A Dataset Details

Our study utilizes 15 molecule datasets of varying sizes obtained from MoleculeNet [14, 7] and ChemBL [12], which can be categorized into three groups based on their applications or source domains: medication (HIV, SIDER, Lipo, ClinTox), quantum mechanics (qm8, qm9), and chemical analysis (ESOL, FreeSolv, ChemBL, MUV, BACE, BBBP, ToxCast, Tox21, PCBA). As for qm8 and qm9, we randomly sample 3,000 graphs to construct the datasets. We use the original split setting, where qm8 and qm9 are randomly split, and scaffold splitting is used for the others. The small molecule datasets with less than 10,000 instances in the training sets are selected as target molecule datasets, i.e., Tox21, BACE, BBBP, ClinTox, SIDER, ToxCast, ESOL, FreeSolv, Lipo, qm8 and qm9. All the involved datasets can be accessed and downloaded from OGB[1] or MoleculeNet repository[2]. The overall statistics are summarized as follows:

	HIV	PCBA	Tox21	BACE	BBBP	ClinTox	MUV
#graphs	41,127	437,929	7,831	1,513	2,039	1,477	93,087
#tasks	1	128	12	1	1	2	17
split	scaffold	scaffold	scaffold	scaffold	scaffold	scaffold	scaffold
metric	ROAUC	AP	ROAUC	ROAUC	ROAUC	ROAUC	AP

Table 1: Dataset statistics(1).

	SIDER	ToxCast	ChemBL	ESOL	FreeSolv	Lipo	qm8	qm9
#graphs	1,427	8,576	456,309	1,128	642	4,200	3,000	3,000
#tasks	27	617	1,310	1	1	1	12	12
split	scaffold	scaffold	scaffold	scaffold	scaffold	scaffold	random	random
metric	ROAUC	ROAUC	ROAUC	RMSE	RMSE	RMSE	MAE	MAE

Table 2: Dataset statistics(2).

B Implementation Details

The core code is provided in the supplementary material.

Backbone model settings. As for GIN [15], we fix the batch size as 128 and train the model for 50 epochs. We use Adam [9] with a learning rate of 0.001 for optimization. The hidden size and number of layers are set as 300 and 5 respectively. We set the dropout rate as 0.5 and apply batchnorm [8] in each layer. All the results are reported after 5 different random seeds.

As for Graphormer [16], we fix the batch size as 128 and train the model for 30 epochs. AdamW [11] with a learning rate of 0.0001 is used as the optimizer. The hidden size, number of layers, and number of attention heads are set as 512, 5, and 8 respectively. We set the dropout rate and attention dropout rate as 0.1 and 0.3. Layernorm [2] is applied across layers. The maximum number of nodes and the distance between nodes in the sampled graph is set as 128 and 5 respectively. The size of position embedding, in-degree embedding, and out-degree embedding are fixed as 512. All the results are reported after 5 different random seeds.

Grouping method settings. As for MolGroup, we apply GIN as the encoder. During the training, we randomly select datasets with equal probability and sample data from them to ensure that each dataset contributes an equal number of samples to the constructed mini-batch. We use MACC [4] fingerprint to calculate the structure affinity score. We fix the number of the filtering iterations as 3 and the balance coefficient λ is set as 0.9. During each round, the auxiliary datasets with affinity scores below 0.6 will be filtered out. If none of the datasets fall below this threshold, we filter out the dataset with the lowest affinity score. The orthogonal initialization [13] is applied to initialize the learnable task embedding $\mathbf{e}^{\text{task}}$ to make sure that the dot product between the target dataset'

