LITEGEM: LITE GEOMETRY ENHANCED MOLECULAR REPRESENTATION LEARNING FOR QUANTUM PROPERTY PREDICTION

SOLUTION TO KDD CUP 2021 PCQM4M-LSC CHALLENGE

Shanzhuo Zhang[*], **Lihang Liu**[*]
Sheng Gao[*], **Donglong He**[*], **Xiaomin Fang**
PaddleHelix Team, Baidu Inc.
{zhangshanzhuo, liulihang}@baidu.com
{gaosheng06, hedonglong}@baidu.com
fangxiaomin01@baidu.com

Weibin Li[*], **Zhengjie Huang**[*]
Weiyue Su, Wenjin Wang
PGL Team, Baidu Inc.
{liweibin02, huangzhengjie}@baidu.com
{suweiyue, wangwenjin02}@baidu.com

ABSTRACT

In this report, we (SuperHelix team) present our solution to KDD Cup 2021-PCQM4M-LSC, a large-scale quantum chemistry dataset on predicting HOMO-LUMO gap of molecules. Our solution, **Lite G**eometry **E**nhanced **M**olecular representation learning (LiteGEM) achieves a mean absolute error (MAE) of 0.1204 on the test set with the help of deep graph neural networks and various self-supervised learning tasks. The code of the framework can be found in https://github.com/PaddlePaddle/PaddleHelix/tree/dev/competition/kddcup2021-PCQM4M-LSC/.

1 Introduction

Molecular property prediction has been widely considered as one of the most critical tasks in computational drug and materials discovery since many methods rely on predicted molecular properties to evaluate, select and generate molecules. With the development of deep neural networks (DNNs), molecular representation learning exhibits a great advantage over feature engineering-based methods, which has attracted increasing research attention to tackling the molecular property prediction problem.

Inspired by our previous work (Fang et al. [2021]), we propose using **Lite G**eometry **E**nhanced **M**olecular representation learning (LiteGEM) for the quantum property prediction: HOMO-LUMO gap (Nakata and Shimazaki [2017]). We add the word "Lite" due to the fact that our original GEM model requires 3D geometry information as the input features, which is absent in this dataset. However, the self-supervised learning strategies proposed in GEM can still be incorporated into LiteGEM and boost the performance.

The report is organized as follows. We briefly introduce graph neural nets and message passing in Section 2. In Section 3, we present the main architecture of our model LiteGEM and its components. Experiment details such as choices of hyper-parameters and performance of ensemble can be found in Section 4.

2 Preliminaries

A molecule consists of atoms, and the neighboring atoms are connected by the chemical bonds, which can be naturally represented by a graph $G = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is a node set and $\mathcal{E}$ is an edge set. An atom in the molecule is regarded as a node $v \in \mathcal{V}$ and a chemical bond in the molecule is regarded as an edge $(u, v) \in \mathcal{E}$ connecting atoms u and v.

Graph neural networks (GNNs) can be seen as message passing neural networks (Gilmer et al. [2017a]), which are useful for predicting molecular properties. Following the definitions of the previous GNNs (Xu et al. [2019]), the features of a node v are represented by $\mathbf{x}_v$ and the features of an edge (u, v) are represented by $\mathbf{e}_{uv}$. Taking node

[*]Equal contribution.

features, edge features and the graph structure as inputs, a GNN learns the representation vectors of the nodes and the entire graph, where the representation vector of a node v is denoted by $\mathbf{h}_v$ and the representation vector of the entire graph is denoted by $\mathbf{h}_G$. A GNN iteratively updates a node's representation vector by aggregating the messages from the node's neighbors. Given a node v, its representation vector $\mathbf{h}_v^{(k)}$ at the k-th iteration is formalized by

$$\begin{aligned}\mathbf{a}_v^{(k)} &= \textit{AGGREGATE}^{(k)}(\{(\mathbf{h}_v^{(k-1)}, \mathbf{h}_u^{(k-1)}, x_{uv} | u \in \mathcal{N}(v)\}),\\ \mathbf{h}_v^{(k)} &= \textit{COMBINE}^{(k)}(\mathbf{h}_v^{(k-1)}, \mathbf{a}_v^{(k)}).\end{aligned} \tag{1}$$

where $\mathcal{N}(v)$ is the set of neighbors of node v, $\textit{AGGREGATE}^{(k)}$ is the aggregation function for aggregating messages from a node's neighborhood, and $\textit{COMBINE}^{(k)}$ is the update function for updating the node representation. We initialize $\mathbf{h}_v^{(0)}$ by the feature vector of node v, i.e., $\mathbf{h}_v^{(0)} = \mathbf{x}_v$.

