Van der Waals Equation of State Coefficient Predictor

Design Report
COSC 402
Blake Milstead, Turner Heath, Karim Chmayssani
February 14, 2025

Executive Summary

This report details the design of a machine learning tool aimed at predicting the Van der Waals constants, a and b, for various molecules. These constants are essential for understanding real gas behavior, impacting fields such as chemical engineering and pharmaceuticals. Traditional methods for determining these constants are often labor-intensive and time-consuming. Our proposed solution integrates Graph Neural Networks (GNNs), and our custom PKAN model. The PKAN model is an integration of a Physics-Informed Neural Network (PINN), and a Kolmogorov-Arnold Network (KAN). Combining physics-based constraints, as well as new degrees of non-linearity, something necessary for modeling molecular behavior, we believe this architecture will suit the problem very well. Representing molecules as Coulomb matrices, described by graphs, we encode the molecule to a fixed length vector for the PKAN model to predict a and b with. The envisioned user-friendly software application will allow users to input molecular structures and receive predicted a and b constants, complete with confidence intervals and visualizations. Target users include chemical engineers, pharmaceutical researchers, and academic educators. The design validation plan encompasses usability testing, iterative refinement, and evaluation metrics such as prediction and model accuracy. The final deliverables will comprise the predictive model, software application, comprehensive documentation, validation report, source code repository, deployment package, and training materials, providing a comprehensive solution that ensures ease of use, transparency, and adaptability for various user requirements.

Contents

List of Figures

List of Tables

Chapter 1

Problem Definition & Background

1.1 Problem Statement

The accurate prediction of the Real Gas constants, a and b, is critical for understanding and modeling molecular behavior under non-ideal conditions. These constants, which account for intermolecular forces and molecular volume respectively, are essential for engineering, chemistry, and various industrial applications. Currently, determining these constants relies heavily on experimental methods, which can be time-consuming, resource-intensive, and infeasible for large-scale applications. A predictive machine learning-based approach using molecular embeddings could significantly improve efficiency and scalability.

1.2 Background and Context

Accurately modeling the behavior of real gases is a longstanding challenge in physical chemistry and thermodynamics. While the Ideal Gas Law offers a simplified estimate, real-world application necessitates the use of the Van der Waals equation, where the a and b constants provide critical corrections. Industries such as petrochemicals, pharmaceuticals, and material sciences often require precise modeling of gas behavior for applications such as reaction engineering, material synthesis, and environmental control. Current experimental methods

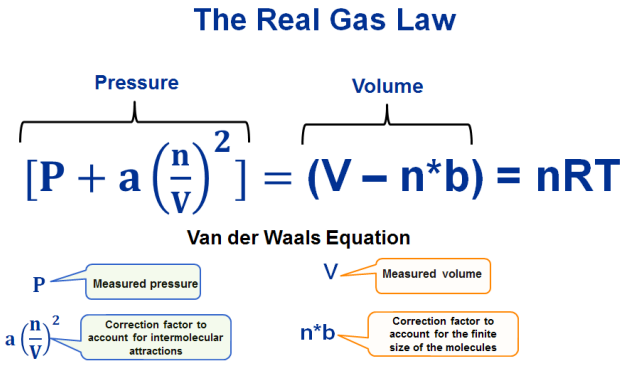

Figure 1.1: Real Gas Law Terms Explained

to determine these constants are laborious and limited in scope, making them unsuitable for rapid prototyping or computational simulations involving large molecular datasets.

A machine learning approach leveraging molecular embeddings offers the potential to revolutionize this process. By mapping molecular structures to high-dimensional feature spaces, embeddings can encode complex molecular properties, enabling rapid and accurate predictions of the Real Gas constants.

1.3 Application Areas

- **Chemical Engineers:** Require precise gas behavior modeling for designing processes and reactors.
- **Materials Science:** Predicting properties of new materials, such as thermal stability or conductivity, to optimize performance in applications like electronics or energy storage.
- **Pharmaceutical Researchers:** Use molecular property predictions for drug development.
- **Software Developers in Simulation Tools:** Need efficient algorithms for integrating predictive capabilities into modeling software.

1.4 Underlying Theory

The Van der Waals equation extends the Ideal Gas Law to Real Gases, including:

- a**:** Quantifies intermolecular attractions, critical for understanding liquefaction and condensation phenomena.
- b**:** Represents the volume of a molecule to account for their physical size.

