Dual-view Molecule Pre-training

Jinhua Zhu [1], **Yingce Xia** [2], **Tao Qin** [2], **Wengang Zhou** [1], **Houqiang Li** [1], **Tie-Yan Liu** [2]
[1] University of Science and Technology of China [2]Microsoft Research Asia
teslazhu@mail.ustc.edu.cn, {zhwg,lihq}@ustc.edu.cn
{yingce.xia,taoqin,tyliu}@microsoft.com

Abstract

Inspired by its success in natural language processing and computer vision, pre-training has attracted substantial attention in cheminformatics and bioinformatics, especially for molecule based tasks. A molecule can be represented by either a graph (where atoms are connected by bonds) or a SMILES sequence (where depth-first-search is applied to the molecular graph with specific rules). Existing works on molecule pre-training use either graph representations only or SMILES representations only. In this work, we propose to leverage both the representations and design a new pre-training algorithm, dual-view molecule pre-training (briefly, DMP), that can effectively combine the strengths of both types of molecule representations. The model of DMP consists of two branches: a Transformer branch that takes the SMILES sequence of a molecule as input, and a GNN branch that takes a molecular graph as input. The training of DMP contains three tasks: (1) predicting masked tokens in a SMILES sequence by the Transformer branch, (2) predicting masked atoms in a molecular graph by the GNN branch, and (3) maximizing the consistency between the two high-level representations output by the Transformer and GNN branches separately. After pre-training, we can use either the Transformer branch (this one is recommended according to empirical results), the GNN branch, or both for downstream tasks. DMP is tested on nine molecular property prediction tasks and achieves state-of-the-art performances on seven of them. Furthermore, we test DMP on three retrosynthesis tasks and achieve state-of-the-result on the USPTO-full dataset. Our code will be released soon.

1 Introduction

While machine learning (especially deep learning) techniques have been applied to cheminformatics and bioinformatics with significant progress [25, 7], their potential is severely limited by the scale of labeled data since it is more costly and time consuming to collect labeled data for tasks in cheminformatics and bioinformatics than those tasks in computer vision (CV) and natural language processing (NLP). Inspired by the great success of exploiting unlabeled data in CV and NLP [20, 16], pre-training has been introduced into cheminformatics [23, 3, 53] and bioinformatics [42, 44]. Among those pre-training works, molecule pre-training has attracted much attention since molecules are a kind of basic units and play a fundamental role in drug discovery and chemical modeling.

Molecules can be represented in different formats. As shown in Figure 2, a molecule can be represented by a molecular graph, where nodes and edges are atoms and bonds respectively. This is the most intuitive way to visualize a molecule. By traversing the graph using depth first search and some predefined rules, the same molecule can be represented by a SMILES[1] [54] sequence. In almost all molecular database like PubChem [28] and ZINC [27], SMILES representations are used due to its simplicity. Correspondingly, there are mainly two lines of works on molecule pre-training, which

[1]SMILES is short for simplified molecular-input line-entry system.

Preprint. Under review.

are built upon graphs or SMILES sequences. For pre-training on graphs [23, 34, 53], graph neural networks (GNNs) are used as backbone models, which explicitly leverages the structural information of a molecule. Hu et al. [23] proposed two pre-training techniques based on GNNs: one is to predict masked atoms or bonds, and the other is to preserve the similarity between a K-hop subgraph of some anchor point v_a and its corresponding context graph (i.e., a ring surrounding v_a). Wang et al. [53] applied contrastive learning [17] to molecule pre-training, where the representations of a molecule extracted by a GNN are forced to be similar to another augmented version of itself while dissimilar to other molecules. For pre-training on SMILES sequences [3], they are treated in a similar way to natural language sequence and Transformer [51], a widely used model in NLP, is adopted as the backbone.

Although different methods have been proposed for molecule pre-training, to our best knowledge, almost all of them only use one kind of molecule representations, either dealing with graphs using GNN only or dealing with SMILES sequences using Transformer only. However, the two types of models have their own strengths and limitations, as shown in Figure 1. Transformer correctly predicts the properties of molecules in Figure 1(a), which have long chains without too many structured groups (e.g., more than three rings concatenated together), but fails on the molecules in Figure 1(b), which have relatively short chains but contain rich structured groups. In contrast, GNN correctly classifies the molecules in Figure 1(b) but fails in Figure 1(a). This suggests that the two views are complementary to each other.

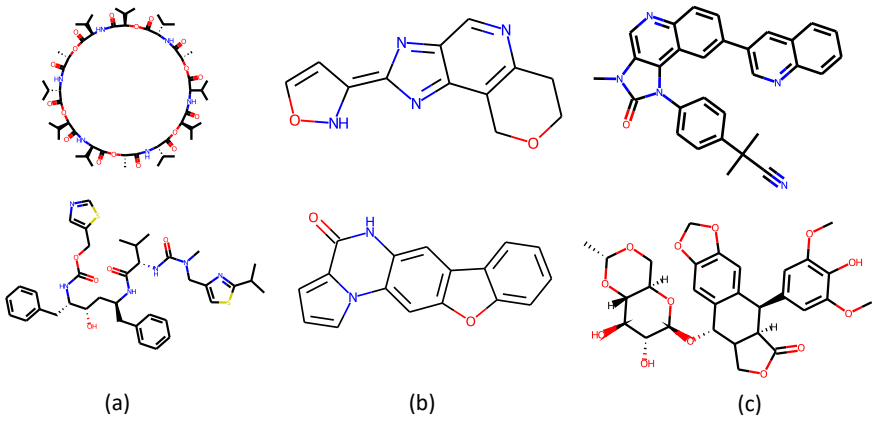

Figure 1: Examples of molecular property prediction from MoleculeNet [55]. (a) Transformer succeeds while GNN fails; (b) GNN succeeds while Transformer fails; (c) both the standard Transformer and GNN fail while our method succeeds.

In this work, we propose a novel pre-training method, *dual-view molecule pre-training* (briefly, DMP), to combine the best of two worlds. In DMP, a molecule M is represented by both a SMILES sequence M_s and a graph M_g. M_s is encoded by a Transformer branch which outputs a high-level representation f_s, and M_g is encoded by a GNN branch which outputs another high-level representation f_g. Since f_s and f_g are representations of the same molecule, they should be similar in some latent space. To achieve this, inspired by the BYOL scheme [15], we propose the *dual-view consistency objective*, where the cosine similarity between the projected variants f_s and f_g should be maximized. In addition, we also adopt the masked language modeling (MLM) [23, 3, 9] objective. Specifically, for the Transformer branch, we randomly mask some tokens (e.g., atoms or bonds) in a SMILES sequence and recover them; for the GNN branch, we randomly mask some atoms and reconstruct them. After pre-training, we can use the Transformer branch, the GNN branch or both for downstream tasks. We recommend using the Transformer branch according to empirical results, which do not introduce extra parameters for downstream tasks compared to standard Transformer while the accuracy is promising.

