Could Chemical Language Models benefit from Message Passing

Jiaqing Xie, Ziheng Chi
Department of Computer Science
ETH Zurich
{jiaxie, zihchi}@student.ethz.ch

Abstract

Pretrained language models (LMs) showcase significant capabilities in processing molecular text, while concurrently, message passing neural networks (MPNNs) demonstrate resilience and versatility in the domain of molecular science. Despite these advancements, we find there are limited studies investigating the relationship between molecular structures and their corresponding textual representations. Therefore, in this paper, we propose two strategies to evaluate whether an information integration can enhance the performance: contrast learning, which involves utilizing an MPNN to supervise the training of the LM, and fusion, which exploits information from both models. Our empirical analysis reveals that the integration approaches exhibit superior performance compared to baselines when applied to smaller molecular graphs, while these integration approaches do not yield performance enhancements on large scale graphs. Furthermore, we conduct experiments to assess the impact of dataset splitting strategies and random seeds on the overall performance.

1 Introduction

The success of attention mechanisms on sequential data has introduced a massive family of large language models based on Transformer architecture (Vaswani et al., 2017). It is evident that these large language models are useful for encoding sequential objects such as text (Liu et al., 2019), molecules (Honda et al., 2019), speech (Huang et al., 2021), and forecasting data (Giuliari et al., 2021). It has been demonstrated that pretrained molecule language models are capable of encoding chemical elements semantically without learning structures (Honda et al., 2019; Xia et al., 2022; Chithrananda et al., 2020; Wang et al., 2019a). Especially for proteins which function as natural components of the human body and a representative of molecule family, they could be efficiently encoded by transformer (Rao et al., 2019; Elnaggar et al., 2021; Rives et al., 2021; He et al., 2021) which acts as masked language modelers.

In contrast to text, molecules contain inherent relationships between their elements, indicating that structural encoding is necessary in addition to word embeddings. Message passing neural network (MPNN), emerging as a prominent method for encoding structural information in recent years, has demonstrated its robustness and versatility within the field of molecular sciences. By leveraging the 2-dimensional topological and 3-dimensional geometrical information as augmented features (Liu et al., 2021; Stärk et al., 2022), it is possible to learn molecular embeddings from structures without sequentially encoding traditional SMILES expressions.

The advent of MPNNs has promoted the exploration of graph-based learning methods for molecular science. Graph contrastive learning captures potential different structural distributions to fine-tune self-learned representations, where both local and global features are enhanced with chemical domain expertise (Stärk et al., 2022; You et al., 2021; Wang et al., 2022). Besides, the success of GPT (Radford et al., 2018) in traditional natural language processing tasks also motivates the research on graph transformers and graph GPT tailored for the molecule domain (Hu et al., 2020b; Bagal et al., 2021; Rong et al., 2020; Ying et al., 2021; Zhu et al., 2022). Few studies has been investigated to appropriately merge text embeddings and graph embeddings for learning molecule representation better. There has been one study which demonstrated such relationship but with additional prompting with GPT model (Chen et al., 2024), which is out of our scope. In this paper, we aim to explore the interplay between molecular graph embeddings and SMILE token embeddings. We propose two categories of techniques for integrating information: contrast learning and fusion. In contrast learning-based methods, we incorporate

Proceedings of the 1st Workshop on Language + Molecules (L+M 2024), pages 11–21

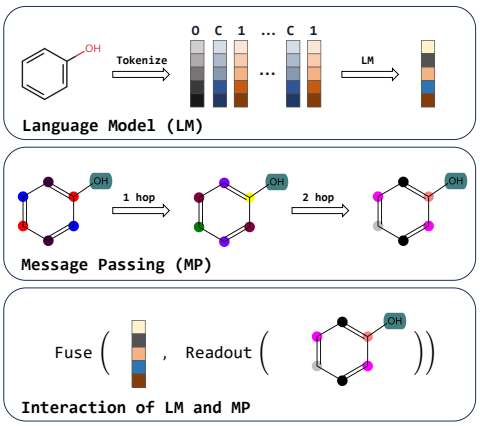

Figure 1: Baseline model: language model (LM) and message passing networks (MPNN). An interaction of LM and MPNN is investigated in this research.

an MPNN as an auxiliary model to supervise the training of the language model, operating at node or graph levels, while we only utilize the language model for downstream tasks. In fusion-based methods, we exploit information from both models to generate outputs for downstream tasks. This is achieved either by merging the output embeddings from both models or by integrating the output embeddings from one model with the input embeddings of the other.

Our main contributions are to as follows:

1. Explore various information integration approaches to assess the necessity of incorporating supplementary structural features in molecular LLMs research, instead of pursuing state-of-the-art performance.

2. Benchmark a series of combination of sequential-based methods (LM) and structural-based methods (MPNN) as baselines for further research.

2 Related Work

2.1 Molecule Representation Learning

The Simplified Molecular Input Line Entry System (SMILES) has become a cornerstone in cheminformatics, providing a compact and standardized representation for chemical structures. Conserving molecular structural information and atom orderings, the SMILES descriptor converts a molecule from its structural representation into a condensed 1-dimensional textual sequence. For example, a phenol molecule (C_6H_5OH) is represented as `C1=CC=C(C=C1)O`. Similar to the tokenization in natural language settings, a molecule is expressed as a sentence and atoms are expressed as words. This allows efficient utilization of large language models in chemical research.

2.2 Pretrained Large Language Models

The advent of the transformer architecture (Vaswani et al., 2023) represents a breakthrough in the field of natural language processing. Over the past few years, many excellent pretraining strategies have been proposed, such as BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019), significantly improving the capabilities of the large language models. As SMILES allows converting molecular structures into textual sentences, it is possible to apply language models for molecular machine learning, which facilitates the research on pretraining molecular language models.

