Unimol+XGBoost 配方预测-初测结果

为了充分验证我们的领先性，我们和2024年新发表在arxiv的配方模型——MolSets进行对标。我们完全遵照MolSets对数据的拆分流程和测试方法进行比较。

对比MolSet

数据集的复现

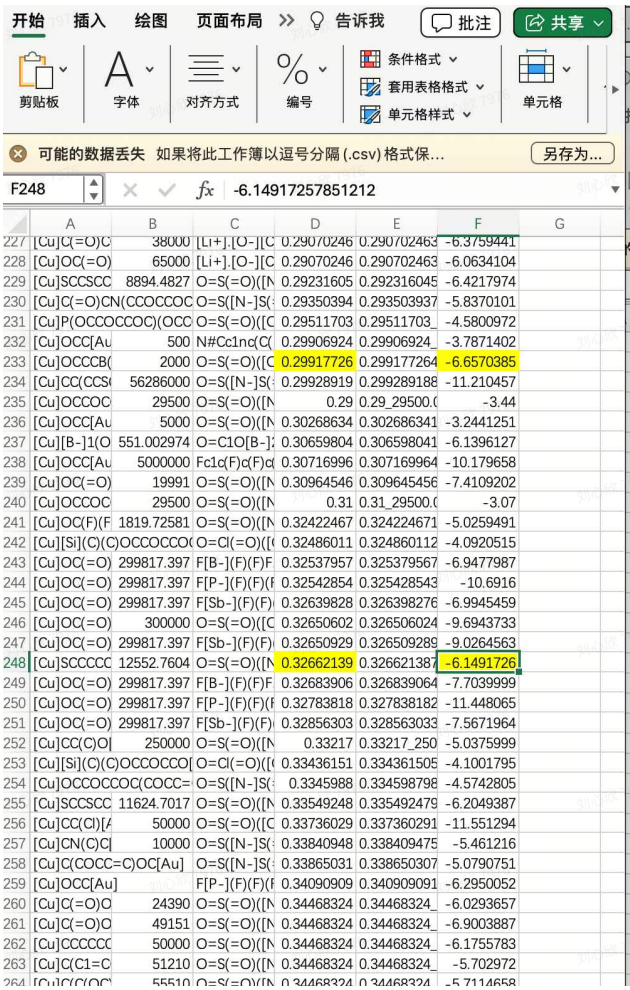
MolSet论文中，用的数据基于ACS cenral 2023，但没有直接使用其中的整份数据集，而是“子集”；对温度的处理是，将分子组成相同但温度不同的entrees集合起来，根据阿累尼乌斯方程拟合成一个TARGET值。

拟合复现

代码如下：

```python
# Arrhenius formula fitting and prediction functions
def fit_and_predict_conductivity(group_data, temp_to_predict):
    # Transformations
    group_data['1/T'] = 1 / group_data['temperature']
    group_data['log_sigma'] = group_data['conductivity']

    # Linear regression features and target
    X = group_data[['1/T']]
    y = group_data['log_sigma']

    # Fit
    model = LinearRegression()
    model.fit(X, y)

    # Predict
    inverse_temp = 1 / temp_to_predict
    predicted_log_sigma = model.predict([[inverse_temp]])
    predicted_log_sigma_col = predicted_log_sigma[0]

    return predicted_log_sigma_col
```

结果如下：



存在一点点误差是因为此处用的273.15和298.15，换用273和298之后结果如下：

可以认为论文中对温度的处理具有科学性；有效性则需要进一步论证。

模型参数

```
xg_reg = xgb.XGBRegressor(
    n_estimators=400, objective='reg:squarederror', colsample_bytree=0.3,
    learning_rate=0.05, max_depth=6, alpha=3,
    # tree_method='gpu_hist' if device.type == 'cuda' else 'auto'
)
```