To test DMP, we first pre-train on $10M$ molecules from PubChem[2] following previous work [3, 53], and then finetune on 9 molecular property prediction tasks from MoleculeNet [55] and three retrosynthesis tasks. We achieve state-of-the-art results on 7 out of 9 prediction tasks, which demonstrate the effectiveness of our algorithm. Specifically, on the classification tasks of MoleculeNet, DMP outperforms MolCLR [53] and GROVER [46] by 1.2 and 2.2 points on average; on retrosynthesis, we achieve $2 \sim 3$-point improvements of top-1 accuracy across 3 different settings and achieve state-of-the-art results on USPTO-full [6, 56, 49]. After that, we conduct pre-training on $100M$ compounds from PubChem, and we find the results can be slightly improved. After using our method, we can successfully predict the properties of molecules with long chains and rich structured groups (as shown in Figure 1(c)), where both standard Transformer only and GNN fail.

Our main contributions can be summarized as follows:

(1) To our best knowledge, we are the first to conduct molecule pre-training taking the advantages of the two different views (i.e., SMILES and molecular graphs).

(2) In addition to MLM, DMP leverages dual-view consistency loss for pre-training, which explicitly exploits the consistency of representations between two views of molecules.

(3) We achieve state-of-the-art results on 7 molecular property prediction tasks from MoleculeNet [55] and one retrosynthesis task (i.e., a kind of molecule generation task), USPTO-full, demonstrating the effectiveness and generalization ability of DMP.

2 Related work

How to represent a molecule by a high-dimensional vector has attracted lots of attentions. Rogers & Hahn [45] proposed extended-connectivity figureprints (ECFP), a kind of circular fingerprints obtained by iteratively encoding the neighbors of atoms. The MACCS (Molecular ACCess System) keys [12] are another commonly used structural descriptors, in which each bit is associated with a predefined SMARTS pattern. As the development of deep learning, molecule fingerprints are learned by neural networks [38, 13]. After the emergence and success of pre-training in NLP and CV, people introduce it to molecule representation.

Transformer-based pre-training: Transformer [51] is originally proposed for sequence learning in NLP, and then widely adopted by other applications, such as speech processing [10, 43], CV [11, 40], and also, cheminformatics. To leverage Transformer-based molecule pre-training, people regard SMILES sequences as natural languages and apply the techniques of pre-training in NLP. Similar to [9, 35] which mainly uses masked language modeling, Wang et al. [52] proposed SMILES-BERT, a 6-layer pre-training model. Chithrananda et al. [3] further extended the pre-trained model to 12 layers and systematically studied the results on MoleculeNet [55]. Honda et al. [22] used auto-encoder to obtain a neural-based fingerprint. Maziarka et al. [37] improved self-attention with inter-atomic distances and molecular graph structure. In addition, there are some works about SMILES-based molecule generation [47, 41] and optimization [18].

GNN-based pre-training: Hu et al. [23] used a GNN to encode the input molecule, and proposed two pre-training strategies, where we should either recover the masked attributes of the input (e.g., atom type), or use constrastive learning [17, 2] to minimize the difference between two subgraphs of within a molecule. A similar idea also exists in Li et al. [32]. Wang et al. [53] applied contrastive learning across different molecules and proposed MolCLR, where a molecule should be similar to an augmented version of itself while dissimilar to others. Liu et al. [34] proposed an N-gram graph to represent molecules: it first learns the representations of atoms using CBoW [39], and then enumerates N-grams (i.e., a walk of length N) in the graph and the final representation is constructed based on the embeddings of all its N-grams. Rong et al. [46] proposed GROVER, which replaces the self-attention layer in Transformer where the number of hops are adaptively determined, and design new pre-training objective functions: one about predicting the statistics of the molecule and the other is the predict the existence of some function groups. To our best knowledge, almost all previous works treat Transformer-based pre-training and GNN-based pre-training independently, and we propose to leverage them together.

[2] `https://pubchem.ncbi.nlm.nih.gov/`

3 Our method

In this section, we first introduce the network architecture, followed by the training objective functions of our method, and finally provide discussions about related methods and limitations.

3.1 Network architecture

Given a molecule M, let M_s and M_g denote the SMILES and molecular graph respectively. Define M_s as $(m_1, m_2, \cdots, m_l)$, where l is the length of M_s. Define M_g as a graph (V_g, E_g), where $V_g = \{v_1, v_2, \cdots, v_n\}$ is a collection of atoms, and E_g is the collection of edges[3]. In a molecule, the bond has several types (e.g., single bond, double bonds, aromatic compounds), which is also modeled.

The network architecture of our method consists of two branches, one is a GNN, which is used to encode its graph view M_g, and the other one is a Transformer, which is to encode its sequence view M_s. Denote the above two models as φ_g and φ_s, which have L_g and L_s layers, respectively.

(1) For the GNN branch, following [31], we choose the DeeperGCN network as the backbone. DeeperGCN is a stack of multiple GCN layers [30], where the batch normalization [26], non-linear activation and residual connections [19] are all used. The information among atoms are propagated along bonds, and the embedding of bond property (i.e., double bond, single bond) is added to atoms. Details of DeeperGCN are left in Appendix A.1 of the supplementary document. After encoded by the L_g layers, each atom has a representation outputted by the last layer of DeeperGCN, i.e., $\{h_1^g, h_2^g, \cdots, h_n^g\} = \varphi_g(M_g)$, where h_i^g is the representation of the i-th atom. We then choose a pooling function pool (e.g., mean pooling, max pooling) and obtain the representation of the molecule, i.e., $f_g = \mathrm{pool}(h_1^g, h_2^g, \cdots, h_n^g)$.

(2) For the Transformer branch, we choose the RoBERTa model [35], which encodes the SMILES sequence M_s of the molecule. Following [35, 9], we add a special token `[CLS]` to the beginning of the sequence. We also use the output of the last layer as the representations of `[CLS]` and tokens, i.e., $\{h_0^s, h_1^s, h_2^s, \cdots, h_l^s\} = \varphi_s(M_s)$, where h_0^s corresponds to `[CLS]` and h_j^s corresponds to the j-th element in the SMILES sequence. In this way, the molecule representation of M_s is h_0^s, i.e., $f_s = h_0^s$.

3.2 Training objective functions

Our method consists of three training objective functions, including two masked language modeling (MLM) loss and one dual-view consistency loss.