Based on the implementation of RoBERTa, ChemBERTa (Chithrananda et al., 2020) employs chemistry oriented masked-language modelling as its pretraining strategy, while the improved version ChemBERTa-2 (Ahmad et al., 2022) adopts multi-task regression as another pretraining task and uses larger training datasets. There are also other BERT-like transformer models, such as MolBERT (Fabian et al., 2020) and SMILES-BERT (Wang et al., 2019b), which are pretrained with different objectives on different molecule datasets.

2.3 Contrastive Learning

Contrastive learning has emerged as a powerful paradigm in self-supervised learning. Unlike traditional methods that rely solely on labeled data, this approach leverages the differences between data to learn representations. Based on the assumption that similar instances should be closer in the embedding space, the objective is to maximize the similarity between positive data pairs while minimizing the similarity between negative data pairs. So far, contrastive learning has demonstrated efficacy across diverse domains. A common practice of this approach is based on data augmentation (You et al., 2021), where the utilization of unlabeled data enhances model generalizability and robustness. Furthermore, this approach is also widely adopted in the field of multimodality (Radford et al., 2021), where the availability of different data forms allows leveraging one representation to supervise the other.

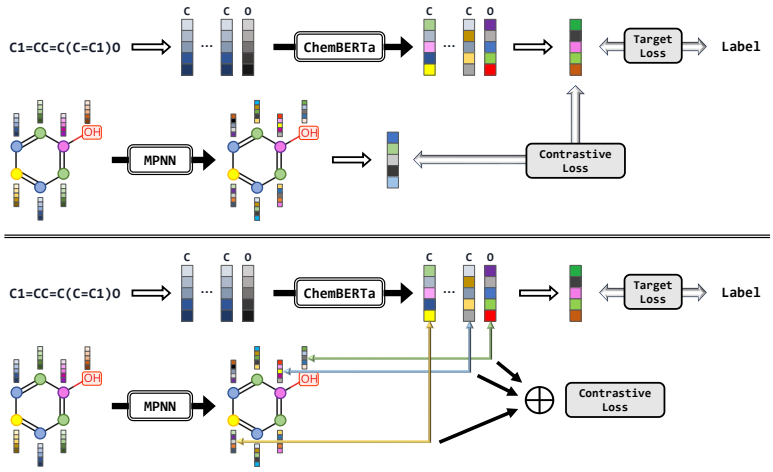

Figure 2: Contrastive Learning. Above: Node level contrastive learning. Below: Graph level contrastive learning.

3 Methods

We investigate two kinds of information merging methods: contrast learning-based methods and fusion-based methods. In contrast learning-based methods, we use GNN as an auxiliary model to supervise the training of the language model, while we only use the language model for downstream tasks. In fusion-based methods, we make use of information from both models to generate the output for downstream tasks. For each kind of method, we consider different model architectures. In this section, we will first briefly review the two baseline models ChemBERTa (Chithrananda et al., 2020) and NNConv (Gilmer et al., 2017), and then describe the contrast learning-based methods and the fusion-based methods.

3.1 Baseline (Fig. 1)

Language Models. We choose ChemBERTa (Chithrananda et al., 2020) as our baseline model. The architecture of ChemBERTa is similar to BERT (Devlin et al., 2019), consisting of an embedding layer and several encoder layers. A molecule is first converted into the textual format through SMILES, and then a SMILES tokenizer is applied to convert the words into input tokens. After embedding lookup, each token is assigned with an embedding. Then the encoder layers which consist of a multi-head self-attention layer and a feed-forward layer transform the input token embeddings to hidden state representations. Finally, the task-specific output layer (classifier or regressor) predicts the result.

We ask the ChemBERTa model to produce three-level information for a molecule. First, node embeddings are extracted from the final hidden state representations. Since the SMILES transformation preserves atom orderings, each atom in the original molecule corresponds to a specific output token embedding. Second, the graph embedding is extracted from the special token at the beginning of the sequence. Third, the property prediction result is the final output.

The entire process can be depicted as follows:

$$
\begin{aligned}
\texttt{Tokens} &= \texttt{Tokenizer}(\texttt{Sequence}) \\
E_{\text{in}} &= \texttt{Embedding}(\texttt{Tokens}) \\
E_{\text{out}} &= \texttt{Encoder}(E_{\text{in}}) \\
N &= E_{\text{out}}[\texttt{Node\_Indices}] \\
G &= E_{\text{out}}[0] \\
P &= \texttt{Predictor}(G)
\end{aligned}
\tag{1}
$$

where E_{in} and E_{out} represent the input token embeddings and final hidden state representations; N, G, and P represent the node embeddings, graph embedding, and property prediction result.

Message Passing Neural Networks. There are different types of graph neural networks, which include graph convolution, graph attention and neural message passing networks (MPNN). Edge attributes or edge features are important in message passing mechanisms (Johannes et al., 2020; Gilmer et al., 2017). For the baseline model, we follow the model setting in the first paper of MPNN for Quantum Chemistry dataset QM9 (Gilmer et al., 2017), which iteratively updates the message m_v^t

and the hidden state h_v^t for each node v:

$$m_v^{(t+1)} = \sum_{u \in \mathcal{N}(v)} \texttt{Aggr}[h_v^{(t)}, h_u^{(t)}, e_{uv}] \quad (2)$$

$$h_v^{(t+1)} = \mathcal{U}(h_v^{(t)}, m_v^{(t+1)}) \quad (3)$$

$\texttt{Aggr}[\cdot]$ is the function that aggregates neighbor node u's information as well as the attributes e_{uv} of the shared edge with u. $\mathcal{U}(\cdot, \cdot)$ updates hidden states for v.