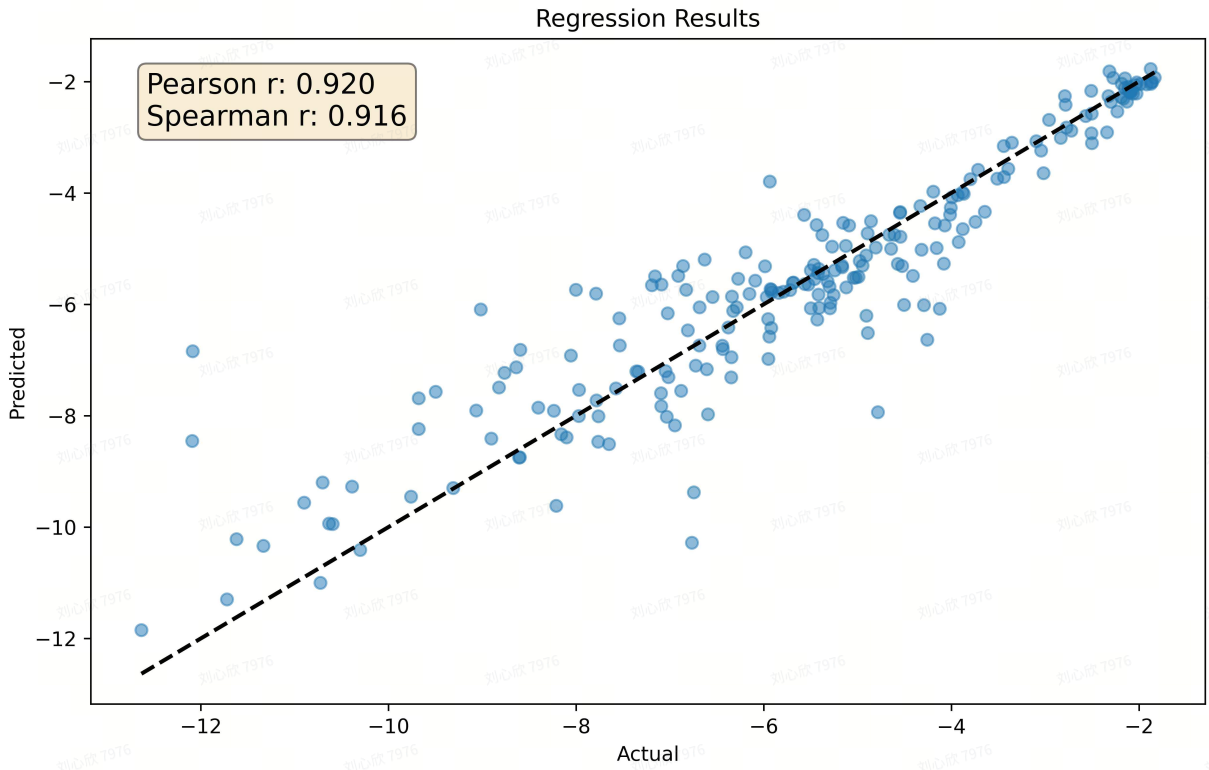
数据集还原

论文中没有直接给出SMILES，尽管数据基于ACS cenral 2023。还原数据集沿用了更早些QSPR论文里我们的格式：

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	SMILES_1	mw_1	SMILES_2	mw_2	SMILES_3	mw_3	SMILES_4	mw_4	SMILES_S1	molality_1	comp_type	mol_type	TARGET
2	CC1CCCO	86.07316	O=C1OCC	88.01604	COC	90.12	CC1COC(=	102.0317	F[As-](F)(F)	1.378305	35	small	-1.81829
3	CC1CCCO	86.07316	O=C1OCC	88.01604	COC	90.12	CC1COC(=	102.0317	F[As-](F)(F)	1.30887	35	small	-1.85078
4	O=C1OCC	88.01604	CC1CCCO	86.07316	CC1COC(=	102.0317	COC	46.04186	F[As-](F)(F)	1.287324	36	small	-1.87347
5	CC1CCCO	86.07316	O=C1OCC	88.01604	COC	90.12	CC1COC(=	102.0317	F[As-](F)(F)	1.296111	35	small	-1.89452
6	COC	46.04186	CC1CCCO	86.07316	O=C1OCC	88.01604	CC1COC(=	102.0317	F[As-](F)(F)	1.527497	37	small	-2.02481
7	COC	46.04186	CC1CCCO	86.07316	O=C1OCC	88.01604	CC1COC(=	102.0317	F[As-](F)(F)	1.429354	37	small	-1.92108
8	O=C1OCC	88.01604	CC1CCCO	86.07316					F[As-](F)(F)	0.919927	38	small	-1.97028
9	ClCCl	83.95336	CC1CCCO	86.07316	O=C1OCC	88.01604	CC1COC(=	102.0317	F[As-](F)(F)	0.843996	39	small	-2.05351
10	ClCCl	83.95336	CC1CCCO	86.07316	O=C1OCC	88.01604	CC1COC(=	102.0317	F[As-](F)(F)	0.910995	39	small	-1.84209
11	ClCCl	83.95336	CC1CCCO	86.07316	O=C1OCC	88.01604	CC1COC(=	102.0317	F[As-](F)(F)	0.98955	39	small	-1.84806
12	CC1CCCO	86.07316	O=C1OCC	88.01604	CC1COC(=	102.0317			F[As-](F)(F)	0.576945	40	small	-2.0281
13	O=C1OCC	88.01604	CC1CCCO	86.07316	CC1COC(=	102.0317			F[As-](F)(F)	1.120814	41	small	-2.09901
