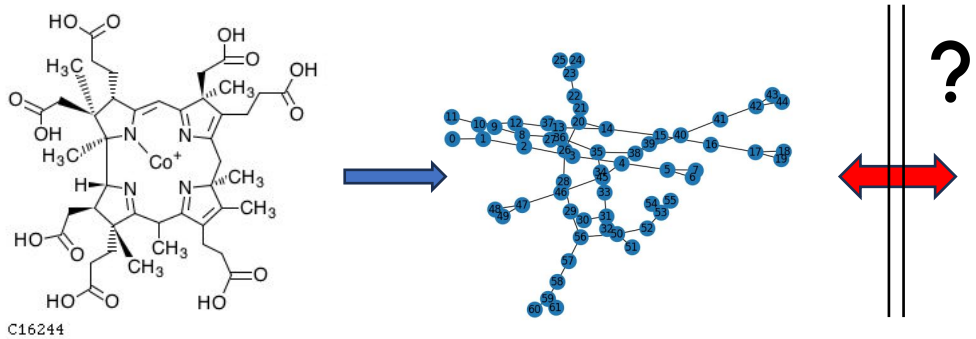
Challenge ALTeGraD

BARILLER Halvard
DENG Victor

January 2024

Introduction: Multimodality

Context: Multimodal representation of molecules



"Cobalt-precorrin-7 is a cobalt corrinoid that is precorrin-7 in which the four pyrrole-type nitrogens are bound to a central cobalt atom. It is a conjugate acid of a cobalt-precorrin-7(7-)."

Figure: *Graph representation of the Cobalt-precorrin-7*

Objective: Retrieve the molecule corresponding to a text query

Understanding the data

Dataset: Pairs of molecule structure (unweighted and undirected graph) and description in natural language

- 26,408 training samples
- 3,301 validation samples & 3,301 test samples

→ Each node has a feature vector in $\mathbb{R}^{300}$ extracted from Mol2Vec [JFT18]

→ Each description is tokenized and represented as a vector in $\mathbb{R}^{256}$

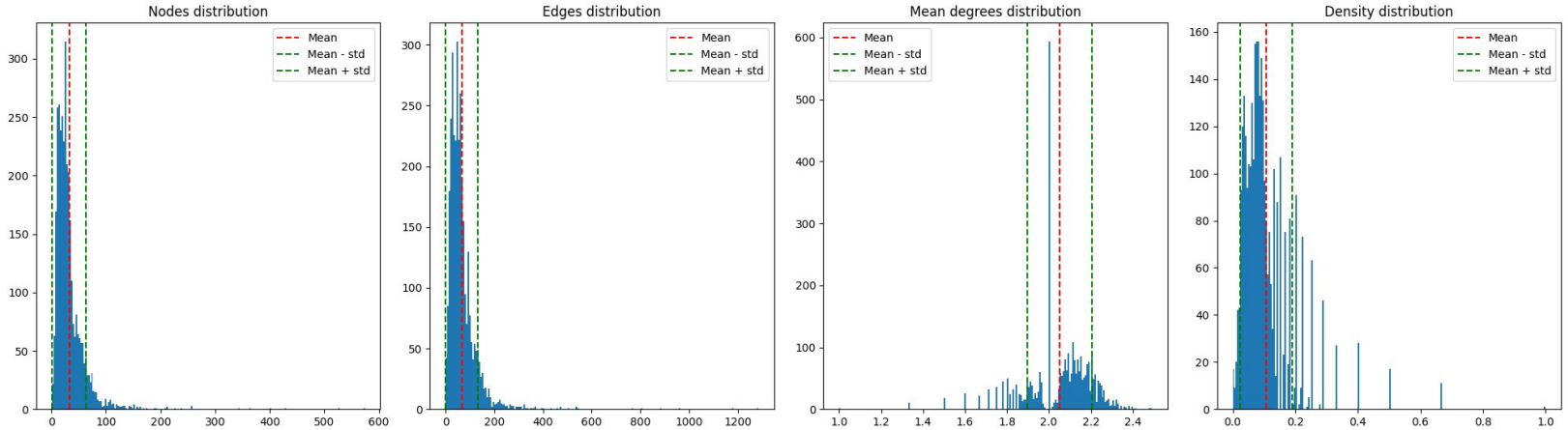


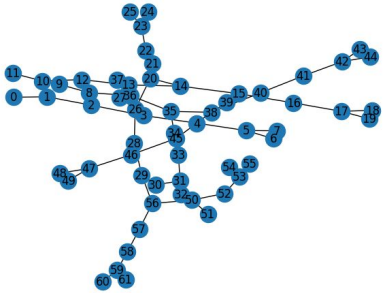
Figure: *Distributions of graphs' metrics on a subset of the training set*