3.2 Integration 1: Contrastive Learning (Fig. 2)

Node Level Contrastive Learning In order to compute the constrastive loss, we need a triple $\langle$**anchor, positive, negative**$\rangle$. Anchor and positive node embeddings are sampled from language models and message passing networks respectively. Negative samples are randomly generated from graphs with a different permutation. For an example node triple $t = \langle a, p, n \rangle$, its corresponding contrastive learning loss triplet loss is given by:

$$L(t) = \max\{d(a, p) - d(a, n) + \text{margin}, 0\} \quad (4)$$

where margin is 1.0 and distance measurement $d(i, j)$ is defined as L_p-norm : $d(i, j) = \|i - j\|_p$. p is often set to 2 as an Euclidean distance metric. In a $\mathcal{M}$ mini-batch of training graphs (number of $\mathcal{N}$ nodes) with $\mathcal{K}$ triples, the triplet loss is given by:

$$\mathrm{L} = \sum_{m=1}^{\mathcal{M}} \sum_{i=0}^{\lceil \frac{\mathcal{N}}{\mathcal{K}} \rceil - 1} \sum_{j=1}^{\mathcal{K}} \max\{d(a_k^m, p_k^m) - d(a_k^m, n_k^m) + \text{margin}, 0\} \quad (5)$$

where $k = |\mathcal{K}|i + j$. Consider a binary classification problem, as mentioned we need to perform `Readout` function to obtain the global information of a graph. If it is an average function, the total loss is given by a prediction loss such as negative log likelihood (NLL) loss, and a regularized triplet contrastive loss which has been defined above:

$$\mathrm{L} = \sum_{m=1}^{\mathcal{M}} \text{NLL}\left(\text{MLP}\left(\frac{1}{\mathcal{N}} \sum_{i=0}^{\lceil \frac{\mathcal{N}}{\mathcal{K}} \rceil - 1} \sum_{j=1}^{\mathcal{K}} a_k^m\right), y^m\right) + \alpha \cdot \sum_{m=1}^{\mathcal{M}} \sum_{i=0}^{\lceil \frac{\mathcal{N}}{\mathcal{K}} \rceil - 1} \sum_{j=1}^{\mathcal{K}} \max\{d(a_k^m, p_k^m) - d(a_k^m, n_k^m) + \text{margin}, 0\} \quad (6)$$

where $k = |\mathcal{K}|i + j$ likewise and α is a regularization term.

Graph Level Contrastive Learning. Apart from establishing negative samples between each pair of nodes at a fine grained level, we estimate contrastive learning at a coarse grained level which aims at computing the difference between language model graph embeddings and MPNN graph embeddings. This could potentially avoid the situation that individual nodes with large difference contribute more to the difference of the molecule property. Moreover, the complexity is pretty lower than the complexity of node level comparison, which will be discussed in part 3.4. Similar to the node level training loss in (6), the graph level training loss is defined as:

$$\mathrm{L} = \sum_{m=1}^{\mathcal{M}} \text{NLL}\left(\text{MLP}\left(a^m\right), y^m\right) + \alpha' \cdot \sum_{m=1}^{\mathcal{M}} \max\{d(a^m, p^m) - d(a^m, n^m) + \text{margin}, 0\} \quad (7)$$

where α' is a regularization term. Note that different molecules could have a similar graph embedding which could lead to a similar quantum property, for example isomers. And also note that we use message passing network outputs to self-supervise (or fine-tune) language model outputs either in node level and graph level settings. It means that we do not directly use MPNN outputs to perform predictions. This is because we want to see if injecting geometry information of molecules is beneficial to the end-to-end training of **language models**. It is different from the collaborative training (fusion) which would be introduced in the following parts.

3.3 Integration 2: Fusion (Fig. 3)

Late Fusion Different from self-supervised learning settings in part 3.2, we introduce another important interaction between LM embeddings and MPNN embeddings: late fusion (Sachan et al., 2021). It is called late fusion since the interaction happens after their corresponding embeddings h_{LM} and h_{MPNN} are extracted. The interaction is given by the notation $\bigoplus$, and the prediction is then given as:

$$y_{\text{pred}} = \text{MLP}\left(h_{\text{LM}} \bigoplus h_{\text{MPNN}}\right) \quad (8)$$

$$\bigoplus = \{+, \max, \|, \odot\} \quad (9)$$

$$L = \sum_{i=1}^{\mathcal{M}} \text{NLL}(y_{\text{pred}}^i, y^i) \quad (10)$$

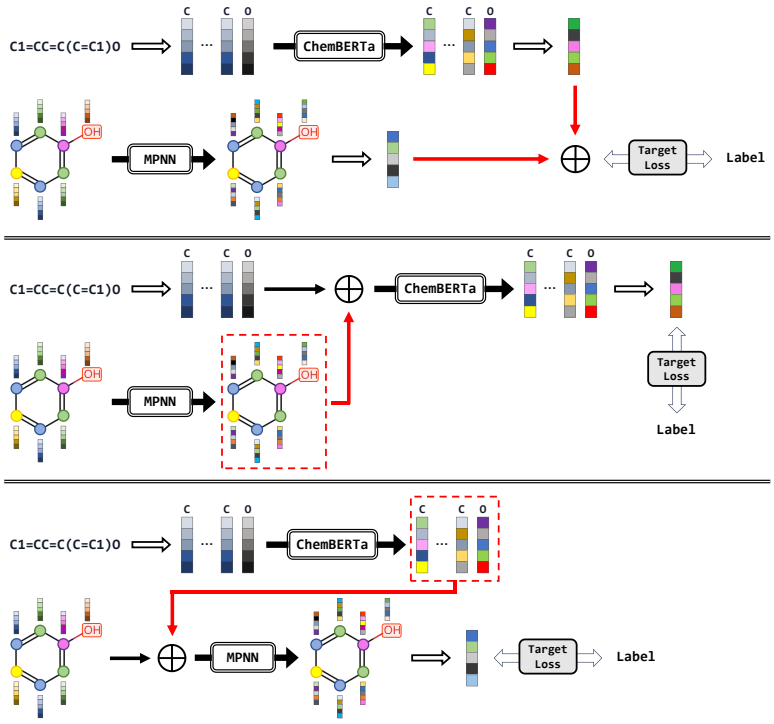

Figure 3: Fusion. Top: Late Fusion. Middle: MPNN2LM Joint Fusion. Bottom: LM2MPNN Joint Fusion.

where the interaction contains element-wise addition, maximization, concatenation and gate function which is used for building highway layers.