14	CC1CCCO	86.07316	O=C1OCC	88.01604	CC1COC(=	102.0317			F[As-](F)(F)	1.137112	40	small	-1.96305
15	CC1CCCO	86.07316	O=C1OCC	88.01604	CC1COC(=	102.0317			F[As-](F)(F)	1.15389	40	small	-2.00208
16	CC1CCCO	86.07316	O=C1OCC	88.01604	CC1COC(=	102.0317			F[As-](F)(F)	1.730836	40	small	-2.06168
17	CC1CCCO	86.07316	O=C1OCC	88.01604	CC1COC(=	102.0317			F[As-](F)(F)	2.307781	40	small	-2.24182
18	CC1CCCO	86.07316	O=C1OCC	88.01604	CC1COC(=	102.0317			F[As-](F)(F)	3.461671	40	small	-2.77267
19	CC1CCCO	86.07316	C1CCOC1	72.05751	O=C1OCC	88.01604	CC1COC(=	102.0317	F[As-](F)(F)	1.133226	42	small	-1.87623
20	C1CCOC1	72.05751	CC1CCCO	86.07316	O=C1OCC	88.01604	CC1COC(=	102.0317	F[As-](F)(F)	1.131017	43	small	-1.83686
21	C1CCOC1	72.05751	CC1CCCO	86.07316	O=C1OCC	88.01604	CC1COC(=	102.0317	F[As-](F)(F)	1.128816	43	small	-1.81294
22	CC1CCCO	86.07316	CC1=CC=(	92.0626	O=C1OCC	88.01604	CC1COC(=	102.0317	F[As-](F)(F)	1.141828	44	small	-1.83618
23	ClCCl	83.95336	CC1CCCO	86.07316					F[As-](F)(F)	0.84959	45	small	-2.48464
24	ClCCl	83.95336	CC1CCCO	86.07316					F[As-](F)(F)	0.921916	45	small	-2.27469
25	CC1CCCO	86.07316	ClCCl	83.95336					F[As-](F)(F)	1.007702	46	small	-2.31562
26	O=S1(CCC	120.0245	CC1CCCO	86.07316					F[As-](F)(F)	0.946565	47	small	-2.2867
27	O=S1(C(C)	134.0402							F[As-](F)(F)	0.793665	48	small	-2.95932
28	O=S1(C(C)	134.0402	COCCOCC	134.0943					F[As-](F)(F)	0.819689	49	small	-2.56866
29	O=S1(C(C)	134.0402	CC1=CC=(	92.0626					F[As-](F)(F)	0.846546	50	small	-2.66461
30	O=S1(C(C)	134.0402	CC1=CC=(	92.0626					F[As-](F)(F)	0.907021	50	small	-2.67902