[1] https://ogb.stanford.edu/
[2] https://moleculenet.org/

Algorithm 1: Iterative filtering process with MolGroup

Input: Target dataset $\mathcal{D}_T$ with N training instances; Candidate auxiliary datasets $\{\mathcal{D}_A\}_M$; GNN model with specific parameters $\theta_T, \{\theta_m\}_M$ for each dataset and routing mechanism $g(\cdot)$; Number of iterations R; Number of epochs E; Learning rate lr; Batch size B.

```
// Filtering round
for r ← 1, ··· , R do
    Random initialize θ_T, θ_1, ··· , θ_M and g(·).
    φ_1, ··· , φ_M ← 0. // Affinity scores
    I ← NM//B. // Number of iteration in each epoch
    // Training epoch
    for e ← 1, ··· , E do
        // Training step
        for iter ← 1, ··· , I do
            Sample mini-batch {B_T, B_1, ··· , B_M} from current datasets.
            Obtain losses l_T, {l_m}_M and affinity scores {α_m}_M by feeding mini-batch to GNN.
            // Bi-level optimization framework
            1) θ_T({α_m}_M) ← θ_T − lr · ∇_{θ_T} Σ_m^M l_m.
            2) Update g(·) through target dataset's loss function with B_T and θ_T({α_m})_M.
            Update all the parameters θ_T, θ_1, ··· , θ_M through l_T, l_1, ··· , l_M.
            if e == E then
                // Average affinity scores in final epoch
                for m ← 1, ··· , M do
                    φ_m ← φ_m + α_m/I.
                end
            end
        end
    end
    // Remove datasets according to threshold
    {D_A}_{M'} ← {D_{A_m} | φ_m ≥ 0.6}.
    M ← M'.
end
```

Output: Auxiliary datasets with high affinity $\{\mathcal{D}_A\}_M$.

embedding and auxiliary dataset's embedding starts with 0. The task embedding size is set as 16, and the number of the processing steps in Set2Set is set as 2. The pseudo-code is presented in Algo.1.

As for beam search, we apply GIN as the encoder. The beam width and search depth are both set to 3. During the search process, we train the model for 3 epochs using each candidate dataset and evaluate its performance using the validation set loss. Additionally, we consider a criterion based on the combination of the difference in fingerprint distribution and performance. Specifically, we average and normalize these two metrics to determine the criterion.

As for Task2vec [1], we employ GIN as the probe network and follow the official implementation[3]. First, we fix the encoder and train the decoder for 10 epochs. Then we apply the Monte Carlo algorithm to compute the Fisher Information Matrix.

As for TAG [5], we use GIN as the encoder and follow the official implementation[4]. TAG involves training all the datasets and computing the lookahead loss between target dataset and auxiliary datasets. The lookahead loss is accumulated over multiple epochs and used as the affinity scores.

As for MTDNN [10], we train all the datasets together for each target dataset and apply an additional task discriminator to classify the source of the dataset for the input instances. We train 50 epochs for GIN and 30 epochs for Graphormer, and the instances in auxiliary datasets that have a probability greater than 0.6 of being classified as the target dataset will be selected.

As for Gradnorm [3], we train all the datasets together and update the weights of the loss based on the gradients of the last shared GNN layer.

[3] https://github.com/awslabs/aws-cv-task2vec

[4] https://github.com/google-research/google-research/tree/master/tag

C Experimental Details

For the preliminary analysis shown in Fig.1, we conduct the study using a set of 15 molecule datasets. Among these datasets, 11 datasets that have less than 10,000 instances in the training set are selected as target datasets. We pair the target datasets with every other dataset and measure the relative improvement the combination achieves. To mitigate the issue of varying dataset sizes, we upsample or downsample the training sets of all datasets to ensure an equal number of training instances, specifically 5,000 instances. All the reported results are based on 5 different random seeds.

For the dataset grouping evaluation, we train the model using the combined datasets and assess its performance on the target datasets. We then report the model's performance on the test set using the best-performing model selected based on its performance on the validation set. Cross-entropy loss is used for classification tasks and mean squared error loss is used for regression tasks. The overall training loss is calculated as the unweighted mean of the losses for all included tasks. All the reported results are based on 5 different random seeds.

D Overall Experimental Results

Running environment. The experiments are conducted on a single Linux server with The Intel Xeon Gold 6240 36-Core Processor, 361G RAM, and 4 NVIDIA A100-40GB. Our method is implemented on PyTorch 1.10.0 and Python 3.9.13.