READOUT function is introduced to integrate the nodes' representation vectors at the final iteration to gain the graph's representation vector h_G, which is formalized as

$$\mathbf{h}_G = \textit{READOUT}(\mathbf{h}_v^{(K)} | v \in \mathcal{V}), \tag{2}$$

where K is the number of iterations. In most cases, *READOUT* is a permutation invariant pooling function, such as summation and maximization. The graph's representation vector $\mathbf{h}_G$ can then be used for downstream task predictions.

3 The LiteGEM Framework

We introduce our solution in this section, including the GNN architectures, self-supervised learning strategies, and feature engineering tricks.

3.1 LiteGEMConv

Unlike Convolutional Neural Networks (CNNs), which are able to take advantage of stacking very deep layers, Graph Convolutional Networks (GCNs) suffer from vanishing gradients, over-smoothing, and over-fitting issues when going deeper. To encode the whole molecular structures, motivated by DeeperGCN from Li et al. [2020], we propose a message-passing strategy for graph convolution: LiteGEMConv, which is formalized as

$$\mathbf{m}_{vu}^{(k)} = \textit{MLP}\left(\mathbf{h}_v^{(k-1)} \| \mathbf{h}_u^{(k-1)} \| \mathbf{e}_{vu}\right), \tag{3}$$

$$\mathbf{a}_v^{(k)} = \textit{SoftMax\_Agg}_\beta\left(\left\{\mathbf{m}_{vu}^{(k)} | u \in \mathcal{N}(v)\right\}\right), \tag{4}$$

$$\mathbf{h}_v^{(k)} = \textit{Linear}\left(\mathbf{h}_v^{(k-1)} + \mathbf{a}_v^{(k)}\right), \tag{5}$$

where $\|$ denotes concatenation of vectors, *MLP* denotes a 2-layer Multi Layer Perceptron (MLP) with SiLU as the activation function and *Linear* denotes the Linear layer. $\textit{SoftMax\_Agg}_\beta$ (Li et al. [2020]) function is used as our aggregation, which is defined as:

$$\textit{SoftMax\_Agg}_\beta(\cdot) = \sum_{u \in \mathcal{N}(v)} \frac{\exp\left(\beta \mathbf{m}_{vu}\right)}{\sum_{i \in \mathcal{N}(v)} \exp\left(\beta \mathbf{m}_{vi}\right)}, \tag{6}$$

where $\mathbf{m}_{vu} \in \mathbb{R}^D$ is the given message set $\{\mathbf{m}_{vu} | u \in \mathcal{N}(v)\}$ and β is the temperature controlling the smoothness of the distribution. Note that we also try to replace $\textit{SoftMax\_Agg}_\beta$ with a simple summation aggregation and observe no degradation in the performance, but we still keep it.

Overall, the LiteGEM consists of 11 LiteGEMConv layers with virtual node representations (Gilmer et al. [2017b]) added at each layer. Node features $\mathbf{h}_v$ initialized as $\mathbf{x}_v$ are updated by LiteGEMConv layer-by-layer, and the output is denoted as $\mathbf{h}_v^{(K)}$. The graph level representation $\mathbf{h}_G$ is obtained via mean pooling over the node representations of all nodes and the final output as the prediction of HOMO-LUMO gap is produced by another MLP as:

$$y_G = \text{MLP}_G(\mathbf{h}_G).$$

Here MLP_G is a 2- or 3-layer MLP with dropout, batch normalization and SiLU activation.

3.2 Auxiliary and Pre-training Tasks

Self-supervised learning methods have proven their effectiveness in computer vision and natural language tasks when the number of labeled data is insufficient. To further enhance our model performance, we adopt both auxiliary tasks and pre-training tasks in a self-supervised fashion for graph neural networks. We follow our previous work (Fang et al. [2021]) and apply topology-level and geometry-level learning tasks as auxiliary and pre-training tasks.

3.2.1 Geometry-level: Bond Length & Bond Angle Prediction

According to the Hohenberg–Kohn theorems in DFT, the ground-state electron density determines all ground-state properties of a molecule, i.e., the ground-state conformation determines the HOMO-LUMO gap of a molecule. On the other hand, as proved in our previous work ChemRL-GEM (Fang et al. [2021]), by incorporating accurate 3D conformation in the QM9 dataset, we can achieve significant improvement in predicting various quantum properties of molecules. However, unlike ChemRL-GEM, due to the inference time limitation of this task, it is nearly impossible to generate accurate 3D conformation using DFT during inference.