预处理环节中我们发现了这篇论文处理数据集的一些问题 (详见“发现的问题”子标题)，但为了对比模型本身的学习能力，本次preliminary测试中依然沿用了和论文完全一致的数据集。

Regression图



Pearson correlation:

Unimol+XGBoost: 0.91999

MolSet: 0.905

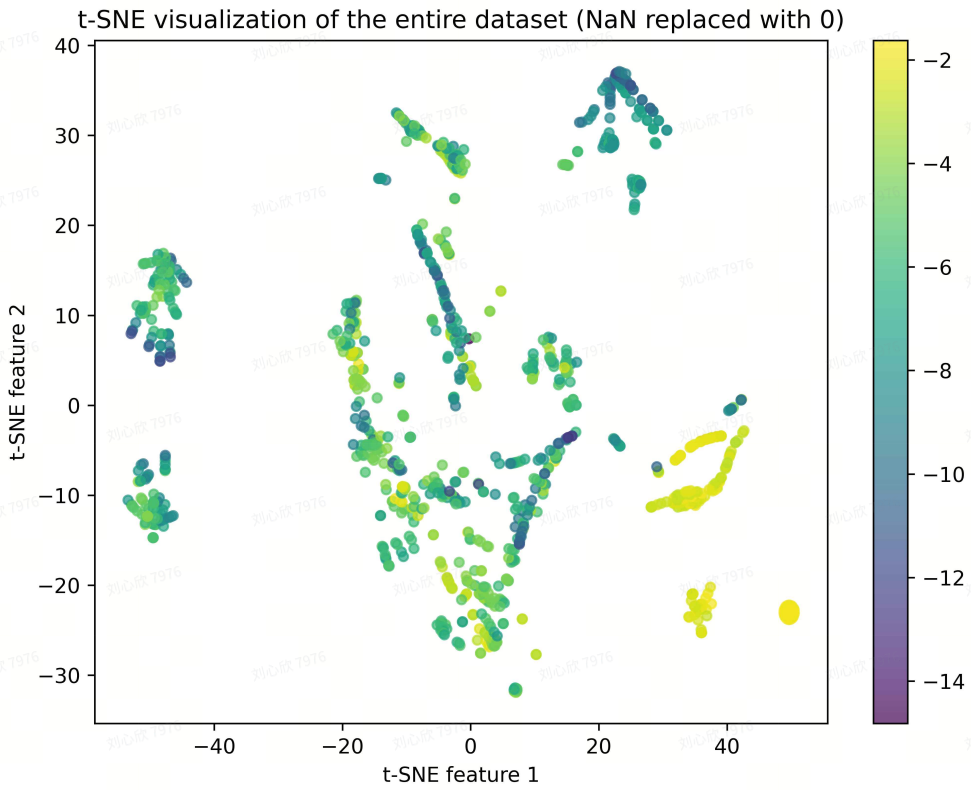
Spearman r:

Unimol+XGBoost: 0.91638

MolSet: 0.907

用与论文完全一致的实验流程和输入，也超越了MolSet的表现。

t-SNE图



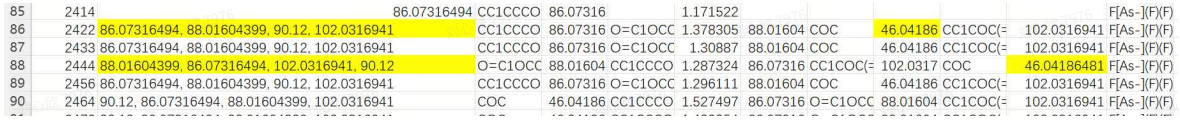
特征太多，或许PCA/热图更直观展示模型学习到的信息。

PCA图

发现的问题

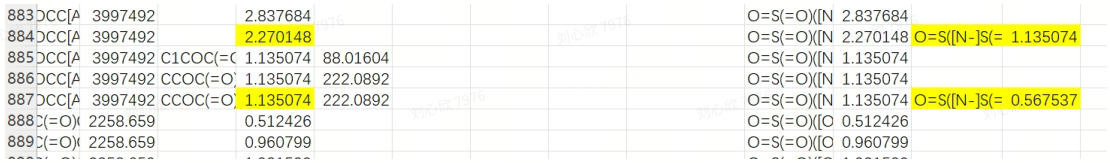
MolSet作为已发表的论文有一些显而易见的问题，例如：

1. mw数据不准确。比如这里，solvent_3的mw_3在molset的pickle导出里是90.12



85	2414		86.07316494	CC1CCCO	86.07316		1.171522					F[As-](F)(F)
86	2422	86.07316494, 88.01604399, 90.12, 102.0316941		CC1CCCO	86.07316	O=C1OCC	1.378305	88.01604	COC	46.04186	CC1COC(=	102.0316941 F[As-](F)(F)
87	2433	86.07316494, 88.01604399, 90.12, 102.0316941		CC1CCCO	86.07316	O=C1OCC	1.30887	88.01604	COC	46.04186	CC1COC(=	102.0316941 F[As-](F)(F)
88	2444	88.01604399, 86.07316494, 102.0316941, 90.12		O=C1OCC	88.01604	CC1CCCO	1.287324	86.07316	CC1COC(=	102.0317	COC	46.04186481 F[As-](F)(F)
89	2456	86.07316494, 88.01604399, 90.12, 102.0316941		CC1CCCO	86.07316	O=C1OCC	1.296111	88.01604	COC	46.04186	CC1COC(=	102.0316941 F[As-](F)(F)
90	2464	90.12, 86.07316494, 88.01604399, 102.0316941		COC	46.04186	CC1CCCO	1.527497	86.07316	O=C1OCC	88.01604	CC1COC(=	102.0316941 F[As-](F)(F)

2. molset的pickle导出里molality是salt_1对应的molality，把少数salt_2无视了。



883	DCC[A	3997492		2.837684						O=S(=O)([N	2.837684	
884	DCC[A	3997492		2.270148						O=S(=O)([N	2.270148	O=S([N-]S(= 1.135074
885	DCC[A	3997492	C1COC(=C	1.135074	88.01604					O=S(=O)([N	1.135074	
886	DCC[A	3997492	CCOC(=O)	1.135074	222.0892					O=S(=O)([N	1.135074	
887	DCC[A	3997492	CCOC(=O)	1.135074	222.0892					O=S(=O)([N	1.135074	O=S([N-]S(= 0.567537
888	C(=O)(	2258.659		0.512426						O=S(=O)([O	0.512426	
889	C(=O)(	2258.659		0.960799						O=S(=O)([O	0.960799	