D.1 Dataset Grouping Evaluation

Here we present the performance comparison over the other 5 target datasets in Table 3 and Table 4. It can be observed that pretrained Graphormer outperforms GIN significantly, consistent with the previous studies. In addition, MolGroup achieves the best performance in most cases, with an average relative improvement of 3.52% and 3.10% for GIN and Graphormer. However, a notable exception occurs with qm9 where our proposed method is unable to surpass beam search. In this instance, MolGroup assigns low-affinity scores to each auxiliary dataset due to the significant disparity between quantum chemistry and the other domains, as shown in Section D.3. Nonetheless, despite the limited efficiency, beam search is capable of identifying a promising candidate by directly comparing the performance of different groupings. It is worth noting that overall, our proposed MolGroup still achieves better performance compared to the other baseline methods.

Additionally, we attribute the poor performance or even worse results of the Unweighted Average, Gradnorm, and Pretrain-Finetune methods to the significant distribution gap between different datasets. These methods struggle to learn a shared representation that can effectively capture the characteristics of all the datasets.

Method	SIDER(↑)	ToxCast(↑)	ESOL(↓)	Lipo(↓)	qm9(↓)
Only-target	$59.35_{0.010}$	$60.69_{0.010}$	$1.563_{0.040}$	$0.8063_{0.015}$	$0.0303_{0.001}$
Beam search(P)	$54.25_{0.024}$	$\underline{63.37_{0.004}}$	$1.431_{0.058}$	$\underline{0.8073_{0.011}}$	$0.0456_{0.001}$
Beam search(P+S)	$56.58_{0.038}$	$61.01_{0.010}$	$1.476_{0.061}$	$0.8147_{0.014}$	$\underline{\mathbf{0.0258_{0.000}}}$
TAG	$55.64_{0.005}$	$58.08_{0.003}$	$\underline{1.417_{0.066}}$	$0.8170_{0.017}$	$0.0453_{0.000}$
Task2vec	$55.74_{0.008}$	$57.45_{0.004}$	$1.436_{0.050}$	$0.8078_{0.010}$	$0.0468_{0.000}$
MTDNN	$55.95_{0.015}$	$59.02_{0.007}$	$1.499_{0.027}$	$0.8155_{0.017}$	$0.0476_{0.001}$
UA	$\underline{56.26_{0.009}}$	$59.93_{0.003}$	$1.480_{0.060}$	$0.9130_{0.017}$	$0.0494_{0.000}$
Gradnorm	$55.69_{0.005}$	$53.31_{0.010}$	$1.541_{0.091}$	$1.0755_{0.001}$	$0.0574_{0.005}$
Pretrain-Finetune	$51.43_{0.009}$	$51.06_{0.005}$	$1.480_{0.099}$	$1.0776_{0.066}$	$0.0498_{0.010}$
MolGroup	$\mathbf{59.21_{0.013}}$	$\mathbf{63.91_{0.005}}$	$\mathbf{1.402_{0.037}}$	$\mathbf{0.7996_{0.007}}$	$0.0303_{0.001}$

Table 3: Performance comparison of GIN on target molecule datasets, with ↑ indicating higher is better and ↓ indicating lower is better.

Method	SIDER(↑)	ToxCast(↑)	ESOL(↓)	Lipo(↓)	qm9(↓)
Only-target	$62.05_{0.021}$	$66.16_{0.004}$	$1.054_{0.053}$	$0.7432_{0.032}$	$0.0273_{0.001}$
Beam search(P)	$60.80_{0.005}$	$\underline{67.36_{0.006}}$	$\underline{0.989_{0.058}}$	$0.7640_{0.031}$	$0.0399_{0.002}$
Beam search(P+S)	$62.67_{0.009}$	$66.00_{0.003}$	$1.026_{0.024}$	$0.7524_{0.017}$	$\underline{\mathbf{0.0262_{0.000}}}$
TAG	$\underline{63.57_{0.004}}$	$65.40_{0.005}$	$1.015_{0.045}$	$\underline{0.7507_{0.014}}$	$0.0432_{0.002}$
Task2vec	$60.69_{0.019}$	$63.28_{0.005}$	$0.997_{0.028}$	$0.7562_{0.015}$	$0.0429_{0.001}$
MTDNN	$61.43_{0.024}$	$65.05_{0.022}$	$1.045_{0.003}$	$0.7716_{0.032}$	$0.0420_{0.000}$
UW	$58.24_{0.014}$	$62.71_{0.020}$	$1.120_{0.084}$	$0.7887_{0.067}$	$0.0508_{0.000}$
Gradnorm	$51.65_{0.018}$	$52.40_{0.007}$	$1.400_{0.096}$	$1.0482_{0.029}$	$0.2154_{0.125}$
MolGroup	$\mathbf{63.75_{0.018}}$	$\mathbf{68.68_{0.003}}$	$\mathbf{0.978_{0.047}}$	$\mathbf{0.7304_{0.018}}$	$0.0273_{0.001}$

Table 4: Performance comparison using Graphormer on target molecule datasets.

