Optimizing MoLFormer-XL for Lipophilicity Prediction via Influence Functions and Efficient Fine-Tuning Exploration

Md Mobashir Rahman
Center for Bioinformatics
mdra00001@stud.uni-saarland.de

Ratnadeep Chakraborty
German Research Center for AI
rach00002@stud.uni-saarland.de

Abstract

This study optimizes MoLFormer-XL for lipophilicity prediction (logD) using SMILES representations. Task 1 compares direct regression fine-tuning (MSE: 0.5196, R^2: 0.6424) to MLM pretraining (MSE: 0.5417, R^2: 0.6272), with hyperparameter tuning yielding a best model (MSE: 0.3817, R^2: 0.7373). Task 2 leverages influence functions for external data selection, improving generalization but increasing test MSE by 7.57% (MSE: 0.4106, R^2: 0.7175) due to data mismatch. Task 3 explores data selection (clustering, target alignment, random) and parameter-efficient fine-tuning (LoRA, iA3, BitFit), with clustering-based selection iA3 performing best (estimated MSE: 1.02, R^2: 0.29651) when applied on the original dataset without adding any external data, however adding external data worsens the model performance. Results highlight embedding-informed data selection and hyperparameter tuning as key to improving model performance.

1 Task1: Fine-Tuning MoLFormer-XL for Lipophilicity Prediction1

We fine-tuned a pre-trained MoLFormer-XL model, originally developed by Ross et al. [Ross et al., 2021], on a Lipophilicity dataset to predict logD values from SMILES representations. The study (: [Rahman, 2025a]) compares direct regression fine-tuning with unsupervised Masked Language Modeling (MLM) pretraining to evaluate their impact on predictive performance. The dataset was stratified into training (80%) and test (20%) sets. Tokenization utilized MoLFormer-XL's tokenizer. A regression head was added following approaches in molecular property prediction [Wang et al., 2024] and trained using MSE loss, AdamW optimization [Loshchilov and Hutter, 2019], gradient accumulation, and early stopping. Additional MLM fine-tuning was conducted before re-training the regression model.

Table 1: (Task 1): Performance Comparison of Fine-Tuning Methods for MoLFormer-XL

Fine-Tuning Method	MSE	MAE	R^2
Direct Fine-Tuning	0.5196	0.5493	0.6424
After MLM Fine-Tuning	0.5417	0.5642	0.6272

Direct fine-tuning outperformed MLM-pretrained regression, suggesting that MoLFormer-XL was already well-suited for lipophilicity prediction. However, this conclusion was based on fixed hyperparameters. To validate and enhance performance, systematic hyperparameter exploration is the next step.

1.1 Hyperparameter Tuning

1.1.1 Initial Grid Sweep

Initially, we used Weights and Biases (WandB) grid sweep [Rahman, 2025b] to explore the hyperparameter space for fine-tuning MoLFormer-XL. The grid search varied epochs, batch sizes, and learning rates to identify the optimal training configuration, summarized in Table 2. Each run was executed sequentially, logging outputs for monitoring and analysis. The process iterated over all possible combinations, evaluating training performance for optimal selection. The top three models, ranked by R^2, outperformed the initial Google Colab run, achieving higher R^2 and lower MSE. The generalization gap, defined as the difference between average validation and training loss at the final supervised epoch, indicates overfitting—larger values suggest the model is fitting the training data more tightly than the validation set.

Hyperparameter	Values
Epochs	3, 5, 10
Batch Size	8, 16, 32
Learning Rate	1e-5, 2e-5, 5e-5

Table 2: Initial Grid search hyperparameter ranges.

Run	Val - Train	Supervised R^2	Supervised MSE	Final Test R^2	Final Test MSE	Final Test MAE
run_15_e5_b16_lr5e-5	0.206	0.714	0.416	0.708	0.425	0.497
run_19_e10_b8_lr1e-5	0.218	0.652	0.506	0.705	0.429	0.493
run_20_e10_b8_lr2e-5	0.280	0.680	0.465	0.702	0.433	0.492

Table 3: Intial Grid Sweep: Top 3 models

1.1.2 Hyperparameter Optimization: Balance of performance and overfitting

Following the analysis of initial results, we refined the hyperparameters by incorporating *weight_decay* and *drop_rate* to enhance model performance while mitigating overfitting. For detailed sweep results, see [Rahman, 2025b].