(1) **MLM on Transformer**: Given a SMILES sequence, we randomly mask some tokens, and the objective is to recover the original tokens, following the practice in NLP pre-training [9, 35].

(2) **MLM on GNN**: Similar to the MLM on Transformer, we randomly mask some atoms in a molecular graph (the bonds remain unchanged), and the objective is to recover the original atoms, following the practice in graph pre-training [23].

(3) **Dual-view consistency**: To model the interaction between the GNN and the Transformer, inspired by the BYOL [15], we propose a dual-view consistency loss, that models the similarity of the output between GNN and Transformer. Based on the empirical discovery of [15], we introduce two non-linear projection layers ψ_g, ψ_s and two prediction layers ρ_g and ρ_s. For the SMILES view M_s and the graph view M_g of a molecule, we randomly mask some tokens/atoms and obtain $\tilde{M}_s$ and $\tilde{M}_g$. f_g is obtained by $\mathrm{pool}(\varphi_g(\tilde{M}_g))$ and f_s is the first output of $\varphi_s(\tilde{M}_s)$, which corresponds to the `[CLS]` token. After that, we apply the projection and prediction layers to them, i.e.,

$$p_g = \psi_g(f_g),\ q_g = \rho_g(p_g);\ p_s = \psi_s(f_s),\ q_s = \rho_s(p_s). \tag{1}$$

Since $p_\cdot$'s and $q_\cdot$'s are the representation of the same molecule but from different views, they should preserve enough similarity. Let $\mathtt{cos}(p, q)$ denote the cosine similarity between p and q, i.e., $\mathtt{cos}(p, q) = (p^\top q)/(\|p\|_2 \|q\|_2)$. Following [15], the dual-view consistency loss is defined as

$$\ell_{\mathrm{dual}}(\tilde{M}_g, \tilde{M}_s; \varphi_g, \varphi_s, \psi_g, \psi_s, \rho_g, \rho_s) = -\mathtt{cos}(q_s, \mathrm{SG}(p_g)) - \mathtt{cos}(q_g, \mathrm{SG}(p_s)). \tag{2}$$

[3] $l \geq n$ since the bonds, numbers and brackets are also included in the SMILES as shown in Figure 2.

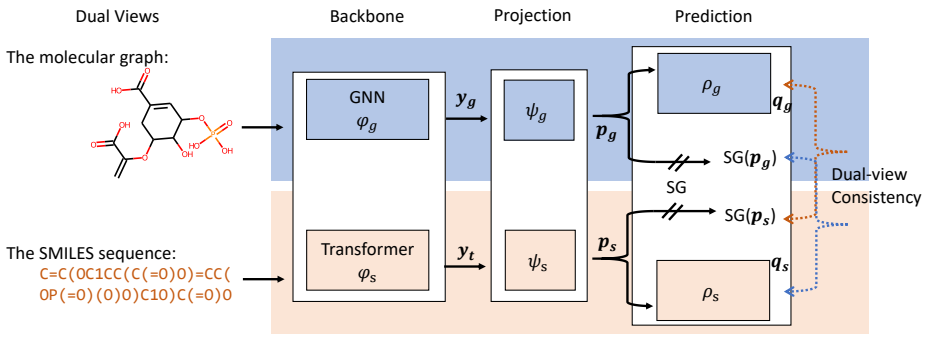

Figure 2: The illustration of our dual-view consistency loss. "SG" indicates the stop-gradient when back-propagating.

In Eqn.(2), $\mathrm{SG}(p_g)$ means that the gradient is not applied to p_g when back-propagating, and neither is $\mathrm{SG}(p_s)$.

After pre-training, we can use the Transformer branch, the GNN branch or both for downstream tasks. According to our empirical study, for molecular prediction tasks, we recommend using the Transformer branch. Using both branches brings further improvement at the cost of larger model size.

3.3 Discussions

(1) *Relation with co-training*: Co-training [1] is a semi-supervised learning method that can be used to construct additional labeled data. The basic assumption to use co-training is that the features can be divided into two conditionally independent sets, where each set is sufficient to train a classifier. Each classifier provides the most confident predictions on the unlabeled data as the additional labeled data. In comparison, DMP is an algorithm which does not need the independent assumption. DMP provides a pre-trained model for downstream tasks instead of newly labeled data.

(2) *Limitations of our method*: Our method has two limitations. First, there is a Transformer branch and a GNN branch in our method, which increases the training cost comparing with previous single-branch pre-training [3, 53]. How to design an efficient pre-training method is an interesting future direction. Second, in downstream tasks, we deal with all molecules using either the Transformer branch or the GNN branch. Recent studies show that a better solution is to use a meta-controller to dynamically determine which branch to use [59, 14] for an individual input. We will also explore this dynamic branch selection in future.

4 Experiments

In this section, first, we introduce the data and model architecture for pre-training. Second, we apply our pre-trained models to nine molecular property prediction tasks from MoleculeNet [55]. Next, we evaluate our method on three retrosynthesis tasks, including USPTO-50k with reaction type known/unknown and USPTO-full [49]. Finally, we visualize the representations of different pre-trained models for comparison.

4.1 Pre-training

Dataset For pre-training, we choose two subsets from PubChem, one with $10M$ compounds which are the same as those in [53, 3], and the other with $100M$ compounds. We first use the RDkit toolkit[4] to process the input. For the input of Transformer branch, we canonicalize SMILES sequences and then tokenize the canonicalized SMILES using the same regular expression as [47]. For the input of the GNN branch, we convert the SMILES to molecular graphs using RDKit.

[4] `https://github.com/rdkit/rdkit`

Model Architecture The Transformer branch is the same as the RoBERTa$_{\text{base}}$ architecture, which consists of 12 layers. The hidden dimension, the dimension of the feed-forward layer, and the number of heads are 768, 3072 and 12, respectively. The GNN branch follows the DeeperGCN [31] backbone, which is a 12-layer network. The hidden dimension of the GNN branch is 384. The graph pooling function above the last layer is the concatenation of `mean` and `max` operations.

Following [2, 15], the projection heads (i.e., ψ_g and ψ_s) are 3-layer MLP networks, and the prediction heads (i.e., ρ_g and ρ_s) are 2-layer MLP networks. All hidden layers in the MLPs are followed by the `ReLU` and `BatchNorm` [26], while the output layers are not.

Optimization Our model is optimized by Adam [29] algorithm with learning rate 5×10^{-4}, $\beta_1 = 0.9$, $\beta_2 = 0.98$, $\epsilon = 10^{-6}$. The weight decay is 0.01. The learning rate is warmed up in the first $10k$ update steps and then linearly decayed. All pre-training experiments are conducted on $8\times$V100 GPUs and the batch size is 12288 tokens. The gradients are accumulated for 16 times. The models are trained for $200k$ iterations.

