Augmented and Programmatically Optimized LLM Prompts Reduce Chemical Hallucinations

Scott M. Reed, University of Colorado Denver

Abstract

Utilizing Large Language Models (LLMs) for handling scientific information comes with risk of the outputs not matching expectations, commonly called hallucinations. To fully utilize LLMs in research requires improving their accuracy, avoiding hallucinations, and extending their scope to research topics outside their direct training. There is also a benefit to getting the most accurate information from an LLM at the time of inference without having to create and train custom new models for each application. Here, augmented generation and machine learning driven prompt optimization are combined to extract performance improvements over base LLM function on a common chemical research task. Specifically, an LLM was used to predict the topological polar surface area (TPSA) of molecules. By using augmented generation and machine learning optimized prompts, the error in the prediction was reduced to 7.44 root mean squared error (RMSE) from 59.41 RMSE with direct calls to the same LLM.

Introduction

LLMs are opening new possibilities for leveraging natural language processing in chemistry and other scientific fields. These models can access and generate chemical information, potentially assisting researchers with tasks such as predicting molecular properties, extracting structured data from text, and even designing new molecules. However, using LLMs in chemical research comes with unique challenges. One prominent issue is "hallucination," where the model produces outputs that are confidently incorrect, often due to gaps or inconsistencies in its training data [White, 2022]. Hallucinations present a substantial obstacle in chemistry, where even minor inaccuracies can lead to significant misinterpretations in predicting molecular properties or reactions [Bran, 2024]. To fully integrate LLMs into chemical research workflows, these hallucinations must be addressed and it is critical to improve the models' ability to better handle chemical data.

Existing research efforts are exploring various ways to improve LLM performance on chemistry-specific tasks. Some groups have developed specialized models, like ChemLLM, which is trained on extensive chemical datasets to ensure it is proficient in a wide array of chemical tasks [Zhang, 2024]. This specialization helps ChemLLM perform well in chemical applications. Instruction tuning is another promising approach; models such as MolecularGPT pre-train models with Simplified Molecular Input Line Entry System (SMILES) strings connected to molecular properties to enhance few-shot learning on chemical properties, outperforming traditional models on certain tasks [Liu, 2024]. Additionally, fine-tuned models have demonstrated success in converting unstructured chemical text into structured data for reaction databases, highlighting LLMs' potential to build organized and accessible chemical knowledge bases [Pang, 2024] [Ai, 2024]. Some studies have also assessed the performance of general-purpose LLMs in chemistry-related programming tasks, such as generating code for chemical data analysis [White, 2022]. Alternatively, custom models can be created from the same transformer architecture that powers LLMs but using molecular properties as the training data.

For example, Prompt-MolOpt uses prompt engineering to improve multi-property optimization and address data scarcity issues common to this field [Wu, 2024]. This method excels in few- and zero-shot learning scenarios due to its ability to leverage single-property datasets to learn generalized causal relationships. Another area where LLMs are being used to automatically design more effective and efficient agentic systems is a novel research field called Automated Design of Agentic Systems (ADAS) [Fateen 2024].

These efforts underscore the progress being made with specialized chemical LLMs and instruction-tuned models, but they come with limitations. Developing or fine-tuning models on dedicated chemical datasets requires substantial computational and energy resources [Strubell 2022] and domain-specific expertise [Zhang, 2024]. Furthermore, once models are fine-tuned for a specific chemical application, their generalizability may suffer, and their adaptability to other domains or newly emerging chemical knowledge can become constrained [Wu, 2024]. Therefore, there is a need for time-of-prompt solutions that can enhance the accuracy of LLM predictions at inference time—without requiring extensive retraining or fine-tuning [Soylu, 2024]. Such techniques would allow LLMs to be applied to a wider range of chemical tasks, even in cases where the model's pre-existing knowledge may be incomplete or out-of-date.

Two emerging approaches that could address these limitations are Retrieval-Augmented Generation (RAG) and the Multiprompt Instruction PRoposal Optimizer (MIPRO). RAG combines a retrieval system with a generative model, enabling LLMs to dynamically fetch or calculate relevant, up-to-date information from external databases or knowledge sources [Lewis, 2021]. In the context of chemistry, RAG could draw on calculations or curated databases to supply the LLM with accurate molecular data or specific molecular properties in real time [Fateen, 2024]. This external grounding could significantly reduce the likelihood of hallucinations by ensuring that the LLM has access to precise chemical data instead of relying solely on its potentially limited training set. RAG is potentially valuable for tasks like predicting properties using group contribution methods, where relationships between molecular structure and molecular properties are complex and require detailed, accurate data that an LLM may not robustly encode [Wu, 2024].

MIPRO is a prompt optimization framework that creates and refines the LLM prompts for improved accuracy and consistency [Soylu, 2024]. MIPRO uses an LLM to generate additional instructions to add to the prompt and then selects few-shot examples that illustrate successful executions of the given task, optimizing the selection of both using a PyTorch powered ML framework [Opsahl-Ong, 2024]. MIPRO can bootstrap examples from training data and dynamically generate instruction candidates to provide structured, task-specific guidance [Zhang, 2024]. Through Bayesian optimization, MIPRO iteratively identifies the optimal combination of examples and instructions, evaluated against a user-generated quantitative metric. This prompt refinement reduces hallucinations by ensuring that the LLM has a clear and relevant framework for understanding underlying data, without the need for creating or fine-tuning a model for a specialized application [Wu, 2024].