Molecular embeddings encode chemical structures into numerical representations, capturing properties such as bond types, electron distribution, and geometric configurations. Techniques such as RDKit-based descriptors, graph neural networks, and cheminformatics algorithms are foundational to this approach. Figure 1.2 shows the relationship between a and b, giving us proof that a correlation exists and can be learned.

1.5 Prior Work

Previous work to find the Real Gas constants includes:

- Experimental determinations using high-precision instruments [2].
- Thermodynamic models based on molecular theory and statistical mechanics.
- Machine learning models predicting related properties, such as boiling or critical points.

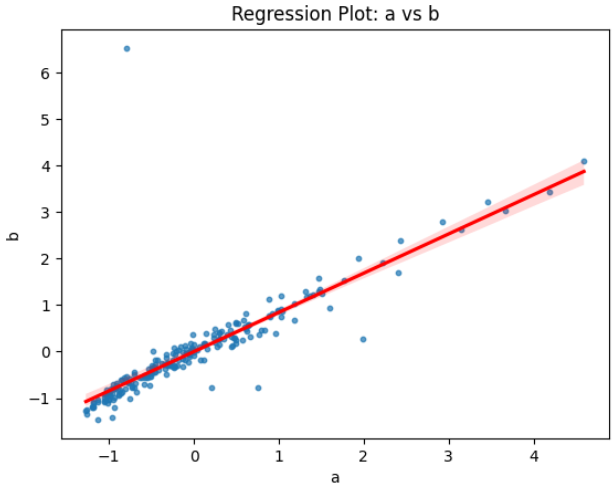

Figure 1.2: A vs B Regression

- Computational chemistry methods such as quantum chemistry and molecular dynamics [3].

1.6 Unaddressed Needs

Current software and tools are either too domain-specific or computationally expensive. This project aims to produce accurate predictions of the Real Gas constants, while remaining lightweight and flexible enough for reapplication.

Chapter 2

Requirement Specification

2.1 Overview

The primary objective of this project is to develop a scalable and efficient prediction pipeline integrated with a user-friendly graphical user interface (GUI). The pipeline will preprocess molecular data, generate embeddings, and use a deep learning model to predict Real Gas properties. The GUI will enable users to input molecular structures, view predictions, and interpret results interactively.

2.2 Functional Requirements

- **Data Input and Processing:**
  - Support for manual molecule input via the GUI with real-time validation of SMILES strings.
  - Automatic computation of molecular descriptors and embeddings using RDKit and PubChemPy.
- **Prediction Model:**
  - The system will predict the Real Gas constants a and b.
  - Predictions will be computed using a pre-trained deep learning model.
  - Provide real-time predictions for single-molecule inputs via the GUI and batch predictions for uploaded datasets.
- **Graphical User Interface (GUI):**
  - A user-friendly interface for inputting molecules, visualizing predictions, and exploring properties.
  - Interactive visualization of input molecules and predicted properties (e.g., tables, plots, or 3D molecule views).

  - Intuitive workflows for both novice and expert users, with minimal technical barriers.

- **Visualization and Reporting:**
  - Scatterplots comparing true vs. predicted values for batch datasets.
  - Exportable results in standard formats (e.g., CSV, JSON).
  - Provide confidence intervals or error metrics for each prediction to aid in decision-making.

2.3 Non-Functional Requirements

- **Performance:**
  - Ensure low prediction latency for single molecule inputs and reasonable processing times for batch predictions.
- **Scalability:**
  - Support for model extensions and other predictive pathways.
- **Reliability:**
  - Ensure the system gracefully handles invalid inputs and provides meaningful error messages.
- **Usability:**
  - Design the GUI to be platform-independent, accessible via major operating systems (Windows, macOS, Linux). Must be user-friendly and easy to use.

2.4 Metrics and Targets

- **Prediction Accuracy:** Achieve a combined mean squared error (MSE) of less than 0.1 and an R^2 score greater than 0.9 on validation datasets.
- **Processing Speed:** Complete model training in time proportional to size of dataset.
- **GUI Usability:** Receive positive usability feedback in user testing.