Configuration	Details
Method	Bayesian Optimization
Accumulation Steps	{1, 2, 4, 8}
Dropout	Uniform(0.1, 0.5)
Epochs	{10, 15}
Learning Rate	Uniform(1e-6, 1e-4)
MLM Epochs	{1, 2, 3}
Train Batch Size	{8, 16, 32, 64}
Weight Decay	Uniform(1e-6, 0.01)

Table 4: Hyperparameter configuration for Bayesian optimization.

We ranked the top 10 models by final test R^2 and MSE, identifying the best configuration. These models demonstrated improved Masked Language Modeling (MLM) learning over direct fine-tuning, highlighting the effectiveness of MLM pretraining. Several models surpassed the initial Google Colab run and preliminary grid search, highlighting the benefits of optimized hyperparameter selection. To assess overfitting, we analyzed the difference between average validation and training loss at the last supervised epoch. A higher positive difference indicated overfitting. Most models showed slight overfitting, while some exhibited minimal overfitting or strong generalization.

We selected the model finetune_1741365955 with the highest R^2 value and slight retrain loss as Task 1 baseline.

1.1.3 Reproducing the Best Model

To assess the reproducibility of our highest-performing model, we conducted a grid sweep using Weights and Biases [Rahman, 2025c].

The experiment was conducted twice using an identical codebase and fixed hyperparameters to as-

Name	Epoch	LR	Accum. Steps	Dropout	MLM Epochs	Train Batch	Weight Decay	Test MSE	Test MAE	Supervised R²	Final R²	MLM Improved	avg val loss - avg train loss)	Overfitting
finetune_1741365955	6	8.5e-5	2	0.104	1	32	0.0034	0.3817	0.4674	0.6993	0.7373	Yes	0.2551	Slight
finetune_1741365398	8	7.5e-5	1	0.128	1	16	0.0007	0.4084	0.4751	0.7197	0.7190	No	0.2952	Slight
finetune_1741377645	11	2.4e-5	1	0.280	1	8	0.0068	0.4098	0.4742	0.6959	0.7180	Yes	0.2806	Slight
finetune_1741375444	10	9.0e-5	4	0.184	2	8	0.0012	0.4137	0.4757	0.7258	0.7153	No	0.3031	Slight
run_15_e5_b16_lr5e-5	5	-	-	-	-	-	-	0.4246	0.4966	0.7140	0.7078	No	0.2060	Minimal
finetune_1741369418	10	8.5e-5	2	0.311	1	64	0.0085	0.4259	0.4859	0.6979	0.7069	Yes	0.2132	Minimal
finetune_1741369949	10	3.8e-5	4	0.288	3	8	0.0069	0.4272	0.4898	0.6795	0.7060	Yes	0.2829	Slight
run_19_e10_b8_lr1e-5	10	-	-	-	-	-	-	0.4290	0.4925	0.6518	0.7048	Yes	0.2181	Minimal
run_20_e10_b8_lr2e-5	10	-	-	-	-	-	-	0.4331	0.4915	0.6799	0.7020	Yes	0.2799	Slight
run_21_e10_b8_lr5e-5	8	-	-	-	-	-	-	0.4365	0.4961	0.7201	0.6997	No	0.2875	Slight

Table 5: (Task1) Summary of the top performing models and corresponding hyperparameter configurations, sorted by Final Test R².

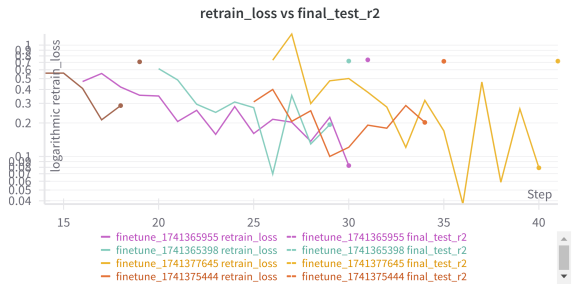

Figure 1: (Task1) Comparison of logarithmic retrain loss and final test R² across different fine-tuning runs. Each line represents an individual model's retrain loss (solid lines) and final test R² values (dashed lines), highlighting relationships between model convergence and predictive performance.

sess reproducibility. Notably, the baseline run (**finetune_1741365955**) exhibited performance deviations when compared to the subsequent runs (**attempt1** and **attempt2**). The experimental sweep configuration is comprehensively outlined in Table 6, while Table 7 presents the key performance metrics for each experimental run, alongside the differences computed relative to the baseline. This discrepancy underscores the critical role of controlled randomization, particularly the explicit setting of random seeds, in ensuring reproducibility within deep learning experiments.