4.2 Molecular Property Prediction

Dataset After pre-training, we first finetune our model on 6 datasets from MoleculeNet [55], a popular well-curated benchmark for molecular property prediction. Note that each dataset might be associated with multiple tasks. There are 44 tasks in total, and they cover a wide range of data volumes. On one hand, following [53], we use the official training, validation and test sets provided by DeepChem [5], whose performance is relatively stable and reproducible. On the other hand, to compare with [46, 32], we follow their ways to split the data and conduct another group of experiments. Specifically, we use *scaffold splitter* to generate the training (80%), validation (10%) and test (10%) sets with 3 different seeds. We compare with [46, 32] on three classification and three regression tasks.

Finetuning settings Following [34, 53], each task is independently finetuned. We explore finetuning on the Transformer branch, the GNN branch and both. The classification/regression head is a 2-layer MLP with `tanh` activation, which takes the representation of `[CLS]` (i.e., f_s) and/or the graph pooling over GNN (i.e., f_g) as input. Detailed hyper-parameters are left in Appendix A.2.

Evaluation Following [55], for classification tasks, we use the area under curve (AUC) of the receiver operating characteristic (ROC) curve as the evaluation metric. For regression tasks, we use the root mean square error (RMSE) or the mean absolute error (MAE) for evaluation depending on the previous work. For datasets containing more than one task, we report the average scores across all tasks. Each task is independently run for 3 times with different random seeds, and the mean and standard derivation of the performances are reported.

Baselines We compare our method with the following baselines:

(1) We compare DMP with four methods without pre-training. Two of them are about applying Random Forest (RF) [21] and Support Vector Machine (SVM) [5] to molecular fingerprints. The other two are GNN-based method, D-MPNN [57] and MGCNN [36], which are specifically designed for molecular property prediction. We also compare our model with previous pre-training methods, including Hu et al. [23], MolCLR [53], GROVER [46] and MPG [32].

(2) We train a standard Transformer and GNN using MLM only, which are denoted as TF (MLM) and GNN (MLM). TF (MLM) is an enhanced implementation of ChemBERTa [3], where we train models with more GPUs and larger batch size, and eventually obtain better results (see Appendix B.1 for more details). For GNN (MLM), we only mask the atoms randomly, without applying bond deletion and subgraph removal, which are left as future work. Also, we implement a variant of DMP, "DMP w/o MLM", where we only use dual-view loss and do not use MLM.

(3) To investigate the effectiveness of using two heterogeneous models, we implement another variants, where the two branches are both Transformer or both GNNs. The MLM loss and dual-view consistency loss are both applied. We randomly choose one branch for finetuning. Denote these variants as TF ($\times$2) and GNN ($\times$2).

Results The results of the official test from DeepChem are shown in Table 1, and those of the test sets from GROVER [46] are in Table 2. After pre-training with DMP, denote the results finetuned

[5] `https://github.com/deepchem/deepchem`

Dataset # Molecules	BBBP 2039	Tox21 7831	ClinTox 1478	HIV 41127	BACE 1513	SIDER 1478
RF	71.4 ± 0.0	76.9 ± 1.5	71.3 ± 5.6	78.1 ± 0.6	86.7 ± 0.8	68.4 ± 0.9
SVM	72.9 ± 0.0	81.8 ± 1.0	66.9 ± 9.2	79.2 ± 0.0	86.2 ± 0.0	68.2 ± 1.3
MGCN [36]	$\mathbf{85.0 \pm 6.4}$	70.7 ± 1.6	63.4 ± 4.2	73.8 ± 1.6	73.4 ± 3.0	55.2 ± 1.8
D-MPNN [57]	71.2 ± 3.8	68.9 ± 1.3	90.5 ± 5.3	75.0 ± 2.1	85.3 ± 5.3	63.2 ± 2.3
Hu et al. [23]	70.8 ± 1.5	78.7 ± 0.4	78.9 ± 2.4	80.2 ± 0.9	85.9 ± 0.8	65.2 ± 0.9
MolCLR [53]	73.6 ± 0.5	$\mathbf{79.8 \pm 0.7}$	93.2 ± 1.7	80.6 ± 1.1	89.0 ± 0.3	68.0 ± 1.1
TF (MLM)	74.9 ± 0.6	77.6 ± 0.4	92.9 ± 0.5	80.2 ± 0.4	88.0 ± 0.5	68.4 ± 0.4
TF ($\times 2$)	75.6 ± 0.7	77.1 ± 0.5	92.0 ± 0.8	80.4 ± 0.4	88.1 ± 0.5	68.2 ± 1.2
DMP w/o MLM⇒TF	71.1 ± 0.4	75.7 ± 0.4	93.8 ± 0.7	79.1 ± 1.7	88.3 ± 0.7	68.1 ± 0.7
DMP⇒TF	78.1 ± 0.5	78.8 ± 0.5	$\mathbf{95.0 \pm 0.5}$	$\mathbf{81.0 \pm 0.7}$	$\mathbf{89.3 \pm 0.9}$	$\mathbf{69.2 \pm 0.7}$
GNN (MLM)	74.5 ± 0.3	74.8 ± 0.5	92.3 ± 0.7	78.5 ± 0.5	84.1 ± 0.4	67.0 ± 0.5
GNN ($\times 2$)	74.1 ± 0.6	75.1 ± 0.3	92.8 ± 0.7	79.2 ± 0.9	85.1 ± 1.0	69.0 ± 0.4
DMP⇒GNN	74.7 ± 0.2	76.7 ± 0.3	94.2 ± 0.4	79.5 ± 1.0	85.7 ± 0.8	68.4 ± 0.5
DMP⇒GNN+TF	77.8 ± 0.3	79.1 ± 0.4	95.6 ± 0.7	81.4 ± 0.4	89.4 ± 0.8	69.8 ± 0.6
DMP(100M)⇒TF	78.4 ± 0.3	79.0 ± 0.3	95.5 ± 0.2	81.1 ± 0.3	89.6 ± 0.3	70.0 ± 0.6
DMP(100M)⇒GNN	75.2 ± 0.6	77.5 ± 0.6	94.7 ± 0.4	80.3 ± 0.5	86.3 ± 1.0	69.2 ± 0.5

Table 1: Test ROC-AUC (%) performance of different methods on 6 binary classification tasks from MoleculeNet benchmark. The training, validation and test sets are provided by DeepChem in advance. The first parts are cited from Wang et al. [53], and the other parts are run with three different seeds.