Chapter 3

Technical Approach

3.1 Overview

Our design plan aims to develop an advanced molecular property prediction system integrating multiple machine learning approaches with domain-specific molecular representations and encodings. Figure A.1 shows the viability of using embeddings to make accurate predictions. This simple proof of concept allows us to move forward with confidence. The pipeline will include molecular data preprocessing, various encoding methods, and multiple predictive models. These models include Graph Neural Networks, as well as our custom PKAN model. Using physics-informed constraints, as well as non-linearity in activation functions, it is designed to accurately capture physical information about the molecule and find non-linear patterns that are common in molecular behavior. The system will also incorporate a graphical user interface (GUI) to facilitate user interaction and accessibility.

3.2 System Architecture

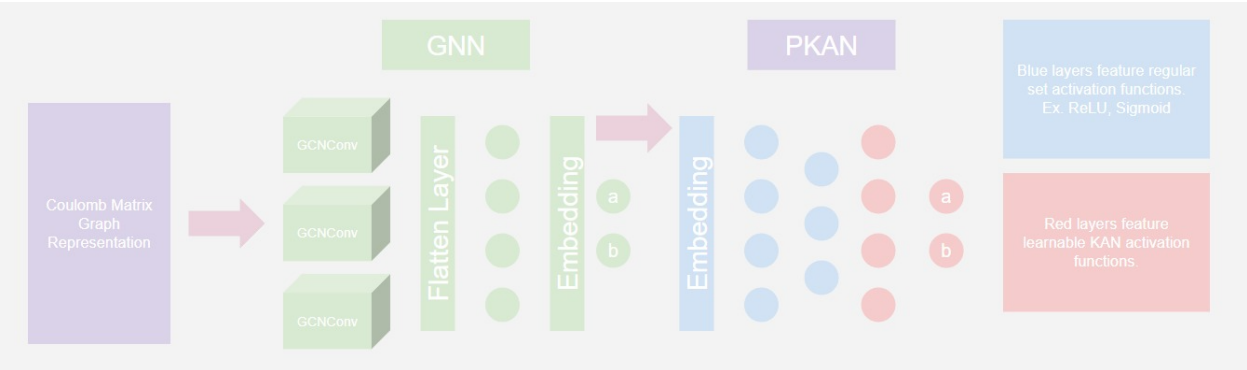

Figure 3.1: The Full EOS Predictor Pipeline

The pipeline will be fully developed in Python and consists of four main sections:

- **Data Preprocessing:** Given the molecular dataset, we generate Coulomb adjacency

matrices. These are generated from the SMILES string, which will be created from the graphical model created by the user.

- **GNN:** The GNN is used to take the Coulomb adjacency matrix representation of a molecule and create a fixed length vector embedding for input into the PKAN model.
- **PKAN:** The true predictive model, the PKAN uses a fixed length embedding to predict a and b. Structurally, it is very similar to a standard PINN model, using fully connected (FC) layers with set activation functions, as well as incorporating the differential equation for the system within the model loss. It differs in the output layers, using a collection of learnable activation functions from the KAN model.
- **Graphical User Interface:** The GUI aims to serve as a landing pad for users, allowing them to create and upload molecules for prediction.

3.3 Functional Decomposition

To design the pipeline as presented above, we have decomposed each of the four sections of our model.

3.3.1 Data Collection and Preprocessing

The data collection and processing section of the pipeline carries many responsibilities, namely:

- **Molecular Data Acquisition**: Gather molecular structures and their corresponding a and b constants from the dataset.
- **Data Cleaning**: Address missing values, remove duplicates, and standardize molecular representations to ensure data quality.
- **Feature Extraction**: Generate molecular embeddings using techniques such as molecular fingerprints, graph-based representations, and sequence-based embeddings [4].
- **Train and Test Split**: Most important to the integrity of our design, we must be very intentional regarding data leakage between training and test data. As the models are trained, we must ensure that they never see any of the test data, as this can skew results and decrease validity of the model. We must ensure that the training data is representative of the full space, allowing the most generalization to occur.