“One Latent Space to Bind Them All”

[SOS] *Cobalt-precorrin-7 is a cobalt corrinoid that is precorrin-7 in which the four pyrrole-type nitrogens are bound to a central cobalt atom. It is a conjugate acid of a cobalt-precorrin-7(7-).* ***[EOS]***

Token Embedding

BERT-like Transformer



Graph Neural Network

MLP

Joint Embedding Space

Pull closer

Text Encoding

Encoding: Pretrained Transformer architecture BERT-like [DCLT18]

- DistilBERT [SDCW19]
- SciBERT [BLC19]
- **RoBERTa [LOG+19]**

Models	Trainable Parameters	Training Corpus	Vocabulary
DistilBERT	65M	16 GB BERT data (3.3B tokens)	30K tokens
SciBERT	110M	~ 16GB Scientific papers (3.17B tokens)	30K tokens
RoBERTa Base	125M	16 GB BERT data + 144 GB additional	52K tokens

Table: *Comparison of BERT variants*

Graph Convolutional Network [KW16]

Used in the baseline. Update rule:

$$\mathbf{X}' = \hat{\mathbf{D}}^{-1/2}\hat{\mathbf{A}}\hat{\mathbf{D}}^{-1/2}\mathbf{X}\mathbf{W}$$

where $\mathbf{X}$ and $\mathbf{X}'$ are the input and output feature matrices, $\hat{\mathbf{A}} = \mathbf{A} + \mathbf{I}$ is the adjacency matrix of the graph with inserted self-loops, $\hat{\mathbf{D}}$ is its diagonal degree matrix and $\mathbf{W}$ is a matrix of trainable parameters

ReLU activation between layers.

Originally: 3 convolution layers, 2-layer MLP after readout, hidden dimensions 300

Extended to 5 convolution layers with hidden dimension 300 and 3-layer MLP after readout with hidden dimension 600

Graph Isomorphism Network [XHLJ19]

Update rule:

$$h_v^{(k)} = \mathrm{MLP}^{(k)}\left(\left(1+\epsilon^{(k)}\right)h_v^{k-1} + \sum_{u\in\mathcal{N}(v)} h_u^{(k-1)}\right)$$

where $h_v^{(k)}$ is the hidden state of node v at layer k, $\mathcal{N}(v)$ denotes the neighbors of v, $\mathrm{MLP}^{(k)}$ is a MLP and $\epsilon^{(k)}$ is a scalar that can be fixed or trainable

Readout: concatenation of sum readouts of the input features and all the GIN convolution layers

Graph Attention Network [VCC+17]

Attention coefficients α_{ij} for the nodes $v_j \in \mathcal{N}(v_i)$:

$$\alpha_{ij}^{(t)} = \frac{\exp\left(\text{LeakyReLU}\left(a^T \left[W^{(t)} h_i^{(t)} || W^{(t)} h_j^{(t)}\right]\right)\right)}{\sum_{k \in \mathcal{N}_i} \exp\left(\text{LeakyReLU}\left(a^T \left[W^{(t)} h_i^{(t)} || W^{(t)} h_k^{(t)}\right]\right)\right)}$$

(where $[\cdot||\cdot]$ denotes concatenation and a is a trainable vector)

Representation $h_i^{(t+1)}$ of a node i at time $t+1$:

$$h_i^{(t+1)} = \sigma\left(\sum_{j \in \mathcal{N}_i} \alpha_{ij}^{(t)} W^{(t)} h_j^{(t)}\right)$$

Graphormer [YCL+21]

- All nodes attend to all other nodes
- Centrality encoding:

$$h_i^{(0)} = x_i + z^-_{\deg^-(v_i)} + z^+_{\deg^+(v_i)}$$

- Attention weights:

$$A_{ij} = \frac{(h_i W_Q)(h_j W_K)^T}{\sqrt{d}} + b_{\phi(v_i, v_j)}$$

where we took $\phi(v_i, v_j)$ to be the shortest path distance between v_i and v_j, as in [YCL+21]

- A virtual node that is connected to all the other nodes of the graph and is used for readout