D.2 Takeaway

Grouping more high-affinity datasets improves performance. Previous studies on task grouping have assumed that low-order relationships can be an effective indicator of high-order ones. It suggests that combining multiple source datasets, which can benefit the target dataset, leads to better performance when learned together. Our experimental results also confirm this phenomenon. We take eight datasets as examples and add them one by one with the top three datasets having the highest affinity score as measured by MolGroup. Results are presented in Table 5, where Top$\{a, b, \cdots\}$ denotes the combination of auxiliary datasets with top-performing ones. We can find that combining more auxiliary datasets leads to better performance in most cases. Besides, training with high-affinity datasets can significantly reduce the variant of FreeSolv ($1.579 \rightarrow 0.279$), indicating a more robust representation learned from the auxiliary datasets.

Combinations	ClinTox	Tox21	FreeSolv	BBBP	BACE	ToxCast	ESOL	Lipo
Only Target	$56.45_{0.023}$	$74.23_{0.005}$	$3.842_{1.579}$	$66.62_{0.028}$	$75.02_{0.026}$	$60.69_{0.010}$	$1.563_{0.040}$	$0.8063_{0.015}$
+ Top{1}	$57.77_{0.028}$	$74.81_{0.006}$	$3.563_{0.989}$	$67.36_{0.023}$	$71.27_{0.045}$	$62.66_{0.002}$	$1.502_{0.043}$	$0.8096_{0.014}$
+ Top{1,2}	$57.48_{0.032}$	$75.22_{0.003}$	$3.462_{0.970}$	$\mathbf{68.64_{0.012}}$	$71.13_{0.019}$	$63.18_{0.006}$	$1.524_{0.077}$	$0.8078_{0.011}$
+Top{1,2,3}	$\mathbf{59.77_{0.027}}$	$\mathbf{75.66_{0.004}}$	$\mathbf{3.116_{0.279}}$	$68.36_{0.016}$	$\mathbf{77.33_{0.015}}$	$\mathbf{63.91_{0.005}}$	$\mathbf{1.402_{0.010}}$	$\mathbf{0.7996_{0.005}}$

Table 5: Performance of top-k combinations.

PCBA is an effective booster. One interesting finding is that dataset PCBA [6] can boost the performance of most of the small molecule datasets, as shown in Table 6. This dataset offers both a diverse range of chemical compounds with unique scaffold structures, comprising over 350,000 training instances, and an extensive collection of 128 bioassay annotations that represent a broad range of biological activities, making it a potent booster for small molecule property prediction tasks that can benefit from both structure and task.

	BBBP(↑)	ClinTox(↑)	ToxCast(↑)	Tox21(↑)	ESOL(↓)	FreeSolv(↓)	Lipo(↓)
Only-target	$66.62_{0.028}$	$56.45_{0.023}$	$60.69_{0.010}$	$74.23_{0.005}$	$1.563_{0.040}$	$3.842_{1.579}$	$0.8063_{0.015}$
+PCBA	$67.11_{0.023}$	$57.77_{0.028}$	$62.05_{0.007}$	$74.81_{0.006}$	$1.463_{0.020}$	$3.563_{0.989}$	$0.8021_{0.009}$

Table 6: Cases whose performance is improved by PCBA.

D.3 Case Study

To give an intuition of the learning process of MolGroup, we show the learning curves of the affinity scores in different filtering rounds in Fig. 1. In each round, auxiliary datasets with affinity scores below 0.6 are removed, and if none of the datasets fall below this threshold, the dataset with the lowest affinity score is filtered out. It can be observed that a significant number of auxiliary datasets are removed in the first or second round. Furthermore, the learning curves tend to converge after 6 epochs in most cases. The dataset PCBA remains in the final round in most cases, indicating its general benefit to the other target datasets. We also notice that the majority of auxiliary datasets are

assigned high-affinity scores for FreeSolv, as demonstrated in the preliminary experiment in Section 1, suggesting that all auxiliary datasets contribute positively to its performance.

