Development of a Polymer Property Foundation Model via GPT-2 Fine-Tuning

Arifin [1] **Araki Wakiuchi** [1] **Ryo Yoshida** [2]

[1]*JSR Corporation, Japan* [2]*The Institute of Statistical Mathematics, Japan.* Correspondence to: Arifin fin_ari@jsr.co.jp.

1. Introduction

Data-driven polymer research has increasingly highlighted the need for systematically designed polymer property databases. To address this demand, RadonPy—the first open-source Python library that fully automates all-atom classical MD simulations for polymers—has been developed.[1] The integration of RadonPy into the SPACIER platform via Bayesian optimization has enabled the discovery of optical polymers that overcome traditional trade-offs.[2] To further accelerate discoveries in polymer research, the development of foundational models for polymer properties is essential. In this work, we investigated the predictive performance of large language models (LLMs) by benchmarking their zero-shot and few-shot prompting capabilities and further examined the fine-tuning of these models.

2. Methods

2.1 Dataset and models

We began by evaluating local LLM models for predicting polymer properties such as density, specific heat capacity at constant pressure (C_p), and refractive index. The dataset, obtained from the RadonPy repository, comprised 1,077 datapoints. For the LLM models, we focused on smaller models, including GPT-2, GPT-Neo 1.3B, BLOOMZ-1b7, and Granite-2B-Instruct (v3.0, v3.1, and v3.2).

2.2 Prompting

The example of the prompts is given as,

```
You are a polymer and computational
chemist.  For each example below,
predict the density of the homopolymer:
{examples_text}.  Now, given monomer SMILES
‘{smiles}’, predict density.  ### Output
only a JSON object with one key ‘‘value’’ and
no extra text.  ### Begin JSON:{
```

where ‘{smiles}’ is corresponded to monomer SMILES. In the case of few-shot prompting, 3 datapoints from the dataset are included in {examples_text}.

2.3 Fine tuning

We examined the fine-tuning for the GPT-2 model by adding a regressor head on top of the final hidden-state tensor. Here, we split the dataset into training and testing sets with an 80/20 split, targeting density, C_p, and refractive index. We then ran the model for 10 epochs. Furthermore, we also fine-tuned GPT-2 using a mixture of queries related to density, C_p, and refractive index, along with their corresponding values. In this process, we employed common loss functions for multi-target optimization, called as symmetric mean absolute percentage error (SMAPE).

2.4 Numerical tokenizer and pre-training

In this work, we implemented the special numerical tokenizer that was introduced in OmniPred paper.[3] The transformation ideas are:

1. Assign a token for the overall sign, <+> or <->.
2. For non-zero values, separates the number into a scientific-notation mantissa and base-10 exponent.
3. Wrap each digit in angle-bracket tokens like <3>.
4. After that, the exponent's sign is tokenized (<+> or <->) and each exponent digit is likewise bracketed.
5. A special shortcut for exact zero is `[<+>, <0>]`.
6. The final list therefore has the structure `[sign] + mantissa_digit_tokens + [exp_sign] + exponent_digit_tokens`.

For the pretraining, the learning rate was set to 10^{-4} and the number of epochs was 50. The loss function then becomes the average negative log-likelihood of each true token under the model's predicted next-token distributions (Causal Language Modeling loss).

3. Results and Discussions

Overall performance of zero-shot predictions was disappointing, as they often produced identical values or NaN outputs. Few-shot approach resulted in similar prediction performance across all small models, with an R^2 of around 0.25, and it eliminated NaN outputs completely. Figure 1 show the density predictions for all polymers in the dataset with few-shot prompting.

The performance of the fine tuning with regressor head on the test sets was significantly better than few-shot prompting method, achieving R^2 values of 0.96 for density, 0.83 for C_p, and 0.92 for refractive index. When we employed the mixed dataset, the predictions work well for density and refractive index, with R^2 values of 0.96 and 0.89, respectively. However, the model performed poorly for C_p. This discrepancy is expected since the scale of the C_p values

is significantly different from that of density and refractive index. Table 1 summarized the prediction performances with regression head.

The pretraining with numerical tokenizer gives similar or better prediction performances than the regression head. By employing this technique, we could train mixed dataset and achieved test R^2 values of 0.94, 0.92, and 0.88 for the predictions of density, C_p, and refractive index, respectively (Figure 2).