3.3.2 Graph Neural Network (GNN)

The GNN serves as an encoder for the PKAN model, producing a fixed length embedding which is passed on. GNNs, similar to CNNs, use convolutional layers to extract information from matrices. Figure 3.2 shows these GCNConv layers, as they are called. Mathematically, they can be described by the equation: [5]

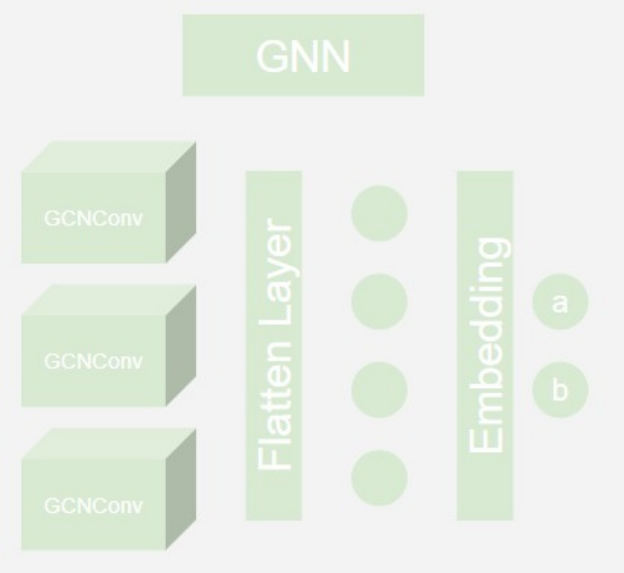

Figure 3.2: GNN Architecture

$$X' = \hat{D}^{-\frac{1}{2}}\hat{A}\hat{D}^{-\frac{1}{2}}XW \tag{3.1}$$

Where $\hat{A} = A + I_N$ and A is the adjacency matrix to undirected graph $G(V, E)$, which is computed before training. $\hat{D_{ii}} = \sum_{j=-0} \hat{A}_{ij}$, a diagonal matrix of $\hat{A}$. After the GCNConv layers, we flatten and move to FC layers. We train on targets a and b, ensuring that we can get predictive capabilities from the GNN model alone. The box labeled "Embedding" in Figure 3.2 represents the last layer of the FC network. This layer is where we will pull our fixed length encoding for the PKAN to use.

3.3.3 Physics-Informed Kolmogorov-Arnold Network (PKAN)

Figure 3.3: PKAN Architecture

The final step of the model, we use the fixed length encoding from the GNN to predict a and b. The PKAN architecture follows the design philosophy of a PINN, which is shown

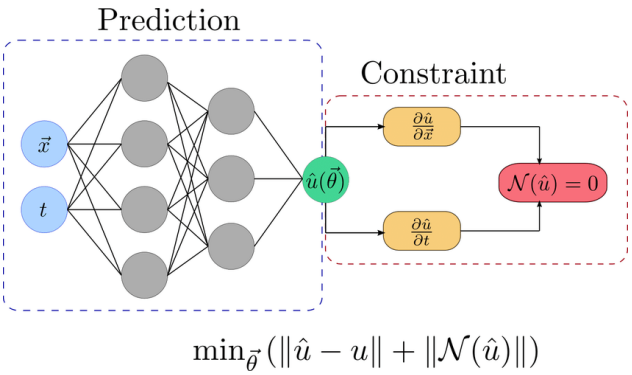

Figure 3.4: PINN Architecture [1]

in Figure 3.4. We differ in our output layers, borrowing the KAN activation layer [6] for increased non-linearity. The KAN layer can be described by the equation:

$$\phi(z) = \sum_{i=1}^{K} w_i k(z, c_i) \tag{3.2}$$

Where $k(z, c_i)$ is a kernel function which acts on input vector z. It features a learnable center, c, and width, σ. We also have the learnable weight w_i, which acts to weigh the outputs before summation. The model is also trained on a and b targets, as the GNN was, but we will take the output as our final predictions this time. Figure 3.4 shows the central idea of a PINN architecture, featuring the physical constraints that bind our system. We have identified constraints:

- $a > 0$
- $b > 0$
- When ordered by molecular weight: $a_i \leq a_{i+1}$, $b_i \leq b_{i+1}$, $\forall i \in \{1, ..., N-1\}$

These constraints are incorporated into the Physics Loss, which is linearly combined with the Model Loss to produce our Total Loss. The Physics Loss is defined as:

$$\begin{aligned} L_{pinn} &= L_{physics} + L_{trend} \\ L_{physics} &= \tfrac{1}{N} \textstyle\sum_{i=1}^{N} (max(0, -a_i) + max(0, -b_i)) \\ L_{trend} &= \tfrac{1}{N} \textstyle\sum_{i=1}^{N} (max(0, a_i - a_{i+1}) + max(0, b_i - b_{i+1})) \end{aligned} \tag{3.3}$$