Joint Fusion. Based on the late fusion, we consider two situations: 1) MPNN2LM: fuse initial graph embeddings and LM outputs $\Rightarrow$ perform MPNN downstream tasks, and 2) LM2MPNN: fuse initial token embeddings and MPNN outputs $\Rightarrow$ finetune LM downsteam tasks.

MPNN2LM We initialized the word embedding for language model: $h_{\text{LM}}^{(0)}$. MPNN embedding is given by h_{MPNN}. Then the fused embedding is $h_{\text{LM}}^{(0)} \bigoplus h_{\text{MPNN}}$, which would be the input embedding for the pretrained language model. The node mask is also considered since in part 3.1 we mentioned that paddings are added to ensure the same length of input. After fine-tuning pretrained **LM**, we readout the global information h' to perform downstream tasks. Overall, the graph embedding prepared for MLP is:

$$h' = \texttt{Readout}\left(\textbf{LM}\left(h_{\text{LM}}^{(0)} \bigoplus h_{\text{MPNN}}, \textbf{mask}\right)\right) \tag{11}$$

LM2MPNN Revisiting how MPNN works by (1) and (2), the node embedding for v are fused with the LM output for mask index v: $h_v^{(t)} \bigoplus h_{\text{LMM}(v)}$. Similar to the neighbor node u, they are fused with the LM output for mask index u: $h_u^{(t)} \bigoplus h_{\text{LMM}(u)}$. Then the message is aggregated by:

$$m_v^{(t+1)} = \sum_{u \in \mathcal{N}(v)} \texttt{Aggr}\left(h_v^{(t)} \bigoplus h_{\text{LMM}(v)}, h_u^{(t)} \bigoplus h_{\text{LMM}(u)}, e_{uv}\right) \tag{12}$$

For the update function $\mathcal{U}(\cdot,\cdot)$, the new update rule:

$$h_v^{(t+1)} = \mathcal{U}\left(h_v^{(t)} \bigoplus h_{\text{LMM}(v)}, m_v^{(t+1)}\right) \tag{13}$$

3.4 Complexity Analysis

Baseline Models. The time complexity for baseline models are mainly dominated by their corresponding model architecture. Assume the input size $\in \mathbb{R}^{N \times d}$ For the pretrained transformer model, the self-attention module is the bottleneck, which is bounded by $O(N^2 \cdot d)$. Assume that there are L self-attention layers, then it will increase to $O(N^2 \cdot d \cdot L)$. The complexity for feed forward layers is $O(N \cdot d^2 \cdot L)$. The overall complexity for baseline LM is then $O(N^2 \cdot d \cdot L + N \cdot d^2 \cdot L)$. For MPNN, computing each message has a complexity of $O(d)$. The total complexity for the message passing step is then $O(E \cdot d)$. Updating nodes will be $O(N \cdot d^2)$. L layers lead to $O(L \cdot E \cdot d + L \cdot N \cdot d^2)$.

So it depends on whether the graph is sparse or not. If graph is sparse, $N^2 \gg E$, then the complexity of LM is greater than the complexity of MPNN. This situation often occurs in real world applications.

Contrastive Learning. For contrastive learning, apart from the basic complexity for LM and MPNN, it also includes the complexity for computing contrastive loss. Take triplet loss as an example, in equation (5), the complexity is dominated by the last term. Assume that computing max only requires O(1). Computing $d(\cdot, \cdot)$ requires $O(d)$ if the dimension of inputs is d. Then the overall complexity is $O(d \cdot N)$ where N is the number of nodes in the graph. A node level contrastive learning then requires $O(N^2 \cdot d \cdot L + 2 \cdot N \cdot d^2 \cdot L + d \cdot N + L \cdot d \cdot E)$. For graph level contrastive learning we find that only complexity of model terms dominates, which leads to $O(N^2 \cdot d \cdot L + 2 \cdot N \cdot d^2 \cdot L + L \cdot d \cdot E + d)$, which is faster than that of node level.

Fusion We simply investigate $\bigoplus$ by choosing $\max$ which requires $O(1)$. The element wise maximization requires $O(N \cdot d)$ since input size is $N \times d$. Then the time complexity of late fusion would be the same as the complexity of nodel level contrastive learning, which is $O(N^2 \cdot d \cdot L + 2 \cdot N \cdot d^2 \cdot L + L \cdot d \cdot (N + E))$. `MPNN2LM` has the same complexity while `LM2MPNN` is much more complex since $\bigoplus$ directly affects the complexity of message passing operation. We already know that the complexity of MPNN is $O(E \cdot d)$. We assume that the average number of nodes is $\frac{2E}{N}$. Then the additional element wise addition contributions additional $O(E \cdot d)$, which leads to the overall complexity for `LM2MPNN`: $O(N^2 \cdot d \cdot L + 2 \cdot N \cdot d^2 \cdot L + L \cdot d \cdot (N + 2E))$.