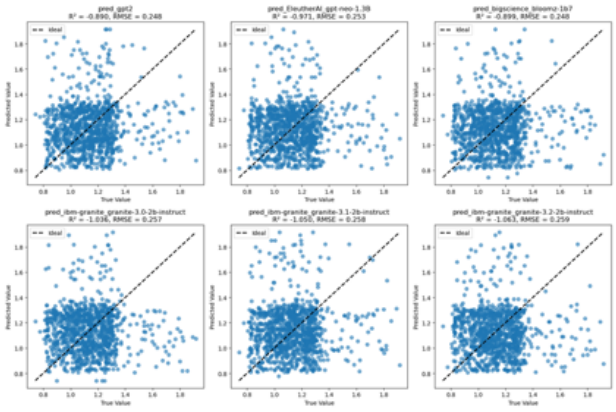

Fig. 1: Few-shot prompting results on several small LLM models

Table 1: Performance results by training method and target for the fine tuning with regressor head.

Training Method	Target	R2
Each Dataset	Density	0.96
	C_p	0.83
	Refractive Index	0.92
Mixed Datasets	Density	0.96
	C_p	-17.45
	Refractive Index	0.89

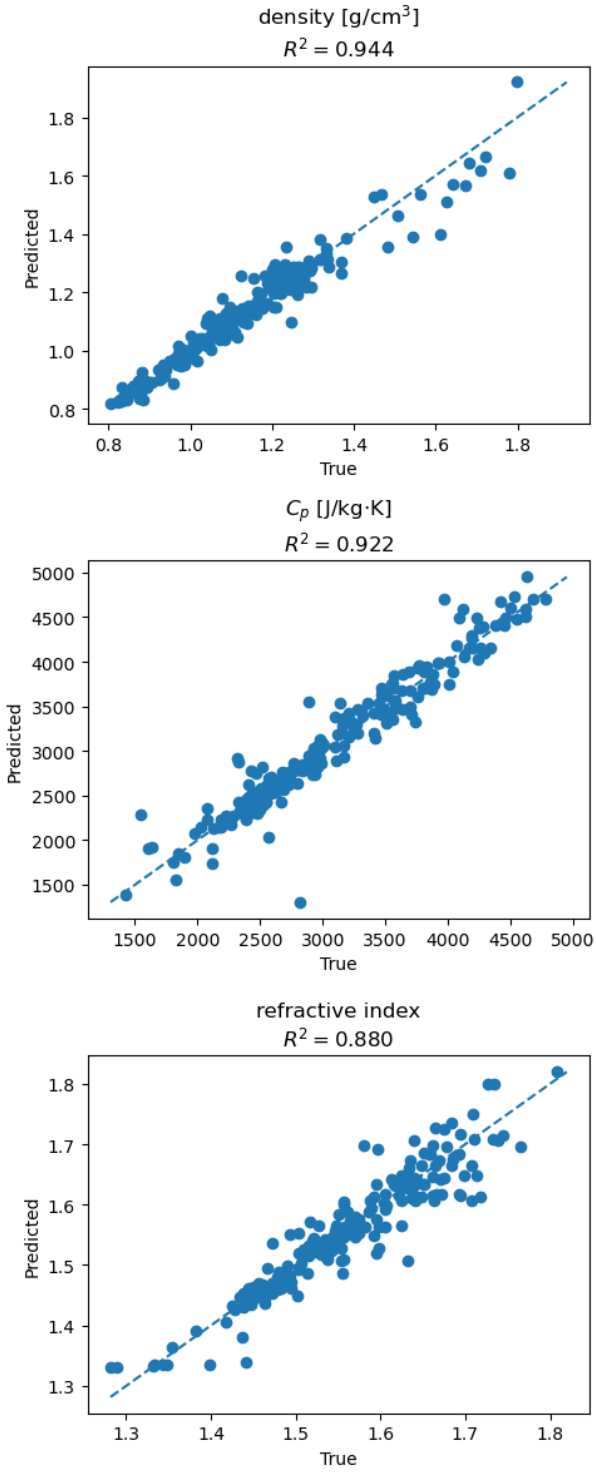

Fig. 2: Prediction on test dataset by GPT-2 pretrained with numerical tokenizer.

References

[1] Yoshihiro Hayashi, Junichiro Shiomi, Junko Morikawa, and Ryo Yoshida. Radonpy: automated physical property calculation using all-atom classical molecular dynamics simulations for polymer informatics. *npj Computational Materials*, 8:222, 2022.

[2] Shun Nanjo, Arifin, Hayato Maeda, Yoshihiro Hayashi, Kan Hatakeyama-Sato, Ryoji Himeno, Teruaki Hayakawa, and Ryo Yoshida. Spacier: on-demand polymer design with fully automated all-atom classical molecular dynamics integrated into machine learning pipelines. *npj Computational Materials*, 11:16, 2025.

[3] Xingyou Song, Oscar Li, Chansoo Lee, Bangding Yang, Daiyi Peng, Sagi Perel, and Yutian Chen. Omnipred: Language models as universal regressors. *arXiv preprint arXiv:2402.14547*, 2024.