	Classification (Higher is better)			Regression (Lower is better)		
Dataset Metric	BBBP ROC-AUC	SIDER ROC-AUC	ClinTox ROC-AUC	ESOL RMSE	QM7 MAE	QM8 MAE
GROVER	$0.940_{(0.019)}$	$0.658_{(0.023)}$	$0.944_{(0.021)}$	$0.831_{(0.025)}$	$72.6_{(3.8)}$	$0.0125_{(0.002)}$
MPG	$0.922_{(0.012)}$	$0.661_{(0.007)}$	$0.963_{(0.028)}$	$0.741_{(0.017)}$	–	–
DMP⇒TF	$0.945_{(0.020)}$	$0.695_{(0.011)}$	$0.968_{(0.007)}$	$0.700_{(0.084)}$	$69.6_{(8.3)}$	$0.0124_{(0.002)}$

Table 2: The performance comparison on molecular property prediction. The raw data are provided by Rong et al. [46] and splitted by scaffold splitter with 3 different seeds.

from the Transformer branch and GNN branch as DMP⇒TF and DMP⇒GNN respectively. We have the following observations:

(1) Compared with the previous supervised methods, DMP⇒TF outperforms almost all previous baselines across different tasks, which leverage well-designed fingerprints or the specially designed GNNs. The results demonstrate the effectiveness of using pre-trained models.

(2) As shown in the second and third parts of Table 1, both MLM loss and dual-view consistency loss help, no matter for the Transformer branch or GNN branch. Take the BBBP task of Transformer branch as an example. With MLM only or dual-view loss only, the accuracy is 74.9 and 71.1 respectively. After using both of them, the accuracy becomes 78.1. On the other hand, if we apply dual-view loss only, we find that in general, the results are slightly worse than using MLM only (see the row "DMP w/o MLM⇒TF").

(3) If we apply our method to two Transformer branches or two GNN branches (i.e., the "TF ($\times 2$)" and "GNN ($\times 2$)" in Table 1), we can see the results is not good as our proposed DMP, although we observe some improvement over using MLM only or dual-view consistency loss only. This is consistent with our discovery in Figure 1, where the Transformer and GNN have complementary views towards processing molecules.

(4) We empirically found that tuning on the Transformer branch is better than tuning on the GNN branch. Therefore, by default, we recommend using the Transformer branch. However, if we do not count for the number of parameters and use both branches for inference, the performances can be further improved (see the row "DMP⇒TF+GNN"), but not too much.

(5) Compared with other pre-training methods, Hu et al. [23] and MolCLR, which have more sophisticated models or training strategy, DMP⇒ TF achieves the best performance on 5 out of 6 binary classification tasks. Also, compared with two recent models, GROVER and MPG, DMP⇒ TF also outperforms them on classification and regression tasks (see Table 2). This shows that the joint pre-training of different views brings impressive benefit for molecular property prediction.

(6) Finally, we also pre-train on $100M$ compounds and then finetune on the downstream tasks. Compared to the results obtained by pre-training on $10M$ data, we observe improvement for both Transformer branch and GNN branch. We will explore how to effectively use more data in the future.

4.3 Experiments on retrosynthesis

In addition to molecule classification, we also conduct experiments on molecule generation. Specifically, we choose the retrosynthesis task: Given a target molecule (i.e., product) which cannot be directly obtained, we want to identify several easily obtained molecules (i.e., reactants) that can synthesize the product.

Dataset: Following [6, 48, 56], we conduct experiments on two widely used datasets, USPTO-50K [33, 4] and USPTO-full [6, 56]. USPTO-50K consists of 50K reactions with 10 reaction types in total, and USPTO-full consists of 950K cleaned reactions from the USPTO 1976-2016 without reaction types. We use the data released by [6], where the training, validation and test has been split in advance and each part contains 80%, 10% and 10% of the total data respectively. For USPTO-50K, we work on two settings where the reaction type is given or not.

Network Architecture: We mainly use the Transformer branch for finetuning. One reason is that Transformer-based models achieved promising improvements [49] in retrosynthesis, and the other is that the graph-based methods either contain no open-sourced code [48], or hard to be combined with pre-training since the GNN module is used to model the templates extracted by specific rules [6].

We implement three models for comparison: (i) the standard Transformer. Both the encoder and decoder have 6 layers, with 4 attention heads, embedding dimension 512 and feed-forward dimension 1024; (ii) We use the pre-trained model to initialize the encoder while the decoder remains the same as (i); (iii) Following [58], we fuse the pre-trained branch with Transformer, where each layer in the encoder and decoder attends to the output of pre-trained model in an attentive way.

Evaluation Metrics: We evaluate the models by the top-k exact match accuracy (briefly, top-k accuracy), which verifies that given a product, whether one of the k generated reactant sets exactly matches the ground truth reactant set, $k \in \{1, 3, 5, 10, 20, 50\}$. For all $k > 1$, we use beam search to generate the reactant sets, and rank them by log likelihoods.

Methods	Top-k accuracy (%)					
	1	3	5	10	20	50
Reaction types unknown on USPTO-50K						
Transformer	42.3	61.9	67.5	72.9	75.5	77.1
Pre-trained model as Encoder	39.6	55.3	59.1	63.2	66.0	68.6
ChemBERTa [3] fusion	43.9	62.2	68.0	73.1	75.4	77.0
DMP fusion	46.1	65.2	70.4	74.3	76.1	77.5
Reaction types give as prior on USPTO-50K						
Transformer	54.2	73.6	78.3	81.3	83.1	84.3
ChemBERTa fusion	56.4	74.7	78.9	81.8	83.3	84.5
DMP fusion	57.5	75.5	80.2	83.1	84.2	85.1
Retrosynthesis results on USPTO-full						
Transformer	42.9	58.0	62.4	66.8	69.8	72.5
DMP fusion	45.0	59.6	63.9	67.9	70.7	73.2

Table 3: Results of top-k exact match accuracy on retrosynthesis.

Results: The results are shown in Table 3. We first train the standard Transformer (denoted as "Transformer"), which achieves 42.3% the top-1 accuracy on the unknown type setting. After initializing the encoder with pre-trained model, we observe that the accuracy drops to 39.6%. This is consistent with the discovery in [58], where simply applying initialization to sequence generation might not lead to good results.

After using the method in [58] (marked as DMP fusion), we found that our model brings improvement to retrosynthesis. Specifically, on USPTO-50k dataset, we improve the top-1 accuracy from 42.3% to 46.1% when the reaction type is unknown, and from 54.2% to 57.5% when the type is given. On

average, DMP improves the standard Transformer by $2 \sim 3$ points w.r.t. top-1 accuracy across all settings. Compared to a previous pre-trained model ChemBERTa [3] which is also used following [58], our method outperforms it by 2.2 and 1.1 on the above two settings, which demonstrates the superiority of DMP again. On the largest dataset, USPTO-full, our method also boosts the Transformer baseline by 2.1 points, setting a state-of-the-art results on USPTO-full. The comparisons of our method with previous methods are shown in Appendix B.3 due to space limitation. We are aware that on USPTO-50K, some template-based methods [6] and well-designed GNNs [48] achieve better results. We leave the combination with them as future work.