GraphSAGE [HYL17]

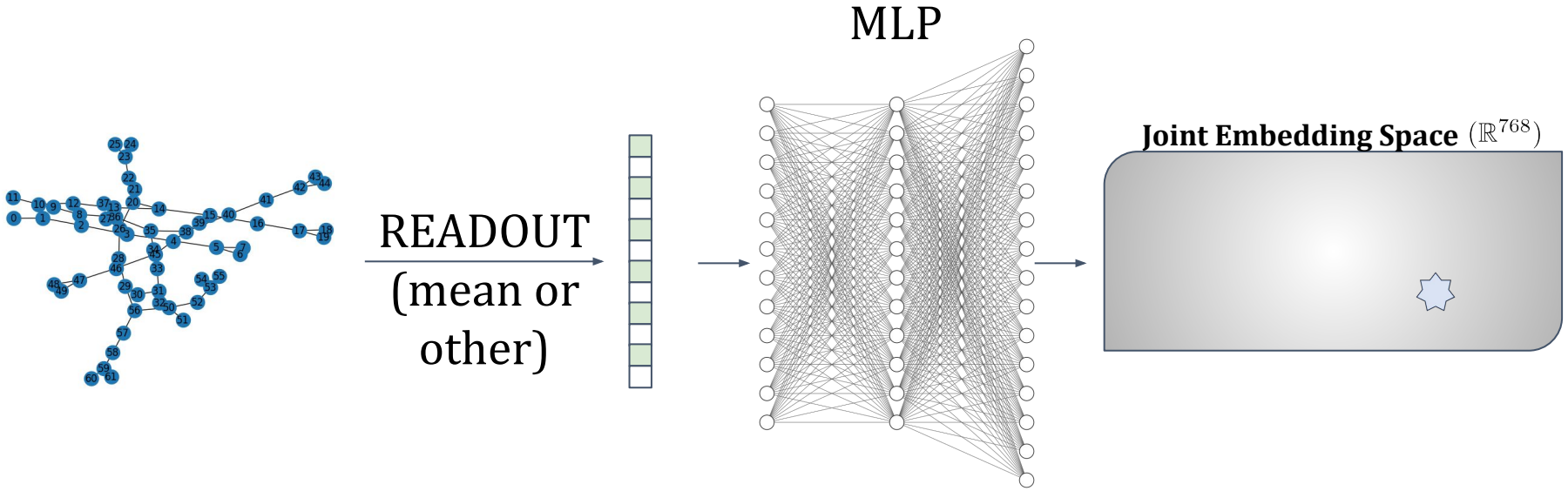
Let $\mathcal{N}^k(v)$ be a uniformly sampled subset of size *k* from the set of neighbors $\mathcal{N}(v)$ of node v .

Representation $h_v^{(t+1)}$ of a node i at time $t+1$:

$$h_v^{(t+1)} = \sigma\left(W^{(t)} \frac{h_v^{(t)} + \sum_{u \in \mathcal{N}^k(v)} h_u^{(t)}}{\deg(v)+1}\right)$$