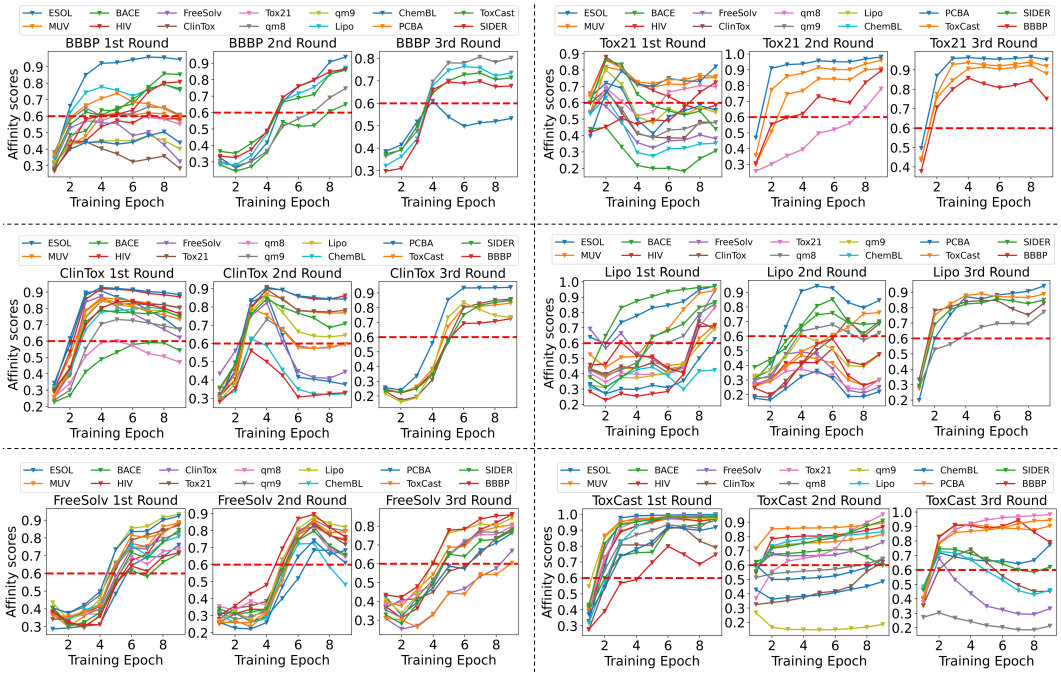

Figure 1: Learning curves of affinity scores where red dashed line represents threshold.

Additionally, we plot the learning curves for qm8 and qm9, which don't have suggested auxiliary datasets. As shown in Fig. 2, all the auxiliary datasets are assigned affinity scores lower than the threshold for both qm8 and qm9, resulting in the removal of all datasets after the first round. We attribute this to the large discrepancy in the structure and task between quantum chemistry and the other domains such as medication. Molecules in quantum chemistry have diverse structures, including both natural and hypothetical compounds. They are studied to explore the behaviors of various functional groups. On the other hand, molecules in medication are more focused on structure. Their study revolves around predicting and optimizing molecular properties relevant to drug design.

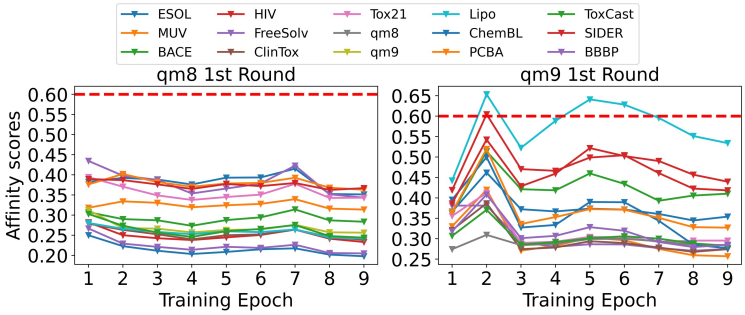

Figure 2: Learning curves for qm8 and qm9.

