Table A: Comparison between different methods. The metric is AUC-ROC. The numbers in brackets are the standard deviation.

	Hu. et al.		GROVER-GIN		GROVER-MPNN	
	w pre-train	w/o pre-train	w pre-train	w/o pre-train	w pre-train	w/o pre-train
BBBP	$0.915_{(0.040)}$	$0.899_{(0.035)}$	$0.925_{(0.036)}$	$0.901_{(0.051)}$	$0.929_{(0.029)}$	$0.917_{(0.027)}$
SIDER	$0.614_{(0.006)}$	$0.615_{(0.007)}$	$0.648_{(0.015)}$	$0.627_{(0.016)}$	$0.650_{(0.003)}$	$0.637_{(0.030)}$
BACE	$0.851_{(0.027)}$	$0.837_{(0.028)}$	$0.862_{(0.020)}$	$0.823_{(0.050)}$	$0.872_{(0.031)}$	$0.852_{(0.034)}$
Average	0.793	0.784	0.812	0.784	0.817	0.802

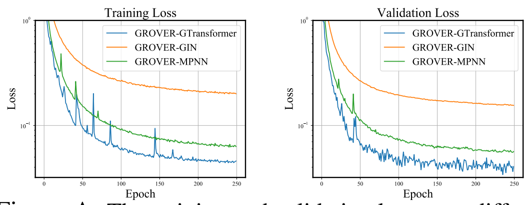

Figure A: The training and validation losses on different architectures.

We sincerely thank all reviewers for their valuable comments. Below are our point-by-point responses.

To Reviewer #1: Thank you for your detailed and valuable suggestions in terms of our writing. In our final version, we will carefully rephrase the introduction of GTransformer (*e.g.* explaining more on high-level concepts, fixing the comparisons with GAT, and rephrasing the outline of the architecture), rewrite and polish the sentences in Section 3 and 4, and remove all unnecessary adjectives to make our paper more direct and concise. We believe such revisions are achievable and do not change the main story of our paper. We address the major concerns as follows. **Q1. The metric in Table 1.** We use ROC-AUC and follow the benchmarking n-fold scaffold setting for train/test split, while Hu et al. report ROC-AUC (%) under a different split strategy. Thus, it's reasonable to produce different numbers for the same method. We will specify this in the caption of Table 1. **Q2. More ablations.** 1. We implement GIN and MPNN based on our framework, and per-train them under the same setting as in Section 5.2. For a fair comparison, we set a large hidden size to ensure them to be nearly as big as GROVER (38M parameters in Section 5.2). As shown in Fig. A, our model still outperforms GIN and MPNN in both training and validation, which again verifies the effectiveness of our GTransformer. 2. We report the results by Hu et al. with and without pre-training in Tab. A. As a comparison, we also involve the performance of GROVER with the backbone GIN. We find that without pre-training, our GROVER-GIN is consistent with Hu et al. on average, thus verifying the reliability of our implementations. However, after pre-training, GROVER-GIN achieves nearly 2% higher number than Hu et al., which supports the advantage of our proposed self-supervised loss. **Q3. On the node and edge GTransformers.** Sorry for the confusion. There is no state exchanging or parameter sharing between these two GTransformers. We apply the disagreement loss on the predictions of the two views to retain the consensus (See L46 in the supplement). We will clarify this point in the final version. **Q5. Other details.** We use SUM aggregation for dyMPN, aggregate2node, and aggregate2edge. In L323, $K_l = 6$ is the mean of the sample distribution in dyMPN (See L112 in the supplement). Fig. 1 depicts only one GTransformer layer. The prior work (Ingraham et al) will be added. We will revise our paper to reflect the above details in the final version.

To Reviewer #2: **Q1. Novelty of contribution.** We would like to highlight that our method clearly differs from the paper by Hu et al. in three aspects: 1. Hu et al. mask only nodes/edges to predict their types, while our subgraph masking task predicts the types of nodes, edges and local structures, which enables to encode more information. 2. Regarding graph-level pre-training, our method is completely unsupervised, whereas Hu et al. implement the full-supervised strategy in the experiments, which would limit the practical usage. 3. The development of GTransformer is novel and crucial. Considering the extra ablations in Fig. A, Our GTransformer achieves better performance than GIN and MPNN provided the same pre-training scenario. Overall, we believe our contributions are novel and remarkable. **Q2. Does pre-training only work for dyMPN?** Not really, and it works for general GNNs. To show this, we have conducted MPNN and GIN with and without pre-training in Tab. A following the same protocol in Section 5.2. The results read that the pre-training stage is also crucial for these two models. **Q3. Justification on alleviating over-smoothing.** The claim that long-range residual connections will alleviate over-smoothing is directly borrowed from JKNet [Xu et al. ICML-18] which will be cited. **Q4. Justification of architectural choices.** Indeed, GROVER-w/o-GTrans represents the traditional transformer with multiple short-range connections. In Section 5.2, we have conducted a comparison between GROVER and GROVER-w/o-GTrans, which can be regarded as a long-range residual connection versus multiple short-range connections. We will further specify this in the final version. **Q5. Other issues.** For the pre-training task, we use MEAN readout function to generate the graph-level representation. We will release the code as well as the pre-train models to facilitate the follow-up researches.

To Reviewer #3: **Q1. Baselines and hyper-parameters.** For the baseline methods, we first follow the suggested hyper-parameters given by the original implementation, and if the results cannot be reproduced, we will conduct hyper-parameter tuning by validation. We will clarify this and provide the parameter-size of each baseline in Table 1 of the final version. Most baselines have smaller architecture than ours, and even we simply enlarge their model sizes, the performances are still worse than our method (see our response to Q2, R#1 and the extra ablations in Fig. A). **Q2. Other issues.** In the final version, we will report the average molecule size for each dataset, move the computational cost of the pre-training models from supplementary to the paper, cite and discuss the raised references by the reviewer.

To Reviewer #4: Thanks for the comments. In detail (as also supported by R1), GAT restricts the attention to each existing edge, while in our model, the transformer-style layer acts as a dense attention over all nodes to capture the graph-level information. We are willing to discuss the difference between our method and GAT in the final version.