To solve this dilemma, we adopt geometry-level self-supervised learning tasks of GEM only in the training procedure. More concretely, we utilize the bond length and bond angle prediction task of GEM as both the pre-training and auxiliary task for training LiteGEM. The bond lengths and the bond angles are the most important molecular geometrical parameters. The bond length is the distance between two joint atoms in a molecule, reflecting the bond strength between the atoms, while the bond angle is the angle connecting two consecutive bonds, including three atoms, describing the local spatial structure of a molecule.

3.2.2 Topology-level: Context Prediction

In context prediction, we use GNN to predict graph sub-structures (Mikolov et al. [2013], Rubenstein and Goodenough [1965]) using node-level representations. The intuition is that we want to enhance the ability of GNN to encode the neighborhood graph structure for each node. In order to represent the neighborhood of node u in a compact way, we design rules to map the neighborhood into a *context string*. Such string then contains all nodes and edges information that are at most 2 hops away from u in the graph. Specifically, the context string contains the following sub-strings:

- *str*(u);
- *str*(*sort*$(\mathcal{N}(u))$): the 1-hop neighboring nodes of u;
- *str*(*sort*$(\{e_{uv} : v \in \mathcal{N}(u)\})$): the 1-hop neighboring edges of u;
- *str*(*sort*$(\{w : w \in \mathcal{N}(v), v \in \mathcal{N}(u)\})$): the 2-hop neighboring nodes of u;
- *str*(*sort*$(\{e_{vw} : w \in \mathcal{N}(v), v \in \mathcal{N}(u)\})$): the 2-hop neighboring edges of u;

where we use $\mathcal{N}(v)$ to denote the neighboring nodes of node u, the function *sort* is used to convert the set into a sorted array and the function *str* is used to convert an array into a compact string. The above 5 sub-strings can then can concatenated to construct the context string. Note that we use the atomic number as the information for the nodes and the bond type for the edges.

Upon obtaining the context string, we hash it into an integer between 0 and 5000 for each node. Then we construct a node-level multi-class classification task to predict these integers by adding a Multi-layer Perceptron Layer (MLP) after the node representations. The MLP we used is composed of 2 linear layers with Batch Normalization and SiLU activation. The only hidden layer has a size equal to half of the input dimension, and the output layer has a size of 5000. This fore-mentioned classification task is then used for both pre-training and auxiliary tasks.

3.3 Smoothing

To further improve the performance, inspired by APPNP algorithm (Klicpera et al. [2018]), we integrate a smoothing strategy after the last LiteGEMConv block. Our smoothing strategy utilizes the relationship between graph convolutional networks (GCN) and PageRankPage et al. [1999] to derive an improved propagation scheme based on personalized PageRank. It achieves linear computational complexity by approximating topic-sensitive PageRank via power iteration. The process is defined as:

$$\begin{aligned} \boldsymbol{Z}^{(0)} &= \mathbf{M}^{L}, \\ \boldsymbol{Z}^{(k+1)} &= (1-\alpha)\widehat{\widehat{\boldsymbol{A}}}\boldsymbol{Z}^{(k)} + \alpha \boldsymbol{M}^{L}. \end{aligned} \tag{7}$$

Table 1: Hyper-parameters

lr	[1e-4, 2e-4, 5e-4, 1e-3]
dropout	[0.2, 0.3]
activation	[SiLU, ReLU, GeLU, SoftPlus]
α	[0, 0.2, 0.5, 0.8]
t	[1, 0, -1]
batch size	256
embed dimension	1024
w_{aux}	[0, 0.1, 0.2]

Here, $\mathbf{M}^L \in \mathbb{R}^{N \times D}$, is the node representations $\mathbf{h}_v^{(K)}$ produced by the DeeperGCN. $\widehat{\widetilde{\boldsymbol{A}}}$ is the adjacency matrix with self-loops and normalization. Note that when $\alpha = 0$, the smoothing layer degenerates into an identity mapping.

3.4 Feature Selection

We extend the standard node and edge features provided by the OGB-PCQM4M dataset (Liu et al. [2021], Chen et al. [2019], Kearnes et al. [2016]). To be specific, we incorporate the *atom type, chirality, degree, formal charge, number of hydrogen atoms connected, radical electron, hybridization, ring size, van der Waals radius, the valence of out shell, partial charge* as atom features, and *aromatic, if in ring, bond type, bond stereo type and conjugated* as bond features.