TPSA is used as a molecular descriptor in drug research because it can efficiently predict a drug's ability to passively cross biological membranes, such as the intestinal lining or the blood-brain barrier [Pajouhesh, 2005]. This efficiency is crucial in early drug discovery stages, where researchers need to evaluate a large number of potential drug candidates. Several studies have shown that TPSA correlates well with drug permeability [Ertl, 2000]. For instance, drugs that are readily absorbed from the gut or those that can penetrate the central nervous system typically have lower TPSA values [Zhang, 2015]. TPSA has also been used in a model that predicts drug exposure in pregnant women and their fetuses. This model relies on a "permeability-limited placenta model" that simulates drug transfer between the mother and fetus [Zhang, 2015].

Together, RAG and MIPRO present a powerful solution for improving LLM performance. RAG addresses the issue of outdated or incomplete information by grounding the LLM’s responses in current, high-quality data sources, ensuring that predictions are accurate and contextually relevant. MIPRO complements this by optimizing the prompt structure, allowing the LLM to interpret and utilize retrieved data more effectively through well-designed instructions and examples. Here, as an example of this approach, I describe a method for predicting TPSA that combines RAG and MIPRO using a commercially available LLM, ChatGPT-4o-mini. In tandem, these approaches enabled the LLM to make accurate, data-driven predictions at inference time, enhancing its reliability without fine tuning the weights of the base model. This approach reduced the root mean squared error (RMSE) from 59.41 for prediction using the GPT-4o-mini LLM directly to 7.44 RMSE when MIPRO and RAG were employed for predictions on a set of random molecules. The individual contribution of the various elements of this approach is described below.

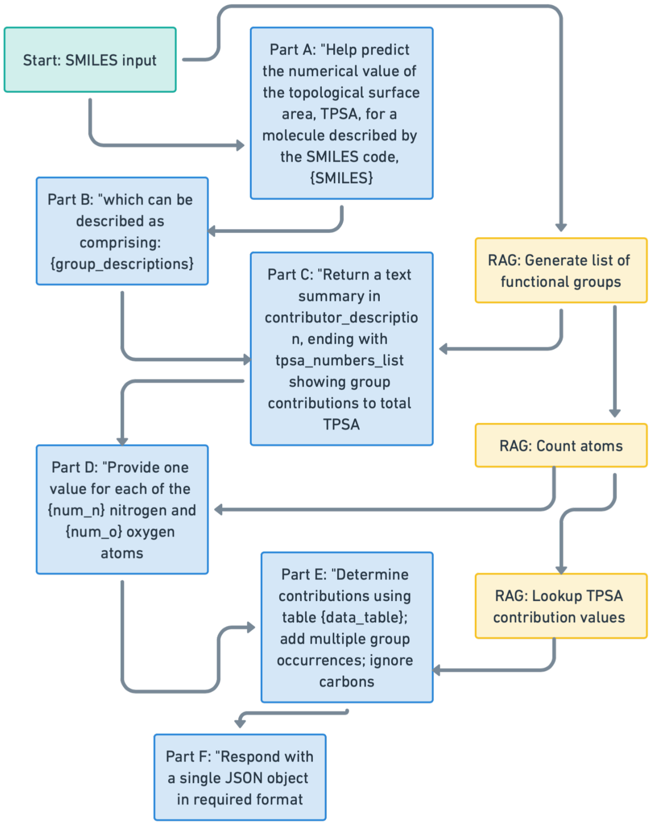

Start: SMILES input
Part A: "Help predict the numerical value of the topological surface area, TPSA, for a molecule described by the SMILES code, {SMILES}
Part B: "which can be described as comprising: {group_descriptions}
Part C: "Return a text summary in contributor_description, ending with tpsa_numbers_list showing group contributions to total TPSA
RAG: Generate list of functional groups
Part D: "Provide one value for each of the {num_n} nitrogen and {num_o} oxygen atoms
RAG: Count atoms
Part E: "Determine contributions using table {data_table}; add multiple group occurrences; ignore carbons
RAG: Lookup TPSA contribution values
Part F: "Respond with a single JSON object in required format

Figure 1. Process for generating prompt components (blue) for tpsa model from input SMILES (green) and RAG components (yellow).

Material and methods

Data Preparation

Molecular data were acquired from PubChem by querying random compound identifiers and fetching properties through the PubChem PUG-REST API. A dataset was constructed by binning TPSA values in intervals of 5 units, with 20 molecules per bin to ensure even distribution across TPSA values and provide a robust dataset for prompt tuning. Since PubChem defines TPSA as "a simple method - only N and O are considered," [https://pubchem.ncbi.nlm.nih.gov/docs/glossary accessed on Nov 12, 2024.] only molecules with C, N, O, and H were included and if the N and O functional groups could not be mapped to one of the specified functional groups [Ertl 2000], they were excluded. Bins were populated by randomly sampling molecules from PubChem until each TPSA interval had 20 molecules. RDKit was used to parse SMARTS patterns, generating a list of functional groups. SMARTS patterns were loaded and iteratively applied to each SMILES string, with RDKit identifying the presence of targeted functional groups in each molecule. These functional group assignments were then linked to TPSA contributions using lookup data containing TPSA values associated with each group.