4 Experiment settings

4.1 Dataset

We follow the following paradigm (Luo et al., 2022) for prediction on quantum chemistry based datasets: first we perform tests on small scale and classifical benchmark molecule datasets. In our future works, we want to test its robustness on large scale and recently proposed benchmarks such as PCQM4Mv2 (Hu et al., 2020a). For small datasets we choose from MoleculeNet dataset (Wu et al., 2018) which collects data from physical chemistry, biophysics and physiology field. It has provided plenty of molecule datasets to play with (Wu et al., 2018). For large datasets, we choose **QM9** (Gilmer et al., 2017) as tested in MPNN. The task is to predict property for each molecule using models in part 3. Selected datasets are **HIV**, **BACE**, **ESOL** and **BBBP** (Wu et al., 2018). A simple description of chosen dataset and task type is listed in table 3. **HIV**, **BBBP**, and **BACE** are used for binary classification settings, while **ESOL** and **QM9** are used for regression settings. For simplicity, we only choose the first target from all 19 classes, which is the Dipole moment μ. For the regression problem, the performance is measured by mean absolute error (mae). As for the classification, it is measured by the mean accuracy (acc). Specifically, the pretrained ChemBERTa is time-consuming on QM9 and HIV dataset.

4.2 Hyper-parameter settings

There are two pretrained model to choose from: ChemBERTa and its improved version ChemBERTa-2. We choose Adam optimizer for optimizing model parameters with default learning rate 0.001 when running with pretrained ChemBERTa-2 (<4G). The initial learning rate is tuned to 0.0002 when running with ChemBERTa since the model size is large (>16G) which requires a small learning rate. We follow a 8:1:1 train-valid-test ratio for MoleculeNet dataset, and follow an approximate 21:2:2 train-valid-test ratio for **QM9** dataset. Hidden dimension is set to 64. The default choice for $\bigoplus$ is sum (addition). Five fixed seeds are 0, 7, 42, 100, 2024 for result reproduction.

4.3 Scalability

A single NVIDIA A100 GPU could satisfy all our experiments. In other words, it is scalable for training all datasets including large scaled ones. The maximum usage is observed when running pretrained ChemBERTa on HIV dataset. For other datasets it's also possible to train on a GeForce RTX 3090 GPU.

5 Results

Observation 0: Protein language models are more preferred. A fundamental observation from experimenting on MoleculeNet is that purely using message passing neural networks are inferior to language models in molecule property prediction. This phenomenon is also mentioned in the previous research work (Xu et al., 2022). This has indicated some works to include the geometric properties such as 3D information and rotation invariant parameters in message passing

Model	HIV (**acc.**) ↑	BACE (**acc.**) ↑	BBBP (**acc.**) ↑	ESOL (**mae.**) ↓
ChemBERTa	0.9776 ± 0.0021	0.8280 ± 0.0319	0.9105 ± 0.0153	0.5529 ± 0.0332
MPNN	0.9774 ± 0.0022	0.8080 ± 0.0256	0.8737 ± 0.0140	0.6252 ± 0.0072
ChemBERTa contra. MPNN (node)	$\mathbf{0.9782 \pm 0.0035}$	0.8280 ± 0.0271	0.9118 ± 0.0245	0.5326 ± 0.0534
ChemBERTa contra. MPNN (graph)	0.9774 ± 0.0022	0.8300 ± 0.0352	0.9131 ± 0.0307	0.5404 ± 0.0495
ChemBERTa + MPNN (graph)	0.9778 ± 0.0020	0.8320 ± 0.0331	0.9065 ± 0.0147	0.5002 ± 0.0339
ChemBERTa ← MPNN	0.9773 ± 0.0022	0.8060 ± 0.0422	0.9053 ± 0.0099	$\mathbf{0.4819 \pm 0.0325}$
ChemBERTa → MPNN	0.9773 ± 0.0022	$\mathbf{0.8380 \pm 0.0366}$	$\mathbf{0.9184 \pm 0.0189}$	0.5561 ± 0.0461

Table 1: Performance of pretrained ChemBERTa on MoleculeNet datasets.

Model	HIV (**acc.**) ↑	BACE (**acc.**) ↑	BBBP (**acc.**) ↑	ESOL (**mae.**) ↓
ChemBERTa-2	0.9792 ± 0.0018	0.8560 ± 0.0206	0.9171 ± 0.0136	0.4738 ± 0.0330
MPNN	0.9774 ± 0.0022	0.8010 ± 0.0392	0.8737 ± 0.0140	0.6252 ± 0.0072
ChemBERTa-2 contra. MPNN (node)	0.9791 ± 0.0011	0.8620 ± 0.0256	$\mathbf{0.9290 \pm 0.0128}$	$\mathbf{0.4393 \pm 0.0338}$
ChemBERTa-2 contra. MPNN (graph)	$\mathbf{0.9800 \pm 0.0017}$	0.8540 ± 0.0258	0.9197 ± 0.0163	0.4643 ± 0.0354
ChemBERTa-2 + MPNN (graph)	0.9791 ± 0.0012	$\mathbf{0.8680 \pm 0.0293}$	0.9263 ± 0.0113	0.4493 ± 0.0328
ChemBERTa-2 ← MPNN	0.9772 ± 0.0016	0.8400 ± 0.0374	0.8974 ± 0.0098	0.5012 ± 0.0335
ChemBERTa-2 → MPNN	0.9789 ± 0.0012	0.8480 ± 0.0204	0.9224 ± 0.0141	0.4516 ± 0.0264

Table 2: Performance of improved pretrained ChemBERTa-2 on MoleculeNet datasets.