In tandem, we calculate the Model Loss using the Mean Squared Error from our predicted (a, b) and the actual targets. We calculate the total loss as:

$$L = A(L_{model}) + B(L_{pinn}) \tag{3.4}$$

3.3.4 GUI

The GUI serves as a simple, streamlined interface that allows users to interact with the model pipeline. It can be broken up into its main components:

- **Molecule Builder**: The molecule builder is the central user experience of the GUI. Users will be able to create arbitrary molecules with any amount of atoms, elements, and bond types.
- **Molecule Viewer**: From the built molecule, the user can view a 2D or 3D representation of their designed molecule. This will allow for assessing, correcting, and visualizing of inputs before prediction.
- **Parameters and Options**: During testing, options for simulating, predicting, or visualization will be incorporated as we identify need.
- **Predictions**: The GUI now imports the created molecule into the prediction pipeline at the click of a button, and once completed will display prediction outputs, confidence intervals, and other relevant information.

3.3.5 Testing and Validation

1. **Model Training and Validation**
    - **Training Strategy**: Employ a coordinated training approach, possibly involving sequential or simultaneous training of the individual components, to optimize the aggregate model's performance.
    - **Validation**: Split the dataset into training and validation sets to assess the model's performance and generalization capabilities.
    - **Hyperparameter Tuning**: Optimize model parameters to enhance predictive accuracy and generalization.
2. **Model Evaluation**
    - **Performance Metrics**: Utilize metrics such as Mean Absolute Error (MAE), Root Mean Square Error (RMSE), and R-squared to evaluate model accuracy.
    - **Cross-Validation**: Implement k-fold cross-validation to ensure robustness and mitigate overfitting.
3. **Deployment**
    - **User Interface**: Develop an accessible GUI for users to input molecular structures and receive predicted a and b constants.
    - **Integration**: Embed the predictive model into our GUI Python application.

Chapter 4

Design Concepts, Evaluation & Selection

4.1 User Interface Design

Figure 4.1 shows an initial design of the GUI, featuring the molecule creation window, as well as parameters and prediction results. Figure A.2 shows a logo design for the Python app.

4.1.1 Intended User Flows and Workflows

The proposed user interface is designed to facilitate the following workflow:

1. **Molecule Input**
    - Users can either draw molecular structures using an integrated editor or import existing structures in standard formats (e.g., SMILES, MOL).
2. **Prediction Execution**
    - Upon submission, the system processes the input through the predictive model, with progress indicators informing users of the computation status.
3. **Results Display**
    - The interface presents the predicted a and b constants alongside confidence intervals. Visual aids, such as graphs comparing predicted values to known data, provide contextual understanding.
4. **Data Export**
    - Users have the option to export results in various formats (e.g., CSV, PDF) for integration into external reports or further analysis.

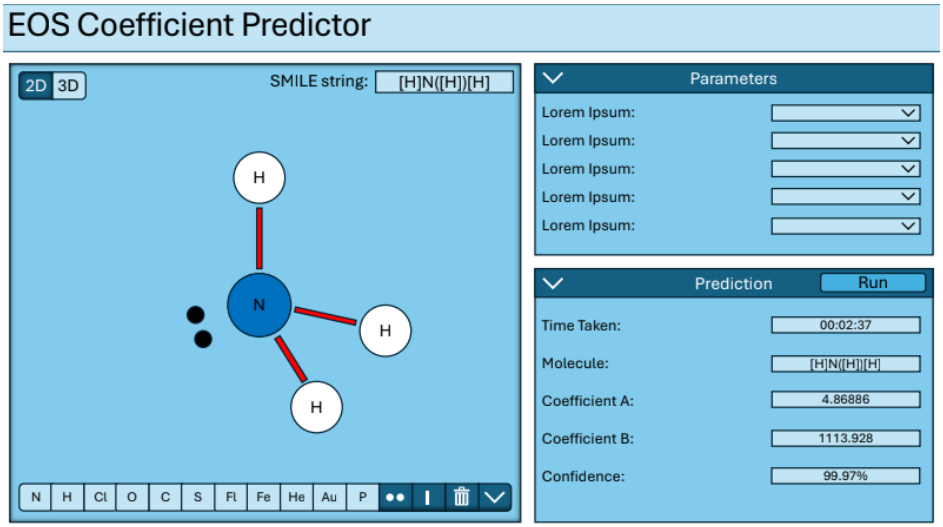

Figure 4.1: GUI Mockup

4.1.2 Target Use Scenarios

The tool is intended for the following user scenarios:

- **Chemical Engineers**
  - Predicting gas behavior in new process designs to aid in selecting appropriate materials and conditions.
- **Pharmaceutical Researchers**
  - Assessing molecular interactions under non-ideal conditions to inform drug formulation and stability studies.
- **Academic Educators**
  - Demonstrating real gas behavior concepts in educational settings, providing students with hands-on predictive capabilities.

4.1.3 Design Validation Plan

To ensure the interface meets user requirements, the following evaluation plan is proposed:

- **Usability Testing**
  - Engage a sample group of target users to perform typical tasks, collecting feedback on intuitiveness and efficiency.
- **Iterative Refinement**
  - Based on user feedback, iteratively refine the interface to address identified issues and enhance user satisfaction.

4.2 Molecular Embedding Techniques

Alternatives Considered

1. **Extended-Connectivity Fingerprints (ECFP)**
    - Circular fingerprints capturing molecular substructures.
2. **Graph Neural Network-based Embeddings**
    - Learn representations by processing molecular graphs.
3. **SMILES-based Sequence Embeddings**
    - Utilize sequential representations of molecules for embedding.

Predicted Performance

- **ECFP**
    - **Advantages**: Simple to compute; captures local structural information.
    - **Disadvantages**: May miss global structural context; fixed size may limit flexibility.
- **Graph Neural Network-based Embeddings**
    - **Advantages**: Captures both local and global structural information; adaptable to various molecular sizes.
    - **Disadvantages**: Higher computational cost; requires substantial training data.
- **SMILES-based Sequence Embeddings**
    - **Advantages**: Leverages existing sequence-based models; straightforward implementation.
    - **Disadvantages**: Sensitive to input representation variations; may not fully capture 3D structural information.

Selected Alternative

Graph Neural Network-based Embeddings are proposed due to their ability to comprehensively capture molecular structures, aligning with the requirement for accurate prediction of a and b constants.

4.3 Model Design Decision Analysis

Several critical design decisions will influence the development and effectiveness of the predictive model:

1. **Architecture of the CNN Component**
    - **Options**: Evaluate different CNN architectures, such as VGG16, ResNet, or ConvNeXt, to determine the most effective for feature extraction from molecular structures.
    - **Implications**: The selected architecture will impact the model's capacity to capture spatial hierarchies and local features within molecular data.
2. **Formulation of the PINN Component**
    - **Options**: Define the specific physical laws and constraints to incorporate into the PINN, such as conservation laws or known thermodynamic relationships.
    - **Implications**: Proper formulation ensures that the model's predictions are physically plausible and adhere to established scientific principles.
3. **Configuration of the KAN Component**
    - **Options**: Determine the structure of the KAN, including the number of layers and the types of univariate functions used.
    - **Implications**: This configuration affects the model's ability to capture complex, nonlinear relationships within the data.

Selected Alternatives

Because our choices will be affected by results, we cannot make decisions about model architectures yet. We plan to investigate all options.

Each design decision will be guided by specific requirements, such as achieving a target predictive accuracy, ensuring computational efficiency, and meeting user needs for interpretability. Our initial design test results can be seen in Figures 4.2 and 4.3. Figure 4.2 implies that we can accurately capture the patterns of a and b values, while 4.3 shows the results of predicting a and b separately for validation.