4.4 Visualization of pre-trained representations

(a) GNN: DB Index 3.52 (b) Transformer: DB Index 3.16 (c) DMP: DB Index 1.64

Figure 3: Visualization of the representations learned by different models. Different colors indicate different scaffolds.

We visualize the pre-trained representations to verify whether they capture the scaffold information [24], which is used to represent the core structures of bioactive compounds. Intuitively, molecules with the same scaffold share similar architectures, and therefore are expected to be close in the high-level representation space. Following [32], we choose ten representative scaffolds (denoted as $\mathcal{S}$)[6] and then randomly select $200k$ compounds. For each compound whose scaffold lies in $\mathcal{S}$, we obtain its three representations: one from GNN pre-training with MLM only, one from Transformer pre-training with MLM only, and the third one from the Transformer branch of our DMP pre-training. We visualize the features through t-SNE [50] and the results are shown in Figure 3. We find that for the representations obtained by GNN pre-training and Transformer pre-training, the molecules in different scaffolds are overlapped and mixed together (e.g., the green and purple nodes for GNN; the green and light blue nodes for Transformer). In contrast, for the representations obtained by DMP pre-training, the molecules in different scaffolds are well separated, which demonstrates that our DMP better captures the scaffold information. Quantitatively, in terms of the DB index [8] (the smaller, the better), which is a metric to evaluate the clustering results, DMP clearly outperforms GNN pre-training and Transformer pre-training.

5 Conclusions and future work

In this work, we proposed DMP, a novel molecule pre-training method that leverages dual views of molecules. The core idea of DMP is to maximize the consistency between the two representations extracred from two views, in addition to predicting masked tokens. We achieve state-of-the-art results on seven molecular property prediction tasks from MoleculeNet and one retrosynthesis task on USPTO-full. For future work, first, we will combine with stronger GNN models, e.g. [46, 32]. Second, as discussed in Section 3.3, it is interesting to design a pre-training method that dynamically determines which view to use for a specific molecule instead of going through both views, so as to

[6] We show the ten scaffolds in Appendix A.3.

improve training efficiency. Third, how to combine our method with template-based retrosynthesis methods deserves attention. Finally, compressing the pre-trained model is another interesting topic.

References

[1] Avrim Blum and Tom Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory*, pp. 92–100, 1998.

[2] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pp. 1597–1607. PMLR, 2020.

[3] Seyone Chithrananda, Gabe Grand, and Bharath Ramsundar. Chemberta: Large-scale self-supervised pretraining for molecular property prediction. *arXiv preprint arXiv:2010.09885*, 2020.

[4] Connor W Coley, Luke Rogers, William H Green, and Klavs F Jensen. Computer-assisted retrosynthesis based on molecular similarity. *ACS central science*, 3(12):1237–1245, 2017.

[5] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[6] Hanjun Dai, Chengtao Li, Connor W Coley, Bo Dai, and Le Song. Retrosynthesis prediction with conditional graph logic network. In *NEURIPS*, 2019.

[7] Laurianne David, Amol Thakkar, Rocío Mercado, and Ola Engkvist. Molecular representations in ai-driven drug discovery: a review and practical guide. *Journal of Cheminformatics*, 12(1):1–22, 2020.

[8] David L Davies and Donald W Bouldin. A cluster separation measure. *IEEE transactions on pattern analysis and machine intelligence*, (2):224–227, 1979.

[9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[10] Linhao Dong, Shuang Xu, and Bo Xu. Speech-transformer: A no-recurrence sequence-to-sequence model for speech recognition. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5884–5888, 2018. doi: 10.1109/ICASSP.2018.8462506.

[11] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=YicbFdNTTy`.

[12] Joseph L Durant, Burton A Leland, Douglas R Henry, and James G Nourse. Reoptimization of mdl keys for use in drug discovery. *Journal of chemical information and computer sciences*, 42(6):1273–1280, 2002.

[13] David Duvenaud, Dougal Maclaurin, Jorge Aguilera-Iparraguirre, Rafael Gómez-Bombarelli, Timothy Hirzel, Alán Aspuru-Guzik, and Ryan P. Adams. Convolutional networks on graphs for learning molecular fingerprints. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'15, pp. 2224–2232, Cambridge, MA, USA, 2015. MIT Press.

[14] Maha Elbayad, Jiatao Gu, Edouard Grave, and Michael Auli. Depth-adaptive transformer. *arXiv preprint arXiv:1910.10073*, 2019.

[15] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, Bilal Piot, koray kavukcuoglu, Remi Munos, and Michal Valko. Bootstrap your own latent - a new approach to self-supervised learning. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 21271–21284. Curran Associates, Inc., 2020. URL `https://proceedings.neurips.cc/paper/2020/file/f3ada80d5c4ee70142b17b8192b2958e-Paper.pdf`.

[16] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent: A new approach to self-supervised learning. *arXiv preprint arXiv:2006.07733*, 2020.

[17] R. Hadsell, S. Chopra, and Y. LeCun. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pp. 1735–1742, 2006. doi: 10.1109/CVPR.2006.100.

[18] Jiazhen He, Felix Mattsson, Marcus Forsberg, Esben Jannik Bjerrum, Ola Engkvist, Christian Tyrchan, Werngard Czechtizky, et al. Transformer neural network for structure constrained molecular optimization. 2021.

[19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[20] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9729–9738, 2020.

[21] Tin Kam Ho. Random decision forests. In *Proceedings of 3rd international conference on document analysis and recognition*, volume 1, pp. 278–282. IEEE, 1995.

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

[23] Weihua Hu, Bowen Liu, Joseph Gomes, Marinka Zitnik, Percy Liang, Vijay Pande, and Jure Leskovec. Strategies for pre-training graph neural networks. In *International Conference on Learning Representations*, 2020. URL `https://openreview.net/forum?id=HJlWWJSFDH`.

[24] Ye Hu, Dagmar Stumpfe, and Jürgen Bajorath. Computational exploration of molecular scaffolds in medicinal chemistry. *Journal of Medicinal Chemistry*, 59(9):4062–4076, 2016. doi: 10.1021/acs.jmedchem.5b01746. URL `https://doi.org/10.1021/acs.jmedchem.5b01746`. PMID: 26840095.

[25] Bing Huang and O Anatole Von Lilienfeld. Communication: Understanding molecular representations in machine learning: The role of uniqueness and target similarity, 2016.