4 Experiments

4.1 Implementation Details

LiteGEM is implemented in PaddlePaddle[2] and PGL[3]. The total number of parameters is about 74 million for a single model. Other hyper-parameters for training the model are listed in Table 1. The training time is about 100 minutes per epoch on an Nvidia Tesla V100. The model usually needs about 100 epochs to converge though we set the total number of epochs to 150. To fully make use of the validation set, we adopt a 2-fold cross-validation scheme. 90% of the original validation set is randomly split into 2 cross-validation sets, and the last 10% is left to verify strategies of the ensemble. We train 5 variants of LiteGEM and save 164 checkpoints for further model ensemble (see Table 2).

The whole training program is preceded by a pre-training stage, where the model is only trained with the self-supervised tasks for 10 epochs. After the pre-training, we combine the auxiliary tasks together with the HOMO-LUMO-gap task. To balance the trade-off between the auxiliary tasks and the HOMO-LUMO-gap task, we use a pre-chosen discount factor: the loss of auxiliary tasks is down-weighted by 1/2 at the 20th epoch and completely removed after the 30th epoch. The initial weight for auxiliary tasks (w_{aux}) is shown in Table 1.

The 3D molecular conformation used in the bond length and bond angle tasks is generated by DFT using PySCF (Sun et al. [2020]). We mimic the DFT process used for calculating the original dataset (Nakata and Shimazaki [2017]) with various simplification to reduce resource consumption and generate 3D molecular conformation for all the 3 million training SMILES. Due to the simplification, only 20% of DFT results are comparable with the original dataset and used for the auxiliary and pre-training tasks. The calculation for each molecule takes about 20 core minutes.

4.2 Performance of Ensemble

Due to the limit of inference time, we need to control the size of ensembles. To achieve this, we first drop more than half the model checkpoints by the coefficients obtained from a Huber Regressor. Specifically, checkpoints with an absolute coefficient less than 0.01 are deemed futile and dropped. All the 164 model checkpoints and the 73 checkpoints for the ensemble are listed in Table 2. Original models are regressed on the 2 cross-validation sets separately.

The predictions of remained checkpoints are then sorted for each molecule, and the largest and smallest 20% predictions are dropped. Remained predictions are regressed on another Huber Regressor. We use this regressor to produce the final prediction on the test set.

Without ensemble, the best single model achieved an MAE of $\sim$0.1235 on the last 10% validation set, and the ensemble further decreased the MAE to 0.1141. On the whole test set, the MAE of our final ensemble model is 0.1204.

[2]https://github.com/paddlepaddle/paddle

[3]https://github.com/PaddlePaddle/PGL

Table 2: Number of model checkpoints before and after ensemble

Auxiliary and Pre-training Tasks					
ContextPred	BondLength	BondAngle	Smoothing	# Ckpts	# Ensemble Ckpts
-	-	-	Yes	36	9
-	-	-	-	40	15
Yes	-	-	-	40	20
Yes	Yes	-	-	24	15
Yes	Yes	Yes	-	24	14
Sum				164	73

Table 3: Valid MAE of LiteGEM on 8-fold cross-validation sets

Fold	LiteGEM w/ ContextPred	Graphormer
0	0.0949	0.0970
1	0.0946	0.0971
2	0.0949	0.0970
3	0.0946	0.0965
4	0.0950	0.0968
5	0.0945	0.0967
6	0.0958	0.0965
7	0.0950	0.0969

4.3 Inference Time

Taking raw SMILES as input, the total inference time for all the 377,423 test molecules is about 10.5 hours on an Nvidia Tesla P40. To be specific, it takes 0.5 hour to generate graph features, 10 hours in inference by all the 73 ensemble models (8 minutes each). The time consumed by the Huber Regressor for the ensemble is negligible.

4.4 Performance on 8-fold Cross-validation Sets

Although not part of our submitted solution, we further test our model with only the context prediction as the auxiliary task on 8-fold cross-validation sets to make a comparison with the 1st-place winner solution proposed by Ying et al. [2021]. Our model achieves lower Valid MAE on all the 8 folds of data (see Table 3), demonstrating the superiority of our architecture.

5 Conclusions and Future Work

In this challenge, our model LiteGEM achieves **test MAE of 0.1204** on PCQM4M with the help of deep graph neural networks and self-supervised learning strategies. For future work, we will continue to study how to effectively incorporate knowledge from quantum mechanics. In the meantime, we sincerely thank the OGB team for their consistent hard work and effort in maintaining the contest and pushing the developments of graph neural networks.