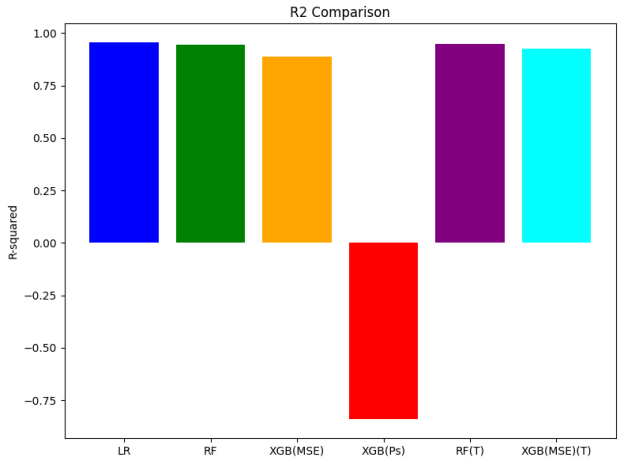

Figure 4.2: R2 Score of Initial Models

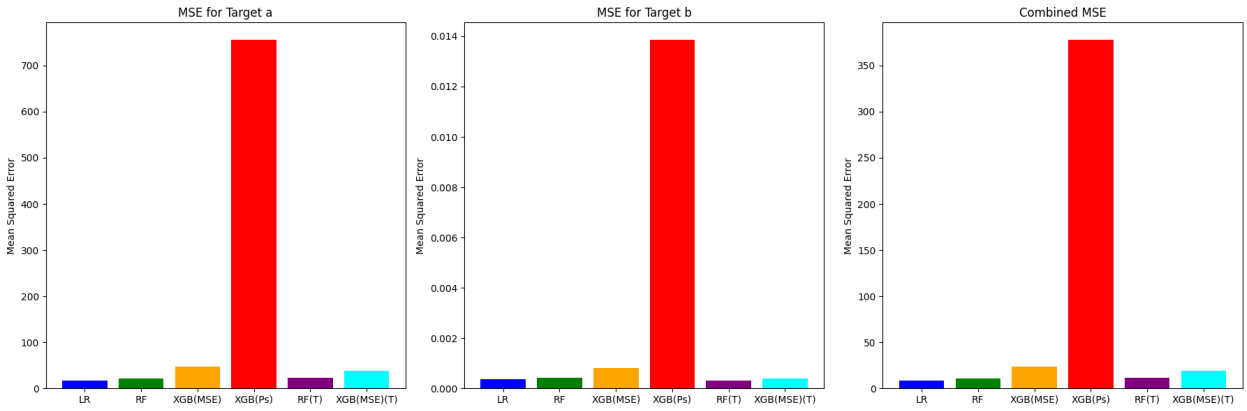

Figure 4.3: Individual a/b MSE Score of Initial Models

Chapter 5

Social Impact Evaluation

5.1 Recognizing and Defining Targeted Needs in a Social Context

This project addresses the need for accurate molecular property predictions, benefiting fields such as drug discovery, materials science, and environmental safety. By improving computational efficiency and predictive accuracy, we enhance scientific research and industrial applications.

5.2 Broader Impacts of Engineering Solutions

The integration of machine learning and physics-based modeling ensures that our system produces reliable and interpretable predictions. This contributes to advancements in cheminformatics, supports sustainable research practices by reducing reliance on costly laboratory experiments, and accelerates innovation in material and pharmaceutical development.

5.3 Ethical and Professional Responsibilities

Ensuring data integrity and model transparency is critical. Our approach prioritizes ethical AI development by:

- Using well-documented and publicly available molecular datasets.
- Clearly stating model limitations and potential biases.
- Ensuring reproducibility by providing open-source implementation details.

5.4 Communication on Societal Issues

Our project highlights the importance of computational methods in addressing global challenges such as efficient drug design and sustainable material development. We will commu-

nicate these impacts through presentations, publications, and collaborations with industry professionals and researchers.

5.5 Understanding Professional and Ethical Responsibilities

As engineering practitioners, we are committed to:

- Upholding rigorous scientific standards in data handling and analysis.
- Promoting accessibility and fairness in AI-driven predictions.
- Maintaining transparency in reporting findings and model performance.

Chapter 6

Deliverables

6.1 Overview

At the conclusion of this project, we will deliver a comprehensive molecular property prediction system, consisting of a user-friendly graphical interface, robust predictive models, and detailed documentation. This system is designed to enable researchers and industry professionals to efficiently analyze molecular properties and leverage machine learning in their workflows.

6.2 Final Deliverables

The project will produce the following key deliverables:

6.2.1 Software System

- **Molecular Property Prediction Pipeline:**
  - Preprocessing tools for generating molecular encodings (e.g., Graph Neural Network-based Embeddings, Coulomb matrices, SMILES-based embeddings).
  - Full predictive model pipeline, including:
    - Graph Neural Network (GNN) for generating fixed length embeddings.
    - PKAN Model for utilizing embeddings to predict a and b constants.
- **Graphical User Interface (GUI):**
  - Fully functional Python application with model-pipeline backend.
  - Interactive molecule input via drawing, file uploads, or SMILES entry.
  - Visualization tools for molecular graphs, Coulomb matrices, and predicted properties.
  - Export functionality for results in CSV or JSON formats.