E Parameter Analysis

E.1 Analysis on balance coefficient λ

To analyze the impact of the balance coefficient λ, we vary λ in the range of $\{0.9, 0.7, 0.5, 0.3\}$ and pick BBBP, Tox21, and Lipo as examples. The learning curves of the affinity scores corresponding to different values of λ are plotted in Fig.3. From the results, we observe that decreasing λ led to lower affinity scores for the auxiliary datasets. However, these scores fail to effectively discriminate the

affinity of individual auxiliary datasets. One reason for this is the instability introduced by parameter initialization and the per-step level computation of structure affinity scores. As a result, lower values of λ cause the MolGroup to assign lower affinity scores to stabilize the training of the target dataset. In light of this, we suggest setting λ to a high value, i.e., 0.9.

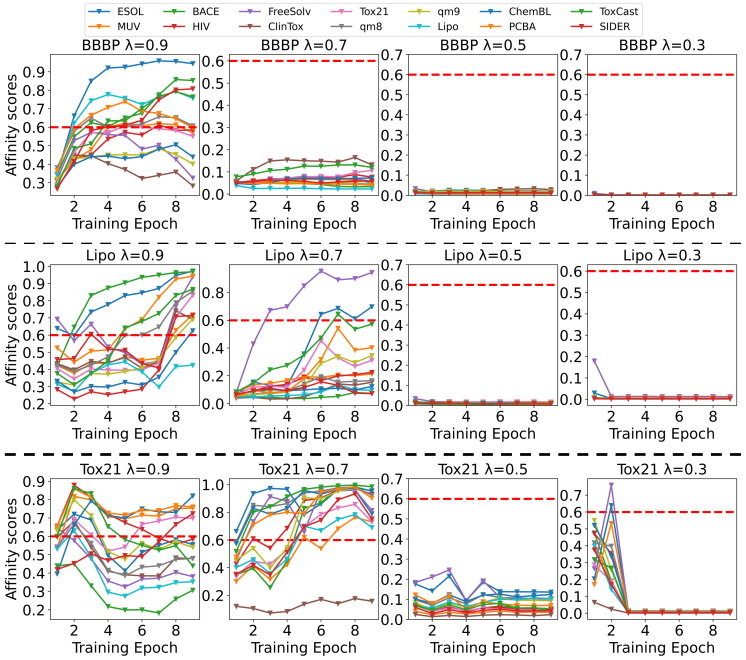

Figure 3: Learning curves of affinity scores with different λ.

E.2 Analysis on number of filtering rounds R

As shown in Fig.1, the auxiliary datasets with the highest affinity scores change as we progress through the different filtering rounds. To investigate the impact of the filtering rounds, we test the performance of the top-3 auxiliary datasets selected in each rounds. The comparison results are shown in Table 7 and the selected datasets are illustrated in Fig. 4. We observe that the auxiliary datasets selected in the 3rd round exhibit the best performance. This is attributed to the filtering process in the 1st and 2nd rounds, which removes negative datasets and helps alleviate interference. As a result, MolGroup can estimate the affinity of each auxiliary dataset more accurately, leading to improved performance on the target dataset.

	BBBP	Tox21	ClinTox	Lipo	FreeSolv	ToxCast
$R=1$	$67.94_{0.018}$	$75.66_{0.004}$	$57.66_{0.027}$	$0.8013_{0.013}$	$3.2288_{0.543}$	$58.82_{0.007}$
$R=2$	$67.99_{0.018}$	$75.66_{0.004}$	$57.66_{0.027}$	$0.8020_{0.031}$	$3.2280_{0.627}$	$63.47_{0.003}$
$R=3$	$\mathbf{68.36_{0.016}}$	$\mathbf{75.66_{0.004}}$	$\mathbf{59.77_{0.027}}$	$\mathbf{0.7996_{0.007}}$	$\mathbf{3.116_{0.279}}$	$\mathbf{63.91_{0.005}}$

Table 7: Performance with different number of filtering rounds R.