Table 6: Task1 Best Model Reproducibility: Grid Sweep Configuration

Parameter	Value
accumulation_steps	2
dropout	0.10386
epochs	15
lr	0.00009
mlm_epochs	1
train_batch_size	32
weight_decay	0.00342

Table 7: Task1 Best Model Reproducibility: Experimental Results and Differences (Baseline vs. Attempts)

Metric	Baseline	Attempt1	Attempt2	Δ(Attempt1)	Δ(Attempt2)
epoch	6	7	6	+1	0
final_test_mse	0.38175	0.45489	0.40246	+0.07314	+0.02071
final_test_r2	0.73732	0.68699	0.72307	-0.05033	-0.01425
mlm_avg_loss	0.31987	0.31199	0.30173	-0.00788	-0.01814
retrain_loss	0.08287	0.12110	0.13112	+0.03823	+0.04825
supervised_test_mse	0.43695	0.46598	0.46695	+0.02903	+0.03000
supervised_test_r2	0.69932	0.67935	0.67868	-0.01997	-0.02064

Discrepancies in training loss and R^2 scores reveal the influence of stochastic factors like initialization and data shuffling, even under identical conditions. This underscores the need for multiple runs to minimize variability and enhance reproducibility in neural network evaluation.

2 Task 2: Influence Function-Based Data Selection for Improved Model Performance

In this task, we attempted to improve model performance by using **influence functions** and the **LiSSA approximation** to select high-impact external samples. These are integrated for fine-tuning and evaluated against the **Task 1 baseline**. Using the **LiSSA approximation**, we computed influence scores to quantify each sample's impact on test loss, ranging from 0.102 to 2.568, highlighting varying sample relevance and quality for the provided external dataset.

By selecting the top 10% of external samples based on influence scores, we filtered out potentially noisy or unhelpful data, integrating only those most likely to enhance model generalization. Fine-tuning the pre-trained model on the combined dataset (original Lipophilicity training set plus selected external samples) was expected to yield competitive test performance.

Metric	Task 1	Task 2	Change (%)
Test MSE	0.3817	0.4106	+7.57%
Test MAE	0.4674	0.4767	+1.99%
Test R^2	0.7373	0.7175	-2.69%

Table 8: Comparison of model performance metrics of the Task 1 Baseline Model and after integrating external data selected using influence functions. Task 2 shows increased error metrics (MSE and MAE) and reduced explanatory power (R^2).

The drop in performance in Task 2 seems to stem from issues with the external dataset itself. Despite using influence functions to carefully select samples, the data likely contained inconsistencies or didn't align well with the original distribution, making it harder for the model to generalize. As a result, adding this external data actually made things worse—MSE increased by 7.57%, MAE by 1.99%, and R² dropped by 2.69%. This underscores an important point, even well-designed data selection methods can't fix poor-quality data. Before integrating external datasets, it's crucial to check their reliability, and also ensure they truly contribute to better learning rather than introducing noise. (Table: 8)

3 Task 3: Exploration of Data Selection and Fine-Tuning Methods

In Task 3, we explored alternative data selection strategies and parameter-efficient fine-tuning to optimize MoLFormer-XL and the best supervised model from Task 1 for lipophilicity prediction.

3.0.1 Data Selection Strategies

We assessed three distinct methods for selecting subsets of external data:

Target Alignment: Selecting samples whose embeddings closely align with the target dataset's distribution, following Ruder and Plank [Ruder and Plank, 2017], who used Bayesian optimization for data selection in transfer learning.

Random Sampling: Selecting samples randomly without considering embeddings or distributions.

Clustering-based Selection: Selecting representative samples from embedding-based clusters utilizing a k-medoids clustering approach as described by Park and Jun [Park and Jun, 2009]..

The proportion of external data selected varied from 30% to 50%.