[26] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pp. 448–456. PMLR, 2015.

[27] John J. Irwin and Brian K. Shoichet. Zinc - a free database of commercially available compounds for virtual screening. *Journal of Chemical Information and Modeling*, 45(1):177–182, 2005.

[28] Sunghwan Kim, Jie Chen, Tiejun Cheng, Asta Gindulyte, Jia He, Siqian He, Qingliang Li, Benjamin A Shoemaker, Paul A Thiessen, Bo Yu, Leonid Zaslavsky, Jian Zhang, and Evan E Bolton. PubChem in 2021: new data content and improved web interfaces. *Nucleic Acids Research*, 49(D1):D1388–D1395, 11 2020. ISSN 0305-1048. doi: 10.1093/nar/gkaa971. URL `https://doi.org/10.1093/nar/gkaa971`.

[29] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[30] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[31] Guohao Li, Chenxin Xiong, Ali Thabet, and Bernard Ghanem. Deepergcn: All you need to train deeper gcns. *arXiv preprint arXiv:2006.07739*, 2020.

[32] Pengyong Li, Jun Wang, Yixuan Qiao, Hao Chen, Yihuan Yu, Xiaojun Yao, Peng Gao, Guotong Xie, and Sen Song. Learn molecular representations from large-scale unlabeled molecules for drug discovery. *arXiv preprint arXiv:2012.11175*, 2020.

[33] Bowen Liu, Bharath Ramsundar, Prasad Kawthekar, Jade Shi, Joseph Gomes, Quang Luu Nguyen, Stephen Ho, Jack Sloane, Paul Wender, and Vijay Pande. Retrosynthetic reaction prediction using neural sequence-to-sequence models. *ACS central science*, 3(10):1103–1113, 2017.

[34] Shengchao Liu, Mehmet F Demirel, and Yingyu Liang. N-gram graph: Simple unsupervised representation for graphs, with applications to molecules. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL `https://proceedings.neurips.cc/paper/2019/file/2f3926f0a9613f3c3cc21d52a3cdb4d9-Paper.pdf`.

[35] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[36] Chengqiang Lu, Qi Liu, Chao Wang, Zhenya Huang, Peize Lin, and Lixin He. Molecular property prediction: A multilevel quantum interactions modeling perspective. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 1052–1060, 2019.

[37] Łukasz Maziarka, Tomasz Danel, Sławomir Mucha, Krzysztof Rataj, Jacek Tabor, and Stanisław Jastrzębski. Molecule attention transformer. *arXiv preprint arXiv:2002.08264*, 2020.

[38] Janosch Menke and Oliver Koch. Using domain-specific fingerprints generated through neural networks to enhance ligand-based virtual screening. *Journal of Chemical Information and Modeling*, 61(2):664–675, 2021. doi: 10.1021/acs.jcim.0c01208. URL https://doi.org/10.1021/acs.jcim.0c01208. PMID: 33497572.

[39] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[40] Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran. Image transformer. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 4055–4064. PMLR, 10–15 Jul 2018. URL http://proceedings.mlr.press/v80/parmar18a.html.

[41] Giorgio Pesciullesi, Philippe Schwaller, Teodoro Laino, and Jean-Louis Reymond. Transfer learning enables the molecular transformer to predict regio-and stereoselective reactions on carbohydrates. *Nature communications*, 11(1):1–8, 2020.

[42] Roshan Rao, Nicholas Bhattacharya, Neil Thomas, Yan Duan, Xi Chen, John Canny, Pieter Abbeel, and Yun S Song. Evaluating protein transfer learning with tape. *Advances in Neural Information Processing Systems*, 32:9689, 2019.

[43] Yi Ren, Yangjun Ruan, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu. Fastspeech: Fast, robust and controllable text to speech. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32, 2019. URL https://proceedings.neurips.cc/paper/2019/file/f63f65b503e22cb970527f23c9ad7db1-Paper.pdf.

[44] Alexander Rives, Joshua Meier, Tom Sercu, Siddharth Goyal, Zeming Lin, Jason Liu, Demi Guo, Myle Ott, C. Lawrence Zitnick, Jerry Ma, and Rob Fergus. Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. *Proceedings of the National Academy of Sciences*, 118(15), 2021. ISSN 0027-8424. doi: 10.1073/pnas.2016239118. URL https://www.pnas.org/content/118/15/e2016239118.

[45] David Rogers and Mathew Hahn. Extended-connectivity fingerprints. *Journal of Chemical Information and Modeling*, 50(5):742–754, 2010. doi: 10.1021/ci100050t. URL https://doi.org/10.1021/ci100050t. PMID: 20426451.

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

[47] Philippe Schwaller, Teodoro Laino, Théophile Gaudin, Peter Bolgar, Christopher A Hunter, Costas Bekas, and Alpha A Lee. Molecular transformer: a model for uncertainty-calibrated chemical reaction prediction. *ACS central science*, 5(9):1572–1583, 2019.

[48] Chence Shi, Minkai Xu, Hongyu Guo, Ming Zhang, and Jian Tang. A graph to graphs framework for retrosynthesis prediction. In *International Conference on Machine Learning*, pp. 8818–8827. PMLR, 2020.

[49] Igor V Tetko, Pavel Karpov, Ruud Van Deursen, and Guillaume Godin. State-of-the-art augmented nlp transformer models for direct and single-step retrosynthesis. *Nature communications*, 11(1):1–11, 2020.

[50] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

[51] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *arXiv preprint arXiv:1706.03762*, 2017.

[52] Sheng Wang, Yuzhi Guo, Yuhong Wang, Hongmao Sun, and Junzhou Huang. 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*, pp. 429–436, 2019.

[53] Yuyang Wang, Jianren Wang, Zhonglin Cao, and Amir Barati Farimani. Molclr: Molecular contrastive learning of representations via graph neural networks. *arXiv preprint arXiv:2102.10056*, 2021.

[54] David Weininger. Smiles, a chemical language and information system. 1. introduction to methodology and encoding rules. *Journal of chemical information and computer sciences*, 28(1):31–36, 1988.

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

[56] Chaochao Yan, Qianggang Ding, Peilin Zhao, Shuangjia Zheng, JINYU YANG, Yang Yu, and Junzhou Huang. Retroxpert: Decompose retrosynthesis prediction like a chemist. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 11248–11258. Curran Associates, Inc., 2020. URL `https://proceedings.neurips.cc/paper/2020/file/819f46e52c25763a55cc642422644317-Paper.pdf`.

[57] Kevin Yang, Kyle Swanson, Wengong Jin, Connor Coley, Philipp Eiden, Hua Gao, Angel Guzman-Perez, Timothy Hopper, Brian Kelley, Miriam Mathea, et al. Analyzing learned molecular representations for property prediction. *Journal of chemical information and modeling*, 59(8):3370–3388, 2019.