References

Xiaomin Fang, Lihang Liu, Jieqiong Lei, Donglong He, Shanzhuo Zhang, Jingbo Zhou, Fan Wang, Hua Wu, and Haifeng Wang. Chemrl-gem: Geometry enhanced molecular representation learning for property prediction. 2021.

Maho Nakata and Tomomi Shimazaki. PubChemQC Project: A Large-Scale First-Principles Electronic Structure Database for Data-Driven Chemistry. *Journal of Chemical Information and Modeling*, 57(6):1300–1308, June 2017. ISSN 1549-9596. doi:10.1021/acs.jcim.7b00083. URL `https://doi.org/10.1021/acs.jcim.7b00083`. Publisher: American Chemical Society.

Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural message passing for quantum chemistry. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine*

Learning Research, pages 1263–1272. PMLR, 2017a. URL http://proceedings.mlr.press/v70/gilmer17a.html.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=ryGs6iA5Km.

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

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

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. *arXiv preprint arXiv:1310.4546*, 2013.

Herbert Rubenstein and John B Goodenough. Contextual correlates of synonymy. *Communications of the ACM*, 8(10): 627–633, 1965.

Johannes Klicpera, Aleksandar Bojchevski, and Stephan Günnemann. Predict then propagate: Graph neural networks meet personalized pagerank. *arXiv preprint arXiv:1810.05997*, 2018.

Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab, 1999.

Ziteng Liu, Liqiang Lin, Qingqing Jia, Zheng Cheng, Yanyan Jiang, Yanwen Guo, and Jing Ma. Transferable multilevel attention neural network for accurate prediction of quantum chemistry properties via multitask learning. *Journal of Chemical Information and Modeling*, 61(3):1066–1082, 2021. doi:10.1021/acs.jcim.0c01224. URL https://doi.org/10.1021/acs.jcim.0c01224. PMID: 33629839.

Chi Chen, Weike Ye, Yunxing Zuo, Chen Zheng, and Shyue Ping Ong. Graph networks as a universal machine learning framework for molecules and crystals. *Chemistry of Materials*, 31(9):3564–3572, Apr 2019. ISSN 1520-5002. doi:10.1021/acs.chemmater.9b01294. URL http://dx.doi.org/10.1021/acs.chemmater.9b01294.

Steven Kearnes, Kevin McCloskey, Marc Berndl, Vijay Pande, and Patrick Riley. Molecular graph convolutions: moving beyond fingerprints. *Journal of Computer-Aided Molecular Design*, 30(8):595–608, Aug 2016. ISSN 1573-4951. doi:10.1007/s10822-016-9938-8. URL http://dx.doi.org/10.1007/s10822-016-9938-8.

Qiming Sun, Xing Zhang, Samragni Banerjee, Peng Bao, Marc Barbry, Nick S. Blunt, Nikolay A. Bogdanov, George H. Booth, Jia Chen, Zhi-Hao Cui, Janus J. Eriksen, Yang Gao, Sheng Guo, Jan Hermann, Matthew R. Hermes, Kevin Koh, Peter Koval, Susi Lehtola, Zhendong Li, Junzi Liu, Narbe Mardirossian, James D. McClain, Mario Motta, Bastien Mussard, Hung Q. Pham, Artem Pulkin, Wirawan Purwanto, Paul J. Robinson, Enrico Ronca, Elvira R. Sayfutyarova, Maximilian Scheurer, Henry F. Schurkus, James E. T. Smith, Chong Sun, Shi-Ning Sun, Shiv Upadhyay, Lucas K. Wagner, Xiao Wang, Alec White, James Daniel Whitfield, Mark J. Williamson, Sebastian Wouters, Jun Yang, Jason M. Yu, Tianyu Zhu, Timothy C. Berkelbach, Sandeep Sharma, Alexander Yu. Sokolov, and Garnet Kin-Lic Chan. Recent developments in the PySCF program package. *The Journal of Chemical Physics*, 153(2):024109, July 2020. ISSN 0021-9606. doi:10.1063/5.0006074. URL https://aip.scitation.org/doi/abs/10.1063/5.0006074. Publisher: American Institute of Physics.

Chengxuan Ying, Mingqi Yang, Shuxin Zheng, Guolin Ke, Shengjie Luo, Tianle Cai, Chenglin Wu, Yuxin Wang, Yanming Shen, and Di He. First place solution of kdd cup 2021 ogb large-scale challenge graph prediction track, 2021.