Name	**#graphs**	**#nodes**	**#features**	**#classes**
HIV	41,127	∼25.5	9	1
BBBP	2,050	∼23.9	9	1
BACE	1,513	∼34.1	9	1
ESOL	1,128	∼13.3	9	1
QM9	130,831	∼18.0	11	19

Table 3: Descriptions of selected datasets from MoleculeNet

networks to reinforce its prediction and expressive power. The explanation of this phenomenon would be that 1) model size of either ChemBERTa-1 or ChemBERTA-2 model is larger than the size of message passing networks and 2) either ChemBERTa-1 or ChemBERTA-2 model has been pretrained on some more larger datasets for example ZINC dataset, while message passing networks do not follow the pretraining scheme of large language models.

Observation 1: Integration on relatively small graphs are more preferred. Using the pretraind ChemBERTa-2, we found that both contrastive learning and fusion methods outperform baseline models in **ESOL**, **BACE**, and **BBBP** where they are relatively small compared with **QM9** and **HIV** datasets. Especially, node level contrastive learning performs the best and it seems to be robust among all tasks, followed by late fusion methods and joint fusion methods when injecting LLM to MPNNs. In large dataset, the tuning strategy might influence the potential performance, where it splits the dataset in a better way therefore we perform one ablation regarding train test split (in section 6) to avoid the difference that brought by dataset itself.

Observation 2: Integration w.r.t both regression and classification are useful. In terms of training convergence, we observe that the accuracy or mean absolute error converges quickly to a high or low score respectively. For small graph datasets **BACE** and **BBBP** on graph classification problem, an improvement of $\approx$**1%** on average accuracy is observed with method `MPNN2LM` for pretrained ChemBERTa. For version 2, 1.4% improvement is observed with late fusion on **BACE** and 1.3% improvement is observed with node contrastive learning on **BBBP** . For small graph dataset **ESOL** on regression problem, a great improvement is observed where 12.8% improvement on mae with MPNN2LM method with pretrained ChemBERTa, and 7.3% improvement on mae with MPNN2LM method with pretrained ChemBERTa-2. For **HIV**, we observe a little improvement with node level contrastive learning. Using a combination of LLM representation and graph representation during the training would make the prediction worse. For **QM9**, most of the injection / fusion methods would potentially improve the performance except for MPNN2LM fusion. Using LM2MPNN would potentially improve 8.6 %. We found that pure MPNN's performance is better than the performance of a chemical LLM (table 4).

Observation 3: Pretrained language models are important for downstream predictions. In comparison to ChemBERTa-2, ChemBERTa performs worse when comparing each entry in table 1 and table 2. Although we could always try to improve those two baselines with different injection or fu-

sion methods, the best of them are not the same. For example, contrastive learning is much more preferred to ChemBERTa-2 while fusion methods are much more preferred to ChemBERTa model. When it comes with a new pretrained large language model, using our proposed method could tell the similarity between tasks and the model's pretraining strategy. As there is no general conclusion about how a chemical LLM and a MPNN could be combined to predict the best, it is still a pioneering area that requires more pretrained models to test its robustness. To select the most appropriate pretrained language model for further training, researchers should first integrate a list of pretrained models, followed by an investigation with different fusion / injection methods.

Observation 4: Joint Fusion to some extent helps learn MPNN better but learn original Chemical LLM worse. We also focus on if such multimodal module (Fig. 2, Fig. 3) helps learn individual module (Fig. 1) better. It improves a lot for single MPNN baseline if we consider its language level information as augmented features. For example, for **BACE** dataset, MPNN has an average accuracy of 0.808 with ChemBERTa. With injecting pretrained language information, an improvement of 3.7% is observed (LM2MPNN). However, it might not work very well on the opposite when we inject information from MPNN to LLM. For simplicity we just examined with pretrained ChemBERTa-2. For ESOL dataset, it decreased from 0.4738 to 0.5012 (5.78%). For BBBP dataet, it decreased from 0.9171 to 0.8974 (2.15%). We further suggest that the researchers should not directly use the structural information from graphs as additional input when they want to modify their LLM models, but trying to leverage them as auxiliary ground-truth to finetune the token embeddings.

Model	QM9 (target = 0)
ChemBERTa-2 baseline	0.4825 ± 0.0113
MPNN baseline	0.4669 ± 0.0065
ChemBERTa-2 contra. MPNN (node)	0.4613 ± 0.0065
ChemBERTa-2 contra. MPNN (graph)	0.4662 ± 0.0046
ChemBERTa-2 + MPNN (graph)	0.4596 ± 0.0078
ChemBERTa-2 $\leftarrow$ MPNN	0.5231 ± 0.0083
ChemBERTa-2 $\rightarrow$ MPNN	$\mathbf{0.4409 \pm 0.0048}$

Table 4: Performance of improved pretrained ChemBERTa-2 on QM9 dataset.

6 Ablation Study

Effects of datasets. We choose another dataset in MoleculeNet to certify that the proposed models are still robust on this dataset. Take **FreeSolv** as an example, we figure out that none of the injection or contrastive learning methods is still robust on this regression task. Even if late fusion performs the best which has an average mae of 0.6568, which is close to the result of pure chemical LLM training (0.6420), there's still a 2.3% decrease in performance. Both `LM2MPNN` and `MPNN2LM` did not work well, but it still commits to our fourth main observation, which is that injecting token embeddings into message passing layers would still improve the performance, but injecting structural information into word embeddings would be a bad idea. A potential reason is that **FreeSolv** is too small. We suggest that researchers should be careful when fine-tuning the individual language model with additional structural features.