To focus on drug-like molecules, SMILES codes with more than 10 hydrogen bond acceptors or more than 5 hydrogen bond donors were removed. Additionally, molecules with a mass greater than 500 were filtered out, further aligning the dataset with criteria typically used for drug-like compound properties. Finally, molecules with a non-zero charge were excluded to maintain focus on neutral compounds. The training set contained 30 structured examples from this list for selecting bootstrap examples from, while the validation set contained a second set of 30 that were used to validate prompt performance.

Structuring Examples for Training

DSPy examples serve as modular, query-answer pairs that allowed standardization of data inputs and generated a comprehensive dataset spanning a wide range of TPSA values. This dataset was balanced across TPSA intervals to prevent biases toward certain values and ensure that the LLM was exposed to a representative set of molecular features. A scaffold split was performed to ensure that the train or test sets would contain examples of any scaffolds that repeated across the data. The examples were then loaded into the LLM program as a structured dataset, where each Example provided the model with a consistent input-output relationship.

Prompt Optimization

GPT-4-o-mini was used as the model for generating and testing prompts, ensuring that both prompt generation and task completion maintained consistent model behavior. GPT-4o-mini which has a reduced model size compared to GPT-4o was used here in part to minimize the risk that prior training data would contain direct answers to the questions being asked. The reduced parameter size means these direct connections are less likely. The most recent version of MIPRO, MIPROv2 from the DSPy package was used. 10 few-shot example sets were proposed during the optimization. By generating 10 sets, MIPRO can experiment with a range of examples, allowing it to assess which examples best aid the model in reducing TPSA prediction errors. An initial temperature of 1.2 was used. This increases prompt diversity at the start. This helps MIPRO to explore various prompt combinations early on, with a controlled decrease in diversity over time for convergence.

25 trials were run, allowing MIPRO to iteratively refine prompts based on validation performance. Each trial generates a new prompt set, and Bayesian Optimization identifies which sets perform best. Minibatch evaluation was performed in batches of 5 examples, enabling efficient prompt evaluation in each trial. This approach allows for broader prompt testing within the given trial limit of 30. The number of few-shot and labeled examples in each prompt was set to a maximum of 8, ensuring manageable prompt length and optimizing example diversity without overwhelming the model with too many examples at once. After every 5 minibatches, a full evaluation on the validation set was performed. This periodic full evaluation providef a more stable performance benchmark, allowing the Bayesian optimizer to adjust prompt selection based on more reliable performance data.

Evaluation Metric

A custom metric was used to calculate the absolute error between the LLM predicted TPSA value and the true TPSA value, using this difference to guide prompt and example selection across bootstrap example selection and prompt optimization. During bootstrap example selection, the metric assesses the accuracy of candidate few-shot examples generated from the training dataset. A threshold-based approach was used, retaining only examples where the absolute error was below 20. This threshold ensures that the examples selected for bootstrapping are reliable representations of a good TPSA prediction, forming a solid foundation for the few-shot examples used in prompt optimization. In prompt optimization, the metric guides Bayesian Optimization by continuously measuring the accuracy of different prompt configurations. At each trial, the effectiveness of a prompt is evaluated by calculating the negative absolute error across a batch of examples. Additionally, every few minibatches the entire validation set was evaluated to confirm that the current prompt configuration performs well on a broader set of examples, enhancing stability and reducing noise in prompt selection. By calculating negative absolute error between predicted and actual TPSA values, this metric guides the optimizer towards more accurate prompt selections.

The TPSA predictor is derived from DSPy Module object and utilizes the TypedPredictor program to ensure responses with correct formatting. The predictor encapsulates the logic for preparing, formatting, and training the model on prompt-optimized TPSA prediction tasks. It utilizes a structured prompt that can integrate molecular descriptions, functional group data, and specific atom counts. These modules include 1) Describing Molecular Functional Groups: The method first calls describe_molecule with the SMILES code. This function returns an assignment of functional groups based on predefined SMARTS patterns.

Augmented Generation

The prompts are generated in segments that are removed selectively during the ablation study. The prompt segments include: 1) Functional Group Information: A function was created using rdkit to provide a list of functional groups present in the smiles code as matched to the list of TPSA contributors [Ertl 2000], 2) Atom Counts: identifies the number of nitrogen and oxygen atoms in the molecule. 3) The total atom count is used to generate specific instructions on how each atom's presence should impact the TPSA value. 4) Data from the published group contribution table to the TPSA for each functional group present 5) Details that specify the response format, ensuring the LLM outputs a JSON object with a list of TPSA values. The predicted TPSA contributions are summed to avoid math hallucinations [Rawte 2023] and to provide a single TPSA value.

Results

The effectiveness of structured prompt optimization using RAG and MIPRO (tpsa model, Figure 1) was compared to a basic prompt (direct model) for predicting the TPSA of a set of 140 molecules. The direct model, which uses a simple, non-augmented prompt without RAG or MIPRO optimizations, results in a mean RMSE of 59.41, with predictions showing little alignment to the actual TPSA values. The prompt used was "Predict the numerical value of the topological surface area, TPSA, for a molecule described by the SMILES code, {*molecule*}," where *molecule* was one SMILES code selected from a list. This basic prompt leads to poor model performance, as the LLM struggles to reliably relate molecular structure to TPSA without the additional context provided in the optimized prompt. SMILES codes are common but if the training data did not include the specific property connected to that specific form of the SMILES code, as appears to be the case, the LLM cannot infer what the values should be. (Figure 2A). The tpsa model, incorporating RAG and optimized with MIPRO's prompt structuring and few-shot example selection, achieves an RMSE of 7.44, with most predictions closely matching the calculated values obtained from PubChem. This suggests that incorporating functional group details and other contextual information and optimizing prompts through MIPRO significantly enhances prediction accuracy. For the tpsa model a multi-part prompt structure (Figure 2B) was used that incorporated RAG components as well as text designed to ensure the response of the LLM followed the request for typed format of a list of float values which were then summed to get the predicted TPSA value. This prompt was used in the MIPRO process which produced examples and a data description that were appended to the prompt at inference. The outliers that had a predicted TPSA > 1 different from the calculated TPSA (supporting info, Figure S1) tended to have longer lists of functional groups (6.4 vs 4.6 mean) and more nitrogen and oxygen atoms suggesting that the more complicated molecules were harder to predict.