3.1 Fine-Tuning Strategies

We examined three parameter-efficient fine-tuning methods:

BitFit: Only bias parameters are updated following the BitFit approach by Ben Zaken et al. [Ben Zaken et al., 2021], which demonstrates efficiency in fine-tuning transformer models.".

LoRA (Low-Rank Adaptation): Adds low-rank adapters to selected linear layers as proposed by Hu et

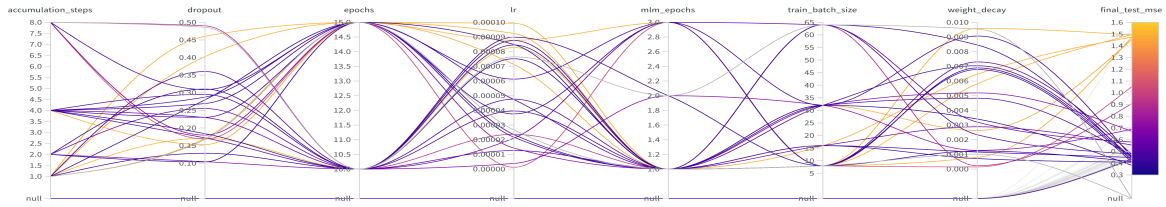

Figure 2: (Task1) A parallel coordinates plot visualizing hyperparameter tuning results. Each line represents a model configuration, linking hyperparameters (accumulation steps, dropout, epochs, learning rate, MLM epochs, train batch size, weight decay) to final test MSE. Darker (purple) lines indicate better performance, while lighter (yellow) lines represent poorer performance.

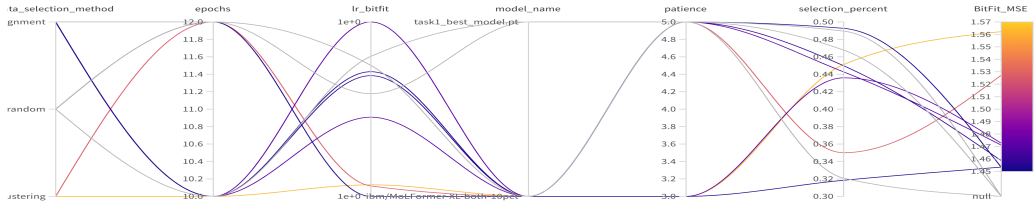

Figure 3: LoRA hyperparameter sweep (MSE).

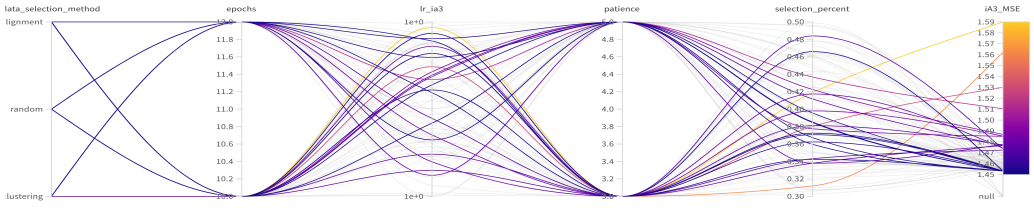

Figure 4: iA3 hyperparameter sweep (MSE).

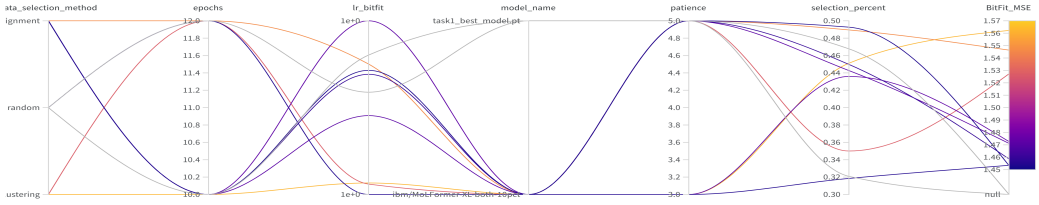

Figure 5: BitFit hyperparameter sweep (MSE).

al. [Hu et al., 2021] for parameter-efficient adaptation of large language models..

iA3 (Implicit Adapter): Introduces learned scaling vectors into selected model layers. [Liu et al., 2022]

Bayesian hyperparameter sweeps were conducted using Weights & Biases (WandB) with the configuration detailed in Table 9.