Model	FreeSolv (**mae.**) $\downarrow$
ChemBERTa-2 baseline	0.6420 ± 0.0814
MPNN baseline	0.9904 ± 0.1375
ChemBERTa-2 contra. MPNN (node)	0.6642 ± 0.0600
ChemBERTa-2 contra. MPNN (graph)	0.6745 ± 0.0995
ChemBERTa-2 + MPNN (graph)	0.6568 ± 0.0658
ChemBERTa-2 $\leftarrow$ MPNN	0.9188 ± 0.0686
ChemBERTa-2 $\rightarrow$ MPNN	0.7475 ± 0.0805

Table 5: Performance of improved pretrained ChemBERTa-2 on FreeSolv dataset

Effects of dataset split. We want to figure out if different splits of training, validation and test datasets lead to different performance. We run on **BBBP** (classification) and **ESOL** (regression). Four ratios are considered: 9:0.5:0.5, 8:1:1, 7:2:1, and 6:2:2. Model prediction power is highest at a ratio of 8:1:1 for **ESOL** while the prediction power is reducing for **BBBP** when ratio of training sets is decreasing.

Train test split	BBBP (**acc.**) $\uparrow$	ESOL (**mae.**) $\downarrow$
9: 0.5 : 0.5	$\mathbf{0.9500 \pm 0.0174}$	0.4672 ± 0.0338
8 : 1 : 1	0.9290 ± 0.0128	$\mathbf{0.4393 \pm 0.0338}$
7 : 2 : 1	0.9211 ± 0.0110	0.4837 ± 0.0447
6 : 2 : 2	0.9152 ± 0.0032	0.4947 ± 0.0060

Table 6: Model (node level contrast.)

Effects of different fusion operations $\bigoplus$ We first follow the default train valid test split of 8:1:1. As mentioned, there are four fusion operations $\bigoplus$,

which are max, sum, concatenation and gate function. Our default fusion operation is sum function. Surprisingly we found that concatenation and max function are better fusion choice for both **BBBP** and **ESOL**. We suggest that researchers could simply concatenate token embeddings and graph embeddings together.

Fusion Operation	BBBP (**acc.**) ↑	ESOL (**mae.**) ↓
sum	0.9263 ± 0.0113	0.4493 ± 0.0328
max	0.9289 ± 0.0146	0.4281 ± 0.0339
concate	$\mathbf{0.9289 \pm 0.0241}$	$\mathbf{0.4255 \pm 0.0335}$
gate	0.9224 ± 0.0197	0.4363 ± 0.0354

Table 7: Model: Late Fusion

Effects of different graph neural networks As mentioned in section 3, there are three types of graph neural networks in mainstream GNN research, which are graph convolution (GraphConv), message passing neural networks (MPNN), and graph attention networks. We substitute MPNN with with a two-layer GraphConv model to see if MPNN is much better than other types of GNN for baselines. The results show that MPNN is more preferred to **BBBP** but GraphConv is more preferred to **ESOL**. Overall the difference would not be too large for a graph convolution network and a neural message passing layer therefore we suggest researchers try out both ways to improve the results.

Fusion Operation	BBBP (**acc.**) ↑	ESOL (**mae.**) ↓
MPNN	$\mathbf{0.9289 \pm 0.0146}$	0.4281 ± 0.0339
GraphConv	0.9237 ± 0.0148	$\mathbf{0.4144 \pm 0.0252}$

Table 8: Model: Late Fusion

7 Conclusion

In this paper, we delved into various information integration approaches to assess whether the collaborative utilization of chemical large language models (chemical LLMs) and message passing neural networks (MPNNs) surpasses the individual efficacy of these models. We evaluated the integration approaches over different graph scales on both classification and regression tasks. Our empirical analysis has demonstrated that the integration approaches outperform the baselines on small-scale graphs but do not yield improvements on datasets of larger scales. Furthermore, we have found that differences in dataset splitting strategies, and aggregation choices in fusion have an impact on the overall performance. We wish to extend our proposed methods on large scale benchmark datasets such as PCQM4Mv2 (Hu et al., 2020a).

References

Walid Ahmad, Elana Simon, Seyone Chithrananda, Gabriel Grand, and Bharath Ramsundar. 2022. Chemberta-2: Towards chemical foundation models.

Viraj Bagal, Rishal Aggarwal, PK Vinod, and U Deva Priyakumar. 2021. Molgpt: molecular generation using a transformer-decoder model. *Journal of Chemical Information and Modeling*, 62(9):2064–2076.

Zhikai Chen, Haitao Mao, Hang Li, Wei Jin, Hongzhi Wen, Xiaochi Wei, Shuaiqiang Wang, Dawei Yin, Wenqi Fan, Hui Liu, et al. 2024. Exploring the potential of large language models (llms) in learning on graphs. *ACM SIGKDD Explorations Newsletter*, 25(2):42–61.

Seyone Chithrananda, Gabriel Grand, and Bharath Ramsundar. 2020. Chemberta: large-scale self-supervised pretraining for molecular property prediction. *arXiv preprint arXiv:2010.09885*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Ahmed Elnaggar, Michael Heinzinger, Christian Dallago, Ghalia Rehawi, Yu Wang, Llion Jones, Tom Gibbs, Tamas Feher, Christoph Angerer, Martin Steinegger, et al. 2021. Prottrans: Toward understanding the language of life through self-supervised learning. *IEEE transactions on pattern analysis and machine intelligence*, 44(10):7112–7127.

Benedek Fabian, Thomas Edlich, Héléna Gaspar, Marwin Segler, Joshua Meyers, Marco Fiscato, and Mohamed Ahmed. 2020. Molecular representation learning with language models and domain-relevant auxiliary tasks.

Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. 2017. Neural message passing for quantum chemistry. In *International conference on machine learning*, pages 1263–1272. PMLR.

Francesco Giuliari, Irtiza Hasan, Marco Cristani, and Fabio Galasso. 2021. Transformer networks for trajectory forecasting. In *2020 25th international conference on pattern recognition (ICPR)*, pages 10335–10342. IEEE.