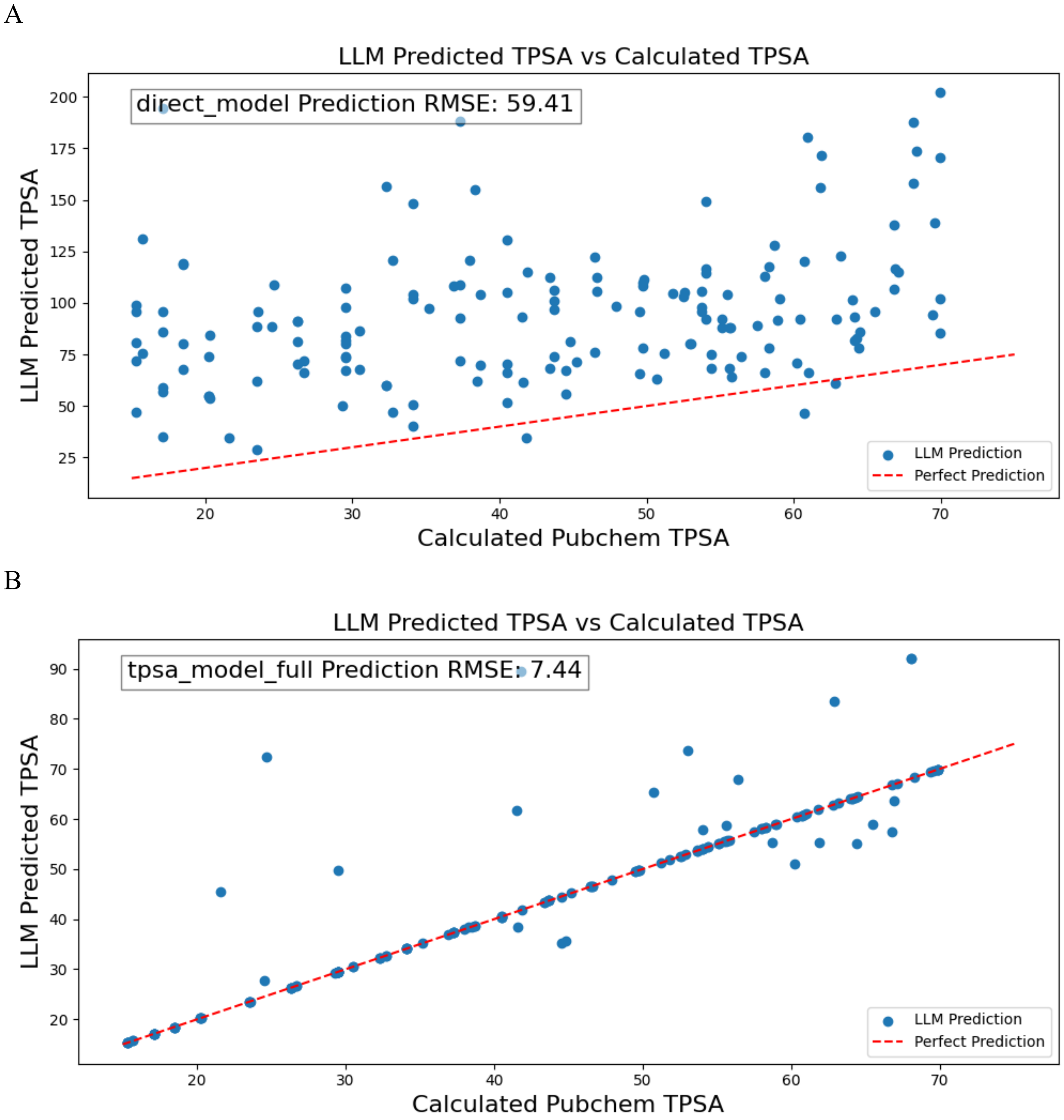
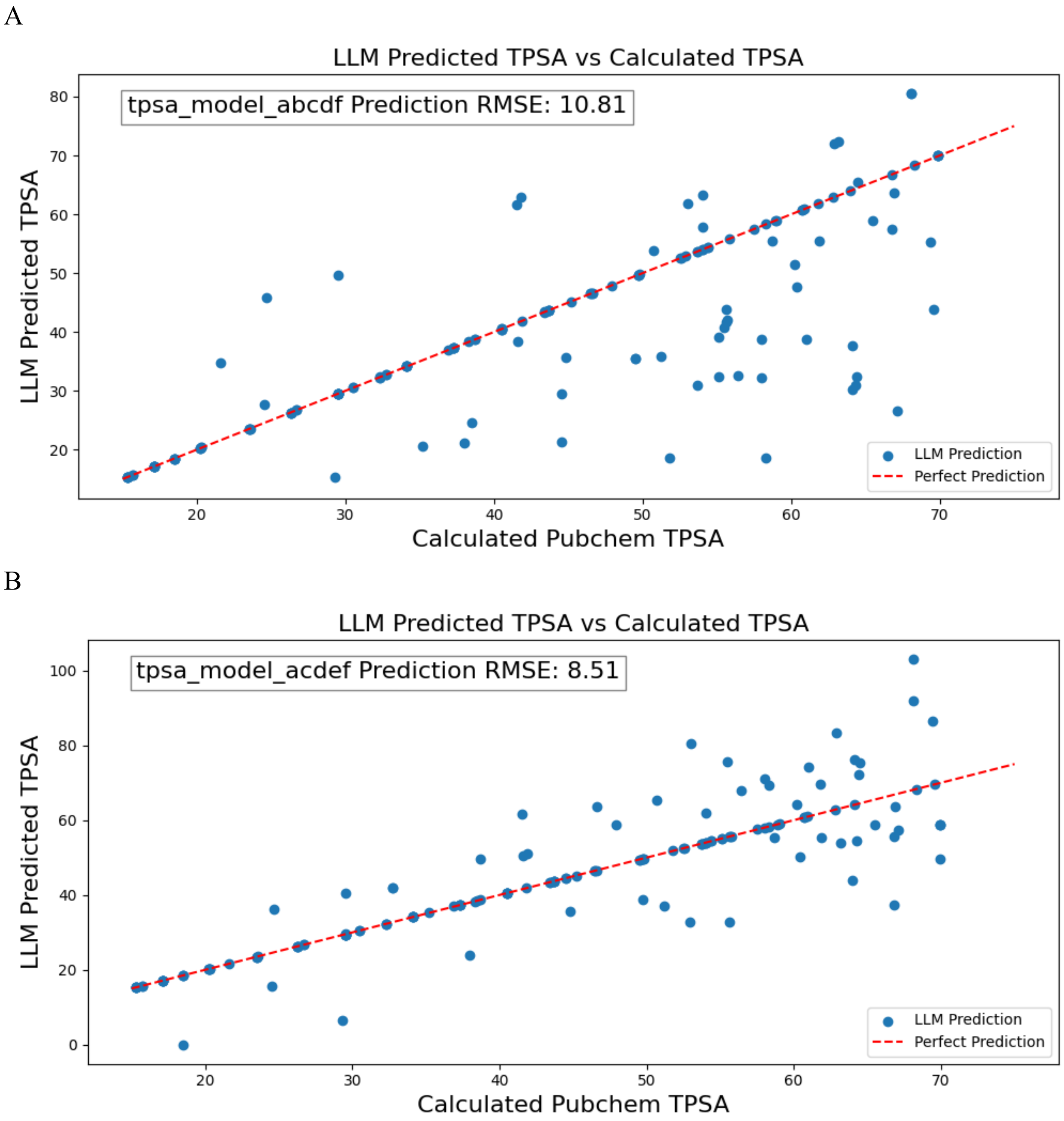