Table 9: WandB Sweep Configuration

Parameter	Values/Range
Method	Bayesian Optimization
Metrics	iA3_MSE, LoRA_MSE, BITFIT_MSE
Batch Size	16 (fixed)
Data Selection Method	Target Alignment, Random, Clustering
Epochs	10, 12
Learning Rate (LoRA)	Log-Uniform $[1 \times 10^{-6}, 5 \times 10^{-5}]$
Learning Rate (iA3)	Log-Uniform $[5 \times 10^{-6}, 1 \times 10^{-4}]$
Patience	3, 5
Selection Percent	Uniform [0.3, 0.5]
Base Models	ibm/MoLFormer-XL-both-10pct, task1_best_model.pt

Parallel coordinates plots (Figures 3, 4, and 5) illustrate the interactions among hyperparameters and their influence on the Mean Squared Error (MSE) for each fine-tuning method: LoRA, iA3, and BitFit.

3.2 Result Analysis

Selection percentage varied across methods with no clear best approach. **Target Alignment** achieved the lowest MSE for LoRA and iA3, while **Random Selection** performed best for BitFit. **Clustering** reduced training time compared to other methods. BitFit was the fastest, while iA3 and LoRA required significantly longer training. Clustering improved training speed for LoRA, whereas BitFit and iA3 trained fastest with Random Selection.

3.2.1 Fine-Tuning Performance (MSE, Generalization)

With the original dataset **BitFit** achieves the lowest MSE, making it the best fine-tuning approach. **iA3** has the highest MSE, indicating weaker generalization. **Target Alignment** minimizes the generalization gap (train loss $\approx$ validation loss).

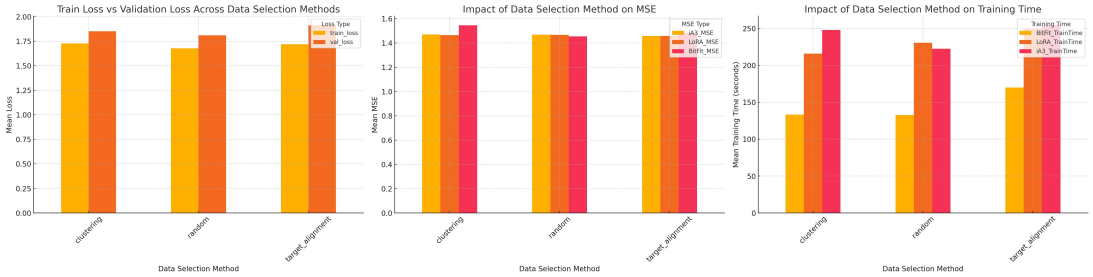

Figure 6: Impact of Data Selection Method on MSE, Training Time, and Loss.

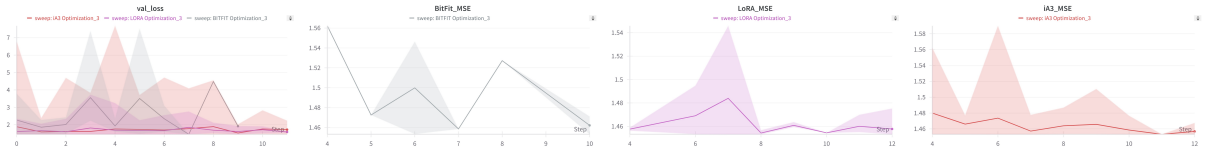

Figure 7: Evolution of mean MSE over Epochs for Different Fine-tuning Strategies.

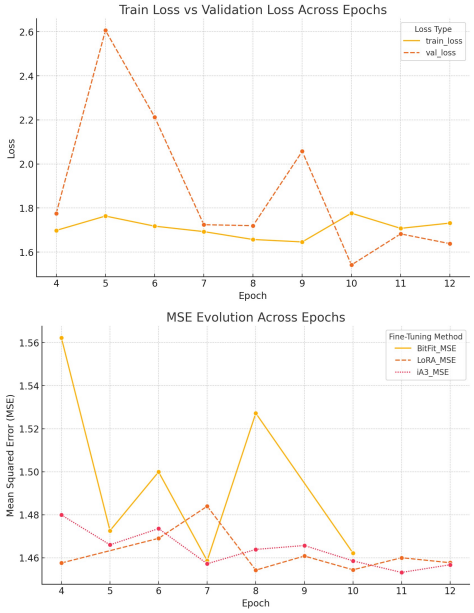

Figure 8: Train Loss vs Validation Loss and MSE Evolution Across Epochs.