Liang He, Shizhuo Zhang, Lijun Wu, Huanhuan Xia, Fusong Ju, He Zhang, Siyuan Liu, Yingce Xia, Jianwei Zhu, Pan Deng, et al. 2021. Pre-training co-evolutionary protein representation via a pairwise masked language model. *arXiv preprint arXiv:2110.15527*.

Shion Honda, Shoi Shi, and Hiroki R Ueda. 2019. Smiles transformer: Pre-trained molecular fingerprint for low data drug discovery. *arXiv preprint arXiv:1911.04738*.

Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. 2020a. Open graph benchmark: Datasets for machine learning on graphs. *Advances in neural information processing systems*, 33:22118–22133.

Ziniu Hu, Yuxiao Dong, Kuansan Wang, Kai-Wei Chang, and Yizhou Sun. 2020b. Gpt-gnn: Generative pre-training of graph neural networks. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1857–1867.

Wen-Chin Huang, Chia-Hua Wu, Shang-Bao Luo, Kuan-Yu Chen, Hsin-Min Wang, and Tomoki Toda. 2021. Speech recognition by simply fine-tuning bert. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7343–7347. IEEE.

Klicpera Johannes, Groß Janek, and Günnemann Stephan. 2020. Directional message passing for molecular graphs. In *International Conference on Learning Representations*.

Shengchao Liu, Hanchen Wang, Weiyang Liu, Joan Lasenby, Hongyu Guo, and Jian Tang. 2021. Pre-training molecular graph representation with 3d geometry. In *International Conference on Learning Representations*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

Shengjie Luo, Tianlang Chen, Yixian Xu, Shuxin Zheng, Tie-Yan Liu, Liwei Wang, and Di He. 2022. One transformer can understand both 2d & 3d molecular data. In *The Eleventh International Conference on Learning Representations*.

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.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Roshan Rao, Nicholas Bhattacharya, Neil Thomas, Yan Duan, Peter Chen, John Canny, Pieter Abbeel, and Yun Song. 2019. Evaluating protein transfer learning with tape. *Advances in neural information processing systems*, 32.

Alexander Rives, Joshua Meier, Tom Sercu, Siddharth Goyal, Zeming Lin, Jason Liu, Demi Guo, Myle Ott, C Lawrence Zitnick, Jerry Ma, et al. 2021. Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. *Proceedings of the National Academy of Sciences*, 118(15):e2016239118.

Yu Rong, Yatao Bian, Tingyang Xu, Weiyang Xie, Ying Wei, Wenbing Huang, and Junzhou Huang. 2020. Self-supervised graph transformer on large-scale molecular data. *Advances in Neural Information Processing Systems*, 33:12559–12571.

Devendra Sachan, Yuhao Zhang, Peng Qi, and William L Hamilton. 2021. Do syntax trees help pre-trained transformers extract information? In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2647–2661.

Hannes Stärk, Dominique Beaini, Gabriele Corso, Prudencio Tossou, Christian Dallago, Stephan Günnemann, and Pietro Liò. 2022. 3d infomax improves gnns for molecular property prediction. In *International Conference on Machine Learning*, pages 20479–20502. PMLR.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2023. Attention is all you need.

Sheng Wang, Yuzhi Guo, Yuhong Wang, Hongmao Sun, and Junzhou Huang. 2019a. Smiles-bert: large scale unsupervised pre-training for molecular property prediction. In *Proceedings of the 10th ACM international conference on bioinformatics, computational biology and health informatics*, pages 429–436.

Sheng Wang, Yuzhi Guo, Yuhong Wang, Hongmao Sun, and Junzhou Huang. 2019b. Smiles-bert: Large scale unsupervised pre-training for molecular property prediction. In *Proceedings of the 10th ACM International Conference on Bioinformatics, Computational Biology and Health Informatics*, BCB '19, page 429–436, New York, NY, USA. Association for Computing Machinery.

Yuyang Wang, Jianren Wang, Zhonglin Cao, and Amir Barati Farimani. 2022. Molecular contrastive learning of representations via graph neural networks. *Nature Machine Intelligence*, 4(3):279–287.

Zhenqin Wu, Bharath Ramsundar, Evan N Feinberg, Joseph Gomes, Caleb Geniesse, Aneesh S Pappu, Karl Leswing, and Vijay Pande. 2018. Moleculenet: a benchmark for molecular machine learning. *Chemical science*, 9(2):513–530.

Jun Xia, Chengshuai Zhao, Bozhen Hu, Zhangyang Gao, Cheng Tan, Yue Liu, Siyuan Li, and Stan Z Li. 2022. Mole-bert: Rethinking pre-training graph neural networks for molecules. In *The Eleventh International Conference on Learning Representations*.

Minghao Xu, Zuobai Zhang, Jiarui Lu, Zhaocheng Zhu, Yangtian Zhang, Ma Chang, Runcheng Liu, and Jian Tang. 2022. Peer: a comprehensive and multi-task benchmark for protein sequence understanding. *Advances in Neural Information Processing Systems*, 35:35156–35173.

Chengxuan Ying, Tianle Cai, Shengjie Luo, Shuxin Zheng, Guolin Ke, Di He, Yanming Shen, and Tie-Yan Liu. 2021. Do transformers really perform badly for graph representation? *Advances in Neural Information Processing Systems*, 34:28877–28888.

Yuning You, Tianlong Chen, Yang Shen, and Zhangyang Wang. 2021. Graph contrastive learning automated. In *International Conference on Machine Learning*, pages 12121–12132. PMLR.

Yanqiao Zhu, Dingshuo Chen, Yuanqi Du, Yingze Wang, Qiang Liu, and Shu Wu. 2022. Featurizations matter: a multiview contrastive learning approach to molecular pretraining. In *ICML 2022 2nd AI for Science Workshop*.