Figure 2. A) direct LLM prediction of TPSA values for a set of randomly selected SMILES codes from PubChem using GPT-4o-mini. B) The same molecules predicted by the full model including RAG and MIPRO components.


A
LLM Predicted TPSA vs Calculated TPSA
tpsa_model_abcdf Prediction RMSE: 10.81
LLM Predicted TPSA
80
70
60
50
40
30
20
Calculated Pubchem TPSA
20
30
40
50
60
70
LLM Prediction
Perfect Prediction
B
LLM Predicted TPSA vs Calculated TPSA
tpsa_model_acdef Prediction RMSE: 8.51
LLM Predicted TPSA
100
80
60
40
20
0
Calculated Pubchem TPSA
20
30
40
50
60
70
LLM Prediction
Perfect Prediction

C

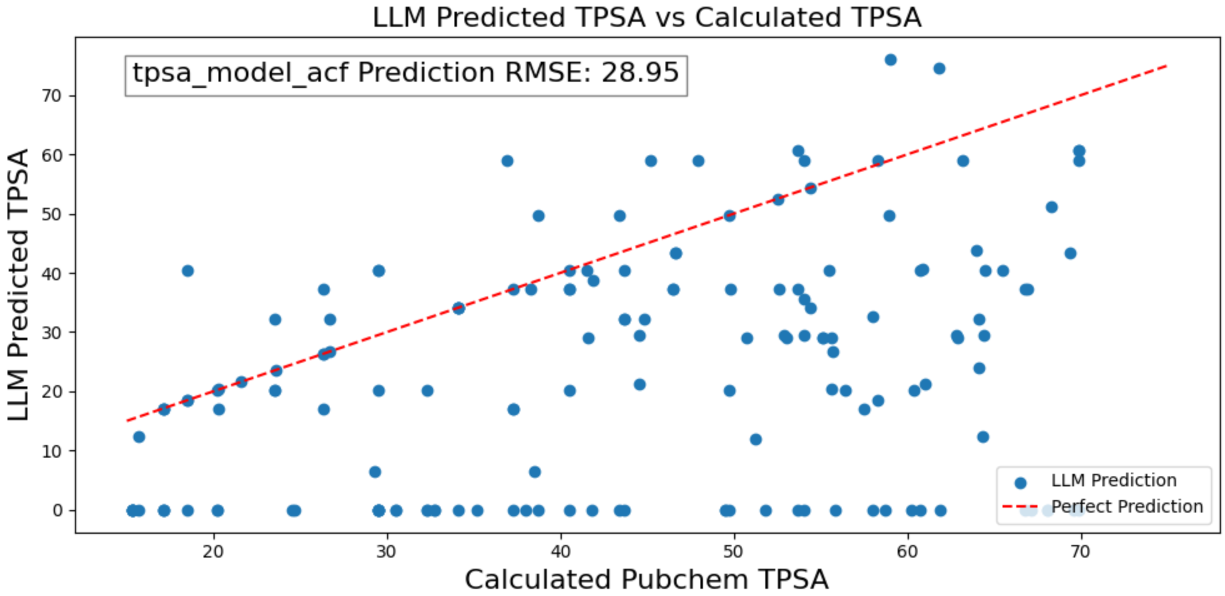


Figure 3. LLM prediction of TPSA values for a set of randomly selected SMILES codes from PubChem using GPT-4o-mini, excluding some RAG components, either A) the list of functional groups in the molecule, B) the table of TPSA contributions that match to the groups, and C) both of these pieces omitted. GPT-4o-mini used in each case.

Next, different components were removed from the complete model to assess the impact each component had on the accuracy improvement over the direct LLM call. The tpsa_model_abcdf configuration excludes the RAG component that contains tabular TPSA contribution data [Ertl 2000] used for calculating group contributions additively for TPSA (Figure 3A). This omission results in a mean RMSE of 10.81. While the RMSE is slightly higher than the fully optimized model, most data points still cluster along the perfect prediction line, with most deviations at higher TPSA values. This suggests that the tabular TPSA data provide some accuracy benefit but are not critical to the model's overall performance. The outliers (supporting info, Figure S2) lean toward the more complex structures with increased heteroatom counts and number of functional groups.

The tpsa_model_acdef omits only the RAG step that provides a list of functional groups present in the SMILES to the LLM. With a mean RMSE of 8.51, this configuration shows only a slight decline in accuracy compared to the fully optimized model, with good alignment between predicted and actual TPSA values (Figure 3B). This result implies that while functional group descriptions add value in helping with SMILES interpretation, the model can still achieve reasonably accurate predictions without them, correctly identifying functional groups from the provided SMILES. The outliers (supporting info, Figure S3) again are more complex with some molecules repeating between this and the prior list.

The tpsa_model_acf, shows a substantial increase in mean RMSE to 28.95 after removing both the functional group list and specific atom counts (Figure 3C). The responses included many zero values for the predicted TPSA when the RAG portions with information about the number and types of functional groups are removed. Without these critical details, predictions become widely dispersed from actual values. This configuration underscores the importance of functional group and functional group information for minimizing hallucinations and achieving reliable TPSA predictions. The outliers (supporting info, Figure S4) contain many of the same molecules as the individual RAG removals as well as some new ones.