3.2.2 Trainable Parameters (Fine-Tuning Efficiency)

BitFit requires the fewest trainable parameters ($\approx 74K$), making it the smallest in terms of size. **LoRA** balances efficiency and accuracy ($\approx 10M$ parameters). **iA3** has the highest computational cost ($\approx 45M$ parameters).

3.2.3 Stability Across Epochs (Convergence)

LoRA converges the fastest and achieves the best R^2 score. **iA3** improves steadily but requires more epochs. **BitFit** has the slowest convergence and lowest R^2, indicating weaker learning capacity.

Table 11 represents the performance of the best iA3, LoRA, and BitFit models in Task 3. The results compare the models' performance when trained solely on the original data, allowing a baseline comparison before integrating additional data, with that when trained with incorporated data from the external dataset. The reported metrics include Mean Squared Error (MSE), coefficient of determination (R^2), training loss, and validation loss.

Criterion	Best Model	Best Data Selection
Lowest MSE (Accuracy)	LoRA	Target Alignment
Fastest Training Time	BitFit	Target Alignment
Generalization (Less Overfitting)	LoRA	Target Alignment
Trainable Parameters (Efficiency)	BitFit	Any
Stability (Convergence Speed)	LoRA	Any

Table 10: Optimal Model and Data Selection Strategy

Model	MSE	R^2	Ext. Data Incorporated?
iA3	1.4574	-0.0029	Yes
LoRA	1.4544	-0.0008	Yes
BitFit	1.4533	-0.0000	Yes
iA3	1.02233	0.29651	No
LoRA	1.31532	0.0949	No
BitFit	1.09152	0.2489	No

Table 11: Performance comparison of iA3, LoRA, and BitFit models in Task3

References

Elad Ben Zaken, Yoav Goldberg, and Shauli Ravfogel. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. *arXiv preprint arXiv:2106.10199*, 2021.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin Raffel. $(ia)^3$: Efficient adapter tuning with input-dependent inference. *arXiv preprint arXiv:2205.05638*, 2022. URL https://arxiv.org/abs/2205.05638.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*, 2019. URL https://arxiv.org/abs/1711.05101.

Hae-Sang Park and Chi-Hyuck Jun. A simple and fast algorithm for k-medoids clustering. *Expert Systems with Applications*, 36(2):3336–3341, 2009.

Md Mobashir Rahman. Task1.pdf: Detailed report for task1.ipynb ran on google colab. see interpretation. https://gitlab.cs.uni-saarland.de/mdra00001/nnti-project/-/blob/main/notebooks/Task1.pdf?ref_type=heads, 2025a. Accessed: 2025-03-09.

Md Mobashir Rahman. Weights and biases grid sweep report for molformer-xl fine-tuning. https://api.wandb.ai/links/mobashirrahman-saarland-university/d9kycyqy, 2025b. Accessed: 2025-03-09.

Md Mobashir Rahman. Nnti task1: Reproducing best model. https://wandb.ai/mobashirrahman-saarland-university/nnti-project/reports/NNTI-Task1-Reproducing-Best-Model--VmlldzoxMTcwNzkyMA?accessToken=twogego6aovdouhmomt14e215ghhettbqxgjcjcheaswh3zcozkiw0wmgitmznjj, 2025c. Accessed: 2025-03-09.

David S Ross, Yichen Xu, Haoran Wang, Jian Su, James Zou, Bo Yu, Emil F Khisamutdinov, Richard P Adams, and Connor W Coley. Large-scale chemical language representations capture molecular structure and properties. *arXiv preprint arXiv:2106.09553*, 2021.

Sebastian Ruder and Barbara Plank. Learning to select data for transfer learning with bayesian optimization. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 372–382. Association for Computational Linguistics, 2017.

Haoran Wang, David S Ross, Yichen Xu, Jian Su, Emil F Khisamutdinov, Richard P Adams, and Connor W Coley. Regression with large language models for materials and molecular property prediction. *arXiv preprint arXiv:2409.06080*, 2024.