[58] Jinhua Zhu, Yingce Xia, Lijun Wu, Di He, Tao Qin, Wengang Zhou, Houqiang Li, and Tie-Yan Liu. Incorporating bert into neural machine translation. *arXiv preprint arXiv:2002.06823*, 2020.

[59] Jinhua Zhu, Lijun Wu, Yingce Xia, Shufang Xie, Tao Qin, Wengang Zhou, Houqiang Li, and Tie-Yan Liu. {IOT}: Instance-wise layer reordering for transformer structures. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=ipUPfYxWZvM`.

A Experiment Setup

A.1 Detailed configuration of the GNN branch

The GNN branch in DMP is a variant of DeeperGCN [31], which is stacked by 12 identical blocks. Each block consists of a batch normalization layer, a nonlinear layer and a graph convolutional layer sequentially, with a skip connection connected the input and the output. In each graph convolutional layer, each node will fuse its neighbor edge representations and its neighbor node representations with an aggregation layer. Specifically, in DMP, the aggregation layer is a concatenation of maximize, minimize and average pooling.

A.2 Finetuning hyperparameters

We summarize the finetuning hyperparameters in Table 4.

Hyperparam	**Classification**	**Regression**
Learning rate	$\{5e-5, 1e-4, 2e-4\}$	$\{5e-5, 1e-4, 2e-4\}$
Batch Size	$\{8, 16, 32\}$	$\{8, 16, 32\}$
Weight Decay	$\{0.1, 0.01\}$	$\{0.1, 0.01\}$
Max Epochs	10	10
Learning Rate Decay	Linear	Linear
Warmup ratio	0.06	0.06
Dropout	0.1	$\{0.1, 0.2, 0.3\}$

Table 4: Hyperparameters for finetuning DMP.

A.3 Scaffolds

In Figure 4, we present the scaffolds used in Figure 3 of the main paper.

B More experimental results

B.1 Comparison with ChemBERTa [3]

We compare our method with ChemBERTa [3] on two classification datasets, BBBP and HIV, which is also used in [3]. The results are summarized in Table 5, where TF (MLM) means that we pre-train a Transformer model using masked language modeling only, and DMP⇒TF means that we pre-train a

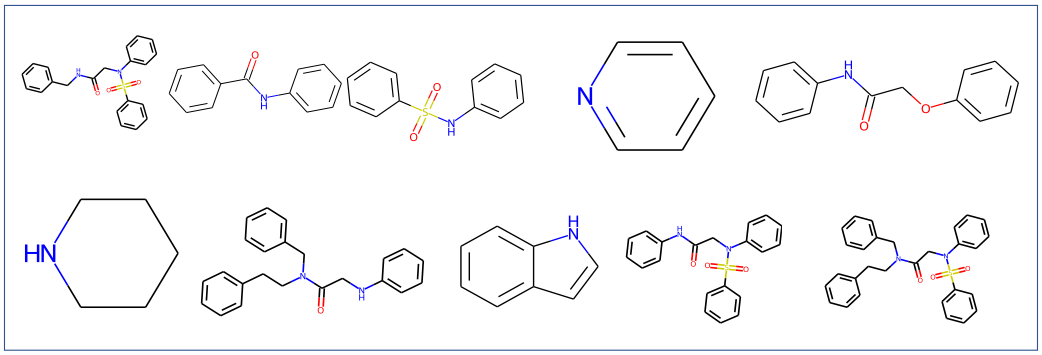

Figure 4: Ten scaffolds used in our visualization.

	BBBP	HIV
ChemBERTa	0.643	0.622
TF (MLM)	0.749	0.802
DMP⇒ TF	0.781	0.810

Table 5: Comparison with ChemBERTa.

Transformer using DMP. TF (MLM) is an enhanced implementation of ChemBERTa, where we train models with more GPUs and larger batch size, and eventually obtain better results. We can see that our proposed method outperforms both ChemBERTa and TF (MLM), which shows the effectiveness of our method.

B.2 Comparison with contrastive learning

Considering dual-view loss is related to contrastive loss, we also explore a corresponding variant: given a molecule m, the corresponding features from Transformer branch and GNN branch are denoted as $f_s(m)$ and $f_g(m)$ respectively. Given several other molecules $m_1, m_2, \cdots, m_n$, $f_g(m)$ should be similar to $f_s(m)$ while dissimilar to $f_s(m_1), \cdots, f_s(m_n)$, and $f_s(m)$ should be dissimilar to $f_g(m_1), \cdots, f_g(m_n)$. Such a variant achieves 77.1% and 94.5% ROC-AUC on BBBP and ClinTox dataset, respectively, and performs a little worse than DMP (78.1% and 95.0%).

B.3 Retrosynthesis

We compare the results on retrosysthesis of existed works and our method in Table 6. The results of Retrosim and neuralsym are from [6]. On the largest dataset, USPTO-full, our method achieves the best result. On USPTO-50K, the template-based method GLN [6] is better than ours, and we will combine with GLN in the future.

Methods	Top-k accuracy (%)					
	1	3	5	10	20	50
Reaction types unknown on USPTO-50K						
Transformer	42.3	61.9	67.5	72.9	75.5	77.1
Pre-trained model as Encoder	39.6	55.3	59.1	63.2	66.0	68.6
ChemBERTa fusion [3]	43.9	62.2	68.0	73.1	75.4	77.0
DMP fusion	46.1	65.2	70.4	74.3	76.1	77.5
RetroSim	37.3	54.7	63.3	74.1	82.0	85.3
NeuralSym	44.4	65.3	72.4	78.9	82.2	83.1
GLN [6]	52.6	68.0	75.1	83.1	88.5	92.1
Reaction types give as prior on USPTO-50K						
Transformer	54.2	73.6	78.3	81.3	83.1	84.3
ChemBERTa fusion [3]	56.4	74.7	78.9	81.8	83.3	84.5
DMP fusion	57.5	75.5	80.2	83.1	84.2	85.1
RetroSim	52.9	73.8	81.2	88.1	91.8	92.9
NeuralSym	55.3	76.0	81.4	85.1	86.5	86.9
GLN [6]	63.2	77.5	83.4	89.1	92.1	93.2
Retrosynthesis results on USPTO-full						
Transformer	42.9	58.0	62.4	66.8	69.8	72.5
DMP fusion	45.0	59.6	63.9	67.9	70.7	73.2
Retrosim	32.8	–	–	56.1		
neuralsym	35.8	–	–	60.8		
GLN [6]	39.3	–	–	63.7		

Table 6: Results of top-k exact match accuracy on Retrosynthesis.