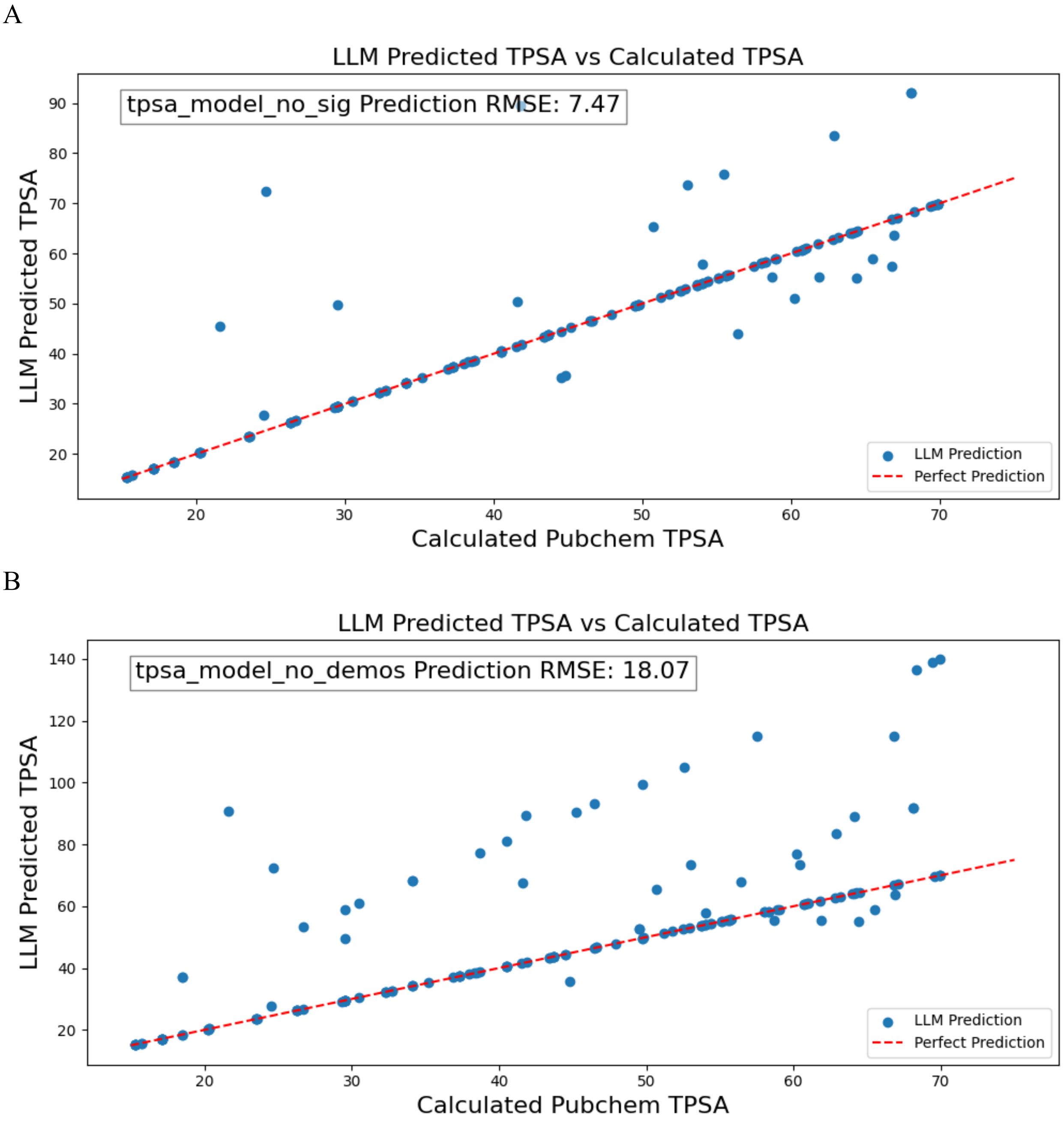


Figure 4. LLM prediction of TPSA values for a set of randomly selected SMILES codes from PubChem using GPT-4o-mini, excluding A) the signature developed by MIPRO, B) the bootstrapped examples produced by MIPRO.

Next, the text added by the MIPRO optimization was iteratively removed from the full model to assess the contribution of MIPRO to the improvement of the overall model. When the description of the dataset (termed signature in DSPy) was removed (Figure 4A) the mean RMSE value increased only slightly to 7.47. In contrast, when the bootstrapped examples were removed, the RMSE increased to 18.07 (Figure 4B). While some predictions remained close to the ideal value, many did not, including 15 values that were exactly doubled over the actual values, a hallucination not observed in the direct prediction.

Discussion

This study provided a simple example of a molecular property prediction that allowed a detailed examination of strategies to reduce LLM hallucinations in scientific applications. The results demonstrate the effectiveness of both RAG and MIPRO individually and in combination to improve the accuracy and reliability of LLMs in predicting molecular properties, a critical aspect of drug research. By augmenting LLMs with both external data retrieval and optimized prompt structures, we observed a significant reduction in prediction errors. Specifically, the fully optimized model achieved an RMSE of 7.44, closely aligning with calculated TPSA values and outperforming models that used only a simple prompt or incomplete prompt components.

MIPRO iteratively identifies the optimal combination of examples and instructions, creating a prompt that enables the model to consider functional group contributions, functional group details, and additive rules when making predictions. The addition of MIPRO's optimized prompts allowed the model to better interpret molecular structure and contextual details, such as functional group contributions which are essential for accurate predictions. Our ablation studies showed that removing specific prompt components led to increased error rates, confirming the importance of each element in minimizing model hallucinations. For instance, omitting functional group descriptions or atom counts resulted in poorer alignment, with RMSE rising to 28.95 when both elements were removed. These findings underscore the necessity of detailed molecular context in LLM prompts, when property predictions depend on molecular features.