我认为只有1077条的数据集，这些问题不能忽视。

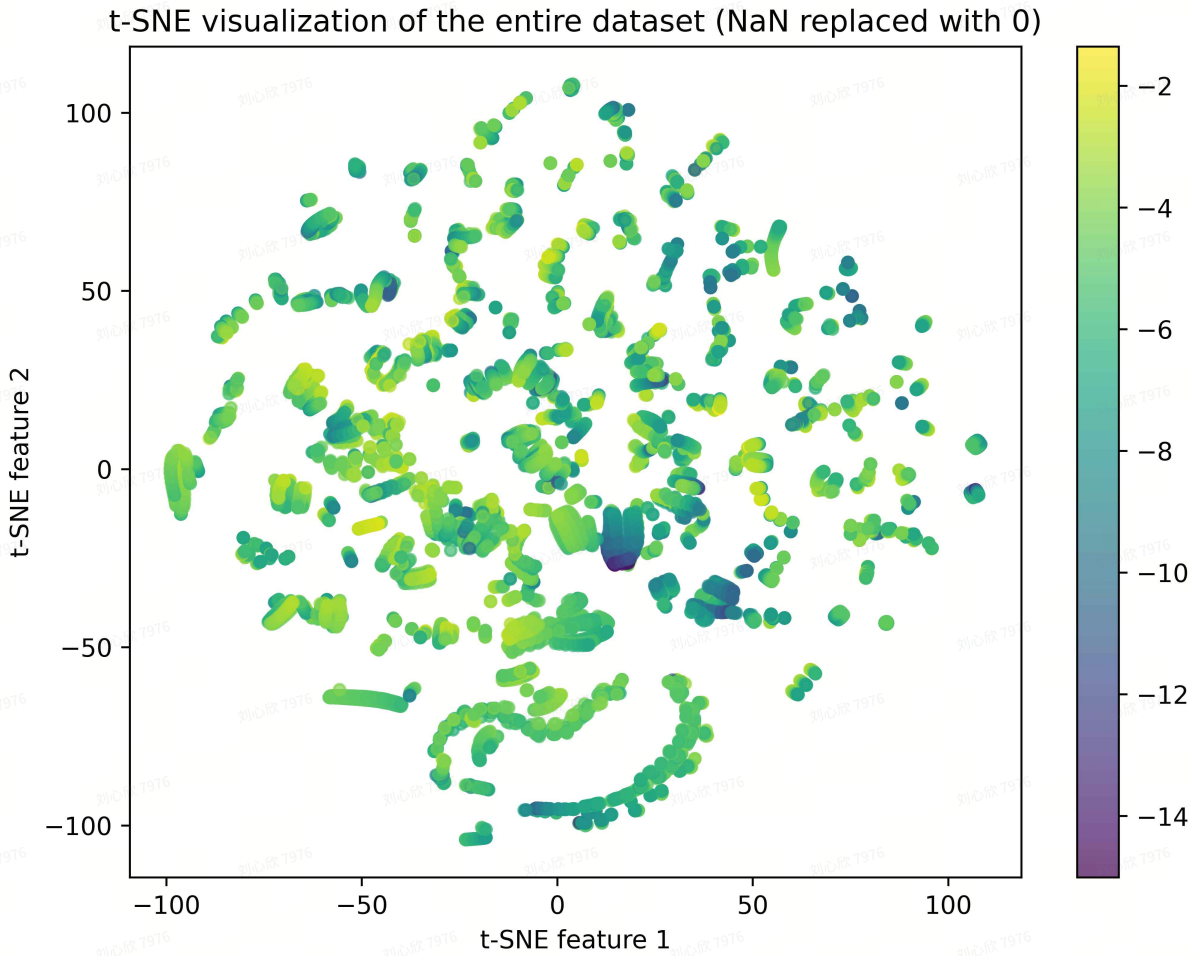
对比ACS central 2023

数据集复现

基本不需要特殊处理；该数据集共10000条；暂未发现明显缺陷。

Regression图

t-SNE



发现的问题

原文的测试流程要求训练集与两个非训练集中不能有相同的polymer分子，在此基础上10-fold crossvalidation。虽然更好地表现了模型的泛化性能，但我认为这样选出的模型参数会有很大欠拟合风险。另外，一万条的数据集不算小数据集，不是很明白为什么要10-fold这么多。我认为多次随机单折、分层抽样更适合【specialist】模型。

总结

Unimol+XGBoost虽简单但有效。