F Broader impact

Impact on machine learning research. We propose a novel strategy that combines the routing mechanism with the meta gradient to quantify the impact of one dataset on another. Previously, the routing mechanism was used to increase the model capacity. Our proposed framework can inspire various extensive applications in machine learning, including neural architecture searching (NAS) and data-centric AI. Specifically, our framework enables the network to determine the optimal routing path through the meta gradient. This empowers the network to control and modify the layerwise architecture, leading to improved performance. Besides, it has the potential to enhance data-centric

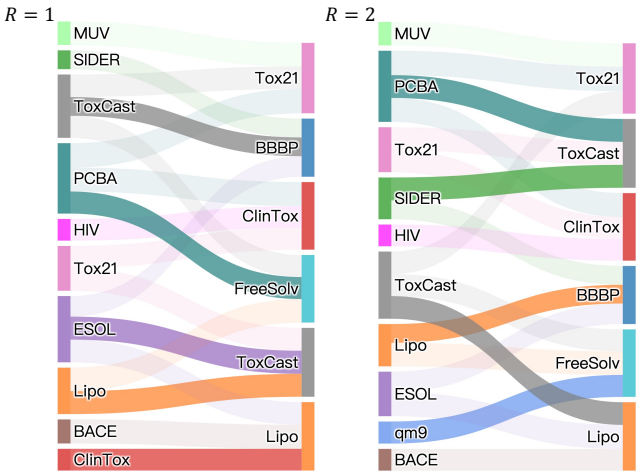

Figure 4: Auxiliary datasets with top-3 affinity scores with different R where we highlight the different edges in these two rounds.

AI approaches by providing a tool to analyze and understand the relationship between different datasets or sub-datasets.

Impact on biological research. We investigate the relationship between molecule datasets, and, focusing on both the structural and task dimensions. our findings shed light on how different molecule datasets impact each other. The analytical approach we employ can be extended to other biological data, such as protein and RNA. Furthermore, our method has the potential to be utilized for data instance filtering in the biological domain. This is particularly important given that biological data often contains various types of noise. By filtering out data instances with negative effects on downstream tasks, we can improve the quality and reliability of biological data used in research.

Impact on the society. Our study has significant implications for society, particularly in the field of biomedicine and drug discovery. By understanding the relationship between molecule datasets and their impact on each other, we can gain insights into how different molecules interact and influence biological processes. It can aid in the development of more effective drugs and therapies by identifying molecules that have a positive impact on specific biological targets or diseases.

References

[1] Alessandro Achille, Michael Lam, Rahul Tewari, Avinash Ravichandran, Subhransu Maji, Charless C Fowlkes, Stefano Soatto, and Pietro Perona. Task2vec: Task embedding for meta-learning. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6430–6439, 2019.

[2] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[3] Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In *International conference on machine learning*, pages 794–803. PMLR, 2018.

[4] Eli Fernández-de Gortari, César R García-Jacas, Karina Martinez-Mayorga, and José L Medina-Franco. Database fingerprint (dfp): an approach to represent molecular databases. *Journal of cheminformatics*, 9(1):1–9, 2017.

[5] Chris Fifty, Ehsan Amid, Zhe Zhao, Tianhe Yu, Rohan Anil, and Chelsea Finn. Efficiently identifying task groupings for multi-task learning. *Advances in Neural Information Processing Systems*, 34:27503–27516, 2021.

[6] Weihua Hu, Matthias Fey, Hongyu Ren, Maho Nakata, Yuxiao Dong, and Jure Leskovec. Ogb-lsc: A large-scale challenge for machine learning on graphs. *arXiv preprint arXiv:2103.09430*, 2021.

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

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

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

[10] Po-Nien Kung, Sheng-Siang Yin, Yi-Cheng Chen, Tse-Hsuan Yang, and Yun-Nung Chen. Efficient multi-task auxiliary learning: selecting auxiliary data by feature similarity. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 416–428, 2021.

[11] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[12] David Mendez, Anna Gaulton, A Patrícia Bento, Jon Chambers, Marleen De Veij, Eloy Félix, María Paula Magariños, Juan F Mosquera, Prudence Mutowo, Michał Nowotka, et al. Chembl: towards direct deposition of bioassay data. *Nucleic acids research*, 47(D1):D930–D940, 2019.

[13] Andrew M Saxe, James L McClelland, and Surya Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. *arXiv preprint arXiv:1312.6120*, 2013.

[14] 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.

[15] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*, 2018.

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