By integrating a retrieval step that generates relevant molecular properties and functional group information, this RAG mitigates the risk of hallucinations. This approach addresses the limitations of relying solely on static training data, which may be unavailable for all possible inputs or insufficiently detailed for specialized tasks. By integrating RAG and MIPRO, the LLM's applicability to the chemical task of TPSA prediction was improved, without retraining or fine-tuning the LLM. These results suggest that RAG and MIPRO can significantly improve the utility of general-purpose LLMs in chemical and other scientific research, providing a flexible, scalable solution that enhances prediction accuracy and contextual relevance. This combined approach offers a promising pathway for leveraging LLMs in chemistry and other fields where accurate, context-aware data interpretation is essential. By allowing the model to retrieve relevant information for each query, RAG helps ensure that its predictions are rooted in reliable data.

By combining RAG's data-driven retrieval with MIPRO's prompt optimization, LLMs can be transformed into more accurate and versatile tools for chemical research, capable of delivering reliable predictions even in complex or unfamiliar contexts [Fateen, 2024; Soylu, 2024]. This approach holds promise not only for chemistry but also for other scientific domains that require precise, contextually informed data interpretation [Bran, 2024]. Together, RAG and MIPRO can enhance the utility of general-purpose LLMs across a wide range of research applications, reducing the need for specialized models and allowing researchers to leverage LLM technology with greater flexibility and accuracy [Wu, 2024].

Training or fine-tuning models [Pang, 2024] with up-to-date information is another powerful approach but comes with drawbacks. Fine-tuning requires significant advance work to prepare a model tailored to a specific need. In contrast, approaches that can be performed at inference time offer the advantage of being applicable to any model without retraining the weights, thereby preserving generalizability. This combination could be especially useful in drug discovery, where accurate molecular property predictions are crucial for assessing drug permeability and potential efficacy early in the development pipeline.

Conclusions

As LLMs and their training data grow in size, their capabilities can seem limitless, however, they cannot be trained on data that does not exist yet. The approach described here takes an LLM incapable of a specific molecular task and makes it substantially more capable through augmented generation and prompt optimization. This approach could allow LLMs to be used as research assistants even when handling data outside of their initial training while maintaining the utility of LLMs in handling language.