$$h_v^{(t+1)} = \frac{h_v^{(t+1)}}{||h_v^{(t+1)}||_2}$$

Graph-level representation



Modality Alignment

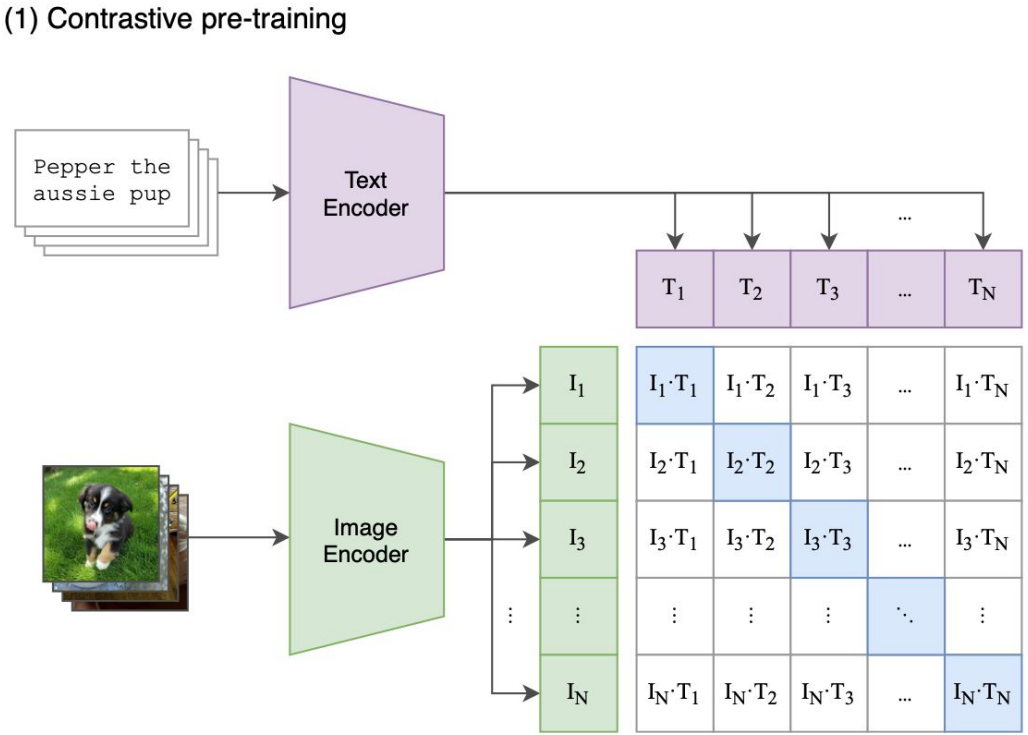


Figure: *From paper "Learning Transferable Visual Models From Natural Language Supervision" (CLIP network) [RKH+21]*

Losses

- Contrastive loss from the baseline:

$$\mathcal{L}((t_i)_i, (m_j)_j) = \frac{1}{N}\sum_{i=1}^{N} \mathrm{CE}((t_i^T m_j)_j, i) + \frac{1}{N}\sum_{j=1}^{N} \mathrm{CE}((m_j^T t_i)_i, j)$$

  where CE is the cross-entropy

- To ensure that $\cos((t_i)_i, (m_j)_j)$ decreases for $i \neq j$ when the loss is optimized, augmented loss:

$$\mathcal{L}_{aug} = \mathcal{L} + \mathcal{L}_{cos}$$

$$\text{where } \mathcal{L}_{cos}((t_i)_i, (m_j)_j) = \mathcal{L}\left(\left(\frac{t_i}{||t_i||_2}\right)_i, \left(\frac{m_j}{||m_j||_2}\right)_j\right)$$

Similarity

Different similarities:

- Cosine similarity: $\cos((t, m)) = \frac{\langle t, m \rangle}{||t||_2 ||m||_2}$
- Dot product similarity: $\text{dot}(t, m) = \langle t, m \rangle$
- Adjusted cosine similarity: given the mean text embedding $\bar{t}$ and the mean molecule embedding $\overline{m}$,

$$\text{adjcos}(t, m) = \cos((t - \bar{t}, m - \overline{m}))$$

Similarity

- A normalized average similarity, which is the average of the normalized cosine, adjusted cosine and dot product similarities, where we define the normalized similarity score from a similarity score sim by: given t and m and molecule embeddings $m_1, ..., m_N$,

$$\overline{\mathrm{sim}}(t, m) = \frac{\mathrm{sim}(t, m)}{\max_{1 \leq i \leq N} \mathrm{sim}(t, m_i)}.$$

Implementation Technicalities

- Saturate the batch size, to increase the number of negative pairs per step [CKNH20, RKH+21]. (*Used batch size of 32*).
- Uncouple the learning rates of the pretrained text encoder and the graph encoder trained from scratch
- Linear learning rate scheduler
- Optimizations to reduce training time (AMP) [MNA+17]

Ensemble Methods: soft ranking

Normalized average similarities, model 1

	m_1	m_2	m_3
t_1	0.48	-0.11	0.98
t_2	0.22	-0.75	-0.11
t_3	0.34	-0.42	0.63

+

Normalized average similarities, model 2

	m_1	m_2	m_3
t_1	0.76	0.42	-0.13
t_2	-0.35	0.17	-0.85
t_3	0.30	0.55	0.83

+

Normalized average similarities, model 3

	m_1	m_2	m_3
t_1	-0.96	-0.09	-0.25
t_2	0.57	0.46	-0.23
t_3	0.01	-0.32	0.45

	m_1	m_2	m_3
t_1	0.28	0.22	0.60
t_2	0.44	-0.12	-1.19
t_3	0.65	-0.19	1.91

Ensemble Methods: hard ranking

Model 1

	m_1	m_2	m_3
t_1	0.48	-0.11	0.98
t_2	0.22	-0.75	-0.11
t_3	0.34	-0.42	0.63

Rank →

	m_1	m_2	m_3
t_1	1	0	2
t_2	2	0	1
t_3	1	0	2

+

Model 2

	m_1	m_2	m_3
t_1	0.76	0.42	-0.13
t_2	-0.35	0.17	-0.85
t_3	0.30	0.55	0.83

Rank →

	m_1	m_2	m_3
t_1	2	1	0
t_2	1	2	0
t_3	0	1	2

→

	m_1	m_2	m_3
t_1	3	1	2
t_2	3	2	1
t_3	1	1	4

Numerical Results

Models	Number of Epochs						Best Performance
	1	30	60	90	140	190	
GCN - Baseline	24.85	68.44	-	-	-	-	68.44
GCN - Extended baseline	12.75	66.15	72.52				72.52
GIN	**31.85**	74.29	79.52	81.51	84.23	-	84.23
Graphormer	01.12	55.48	65.91	76.25	78.05	82.57	82.57
GraphSAGE	24.35	**76.74**	**79.71**	**81.59**	**85.14**	-	**85.14**
GAT	24.96	71.69	76.54	79.08	79.55	80.31	80.31

Table 1. LRAP on Validation Set

Models	Public Score		Private Score	
	Soft Ranking	Hard Ranking	Soft Ranking	Hard Ranking
GIN + Graphormer + SAGE	**88.70**	88.49	89.56	**89.74**
GIN + Graphormer + SAGE + GAT	87.21	86.66	88.07	87.41

Table 2. LRAP on Validation Set for Ensemble Methods

Improvement Ideas

- Data augmentation
  - Text replacement with Contextual Word Embedding
  - Chemically-sound graph augmentation [MWL+17]
- Optimizers' hyperparameters tuning
- Dimension of the embedding space

Conclusion

- Tested a variety of graph and text encoders
- Ensemble methods helped achieve rather good performance
- Hands-on experience with model training and GPU programming

References

[BLC19] Beltagy, I., Lo, K., & Cohan, A. (2019). SciBERT: A pretrained language model for scientific text. arXiv preprint arXiv:1903.10676.

[CKNH20] Chen, T., Kornblith, S., Norouzi, M., & Hinton, G. (2020, November). A simple framework for contrastive learning of visual representations. In *International conference on machine learning* (pp. 1597-1607). PMLR.

[DCLT18] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.

[HYL17] Hamilton, W., Ying, Z., & Leskovec, J. (2017). Inductive representation learning on large graphs. *Advances in neural information processing systems, 30*.

[JFT18] Jaeger, S., Fulle, S., & Turk, S. (2018). Mol2vec: unsupervised machine learning approach with chemical intuition. *Journal of chemical information and modeling, 58(1)*, 27-35.

[KW16] Kipf, T. N., & Welling, M. (2016). Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907.

References

[LOG+19] Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., ... & Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach. arXiv preprint arXiv:1907.11692.

[MNA+17] Micikevicius, P., Narang, S., Alben, J., Diamos, G., Elsen, E., Garcia, D., ... & Wu, H. (2017). Mixed precision training. arXiv preprint arXiv:1710.03740.

[MWL+17] Magar, R., Wang, Y., Lorsung, C., Liang, C., Ramasubramanian, H., Li, P., & Farimani, A. B. (2022). AugLiChem: data augmentation library of chemical structures for machine learning. *Machine Learning: Science and Technology, 3(4)*, 045015.

[RKH+21] Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., ... & Sutskever, I. (2021, July). Learning transferable visual models from natural language supervision. In *International conference on machine learning* (pp. 8748-8763). PMLR.

[SDCW19] Sanh, V., Debut, L., Chaumond, J., & Wolf, T. (2019). DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. arXiv preprint arXiv:1910.01108.

References

[VCC+17] Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., & Bengio, Y. (2017). Graph attention networks. arXiv preprint arXiv:1710.10903.

[XHLJ19] Xu, K., Hu, W., Leskovec, J., & Jegelka, S. (2018). How powerful are graph neural networks?. arXiv preprint arXiv:1810.00826.

[YCL+21] Ying, C., Cai, T., Luo, S., Zheng, S., Ke, G., He, D., ... & Liu, T. Y. (2021). Do transformers really perform badly for graph representation?. Advances in Neural Information Processing Systems, 34, 28877-28888.