Acknowledgements

Financial support from NIH 1R15GM151726-01 is acknowledged. Kenny Lipkowitz is thanked for his critical reading of this manuscript.

Data and Software Availability statement

All code used in this study will be made available in a public repository upon publication.

1. Rawte, V., Sheth, A. & Das, A. A Survey of Hallucination in Large Foundation Models. Preprint at http://arxiv.org/abs/2309.05922 (2023)
2. M. Bran, A., Cox, S., Schilter, O., Baldassari, C., White, A. D. & Schwaller, P. Augmenting large language models with chemistry tools. *Nat Mach Intell* **6,** 525–535 (2024).
3. Hu, S., Lu, C. & Clune, J. Automated Design of Agentic Systems. Preprint at http://arxiv.org/abs/2408.08435 (2024)
4. Fateen, M., Wang, B. & Mine, T. Beyond Scores: A Modular RAG-Based System for Automatic Short Answer Scoring with Feedback. Preprint at http://arxiv.org/abs/2409.20042 (2024)
5. Zhang, D., Liu, W., Tan, Q., Chen, J., Yan, H., Yan, Y., Li, J., Huang, W., Yue, X., Ouyang, W., Zhou, D., Zhang, S., Su, M., Zhong, S. & Li, Y. ChemLLM: A Chemical Large Language Model.
6. White, A. D., Hocky, G. M., Gandhi, H. A., Ansari, M., Cox, S., Wellawatte, G. P., Sasmal, S., Yang, Z., Liu, K., Singh, Y. & Peña Ccoa, W. J. Do large language models know chemistry? Preprint at https://doi.org/10.26434/chemrxiv-2022-3md3n (2022)

7. Strubell, E., Ganesh, A. & McCallum, A. Energy and Policy Considerations for Deep Learning in NLP. in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics* 3645–3650 (Association for Computational Linguistics, 2019). doi:10.18653/v1/P19-1355

8. Ai, Q., Meng, F., Shi, J., Pelkie, B. & Coley, C. W. Extracting Structured Data from Organic Synthesis Procedures Using a Fine-Tuned Large Language Model. Preprint at https://doi.org/10.26434/chemrxiv-2024-979fz (2024)

9. Ertl, P., Rohde, B. & Selzer, P. Fast Calculation of Molecular Polar Surface Area as a Sum of Fragment-Based Contributions and Its Application to the Prediction of Drug Transport Properties. *J. Med. Chem.* **43,** 3714–3717 (2000).

10. Soylu, D., Potts, C. & Khattab, O. Fine-Tuning and Prompt Optimization: Two Great Steps that Work Better Together. Preprint at http://arxiv.org/abs/2407.10930 (2024)

11. Zhang, W., Wang, Q., Kong, X., Xiong, J., Ni, S., Cao, D., Niu, B., Chen, M., Li, Y., Zhang, R., Wang, Y., Zhang, L., Li, X., Xiong, Z., Shi, Q., Huang, Z., Fu, Z. & Zheng, M. Fine-tuning large language models for chemical text mining. *Chem. Sci.* **15,** 10600–10611 (2024).

12. Yang, C., Wang, X., Lu, Y., Liu, H., Le, Q. V., Zhou, D. & Chen, X. Large Language Models as Optimizers. Preprint at http://arxiv.org/abs/2309.03409 (2024)

13. Wu, Z., Zhang, O., Wang, X., Fu, L., Zhao, H., Wang, J., Du, H., Jiang, D., Deng, Y., Cao, D., Hsieh, C.-Y. & Hou, T. Leveraging language model for advanced multiproperty molecular optimization via prompt engineering. *Nat Mach Intell* (2024). doi:10.1038/s42256-024-00916-5

14. Pajouhesh, H. & Lenz, G. R. Medicinal chemical properties of successful central nervous system drugs. *Neurotherapeutics* **2,** 541–553 (2005).

15. Liu, Y., Ding, S., Zhou, S., Fan, W. & Tan, Q. MolecularGPT: Open Large Language Model (LLM) for Few-Shot Molecular Property Prediction. Preprint at http://arxiv.org/abs/2406.12950 (2024)

16. Opsahl-Ong, K., Ryan, M. J., Purtell, J., Broman, D., Potts, C., Zaharia, M. & Khattab, O. Optimizing Instructions and Demonstrations for Multi-Stage Language Model Programs. Preprint at http://arxiv.org/abs/2406.11695 (2024)

17. Zhang, Y.-H., Xia, Z.-N., Yan, L. & Liu, S.-S. Prediction of Placental Barrier Permeability: A Model Based on Partial Least Squares Variable Selection Procedure. *Molecules* **20,** 8270–8286 (2015).

18. Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W., Rocktäschel, T., Riedel, S. & Kiela, D. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. Preprint at http://arxiv.org/abs/2005.11401 (2021)

19. Lu, C., Lu, C., Lange, R. T., Foerster, J., Clune, J. & Ha, D. The AI Scientist: Towards Fully Automated Open-Ended Scientific Discovery. Preprint at http://arxiv.org/abs/2408.06292 (2024)

20. Pang, J., Pine, A. W. R. & Sulemana, A. Using natural language processing (NLP)-inspired molecular embedding approach to predict Hansen solubility parameters. *Digital Discovery* **3,** 145–154 (2024).