6.2.2 Documentation

- **User Guide:**
  - Instructions on how to install, configure, and use the system.
  - Examples of workflows for different use cases (e.g., drug discovery, materials optimization).
- **Technical Documentation:**
  - Detailed explanation of the system architecture, model training process, and hyperparameter optimization.
  - Guidelines for extending the system (e.g., adding new encodings or models).

6.2.3 Evaluation Results

- **Performance Metrics:**
  - Validation results for prediction accuracy (R^2, mean squared error) across benchmark datasets.
  - Computational performance benchmarks, including runtime and scalability for large datasets.
- **User Feedback:**
  - Summarized findings from usability testing of the GUI.
  - Recommendations for future iterations based on user feedback.

6.2.4 Source Code and Deployment Tools

- Fully commented source code for the pipeline and GUI.
- Deployment scripts for running the system locally or on a cloud platform.
- Pre-trained models for immediate use without retraining.

6.3 Future Extensions

While not part of the primary deliverables, the system will be designed to facilitate future enhancements, such as:

- Integration with additional molecular encoding methods.
- Support for predicting more advanced molecular properties.
- Scalability to handle larger datasets and distributed computing environments.

Chapter 7

Project Management

7.1 Overview

Effective project management is critical to ensuring the successful and timely delivery of this molecular property prediction system. While the specifics of certain components may evolve, we have developed a preliminary timeline that outlines the major milestones and their corresponding deadlines. This schedule will be refined as the project progresses and new challenges or opportunities arise.

7.2 Project Timeline

Table 7.1 summarizes the key milestones and deliverables for the project:

7.3 Team Roles and Responsibilities

To ensure efficient progress, the following team roles have been defined:

- **Project Manager - Karim Chmayssani:** Oversees the project schedule, ensures deadlines are met, and coordinates between team members.
- **Data Scientist - Blake Milstead:** Develops preprocessing tools and ensures the accuracy of molecular encodings.
- **Model Developer - Karim Chmayssani, Blake Milstead:** Focuses on designing, training, and optimizing predictive models (GNNs, PINNs, KANs).
- **UI/UX Designer - Turner Heath:** Leads the design and implementation of the graphical user interface.
- **Tester - Turner Heath:** Conducts usability and performance testing, ensuring system reliability and user satisfaction.

Milestone	Deadline	Description
Initial Planning and Research	Week 1	Finalize project objectives, gather requirements, and review relevant literature.
Data Preprocessing Pipeline	Week 3	Implement and test tools for generating molecular encodings (Coulomb matrices, SMILES embeddings).
GUI Prototype	Week 5	Build a basic GUI prototype to input molecular data and display results.
GNN Model Development	Week 7	Train and evaluate the GNN for molecular graph analysis.
PINN Model Development	Week 10	Incorporate physics-informed constraints into a predictive model.
KAN Aggregation Implementation	Week 11	Develop and integrate the kernel attention network for prediction aggregation.
System Integration	Week 12	Combine all components (models, preprocessing, GUI) into a cohesive pipeline.
Performance Optimization	Week 15	Optimize models and preprocessing for speed and accuracy.
Usability Testing	Week 17	Conduct testing with end-users to gather feedback and refine the GUI.
Final Deliverables Submission	Week 18	Submit the completed system, documentation, and evaluation results.

Table 7.1: Preliminary Project Timeline

7.4 Future Refinements

This timeline and role allocation will be revisited at regular intervals to ensure alignment with project goals and to address unforeseen challenges. Updates will be documented and incorporated into subsequent phases of the project.

Chapter 8

Budget

8.1 Overview

This section outlines the preliminary budget estimates for the molecular property prediction project. The budget considers software, hardware, and personnel costs, with engineering time being the most significant expenditure. The estimates will be refined as the project progresses and specific requirements are finalized.

8.2 Estimated Expenditures

The anticipated costs are categorized as follows:

Category	**Estimated Cost (USD)**	**Description**
Software Licenses	0	All used software is free
Cloud Services	300	Google Colab computing hours, GPU computing credits ($10/100hrs)
Miscellaneous	100	Expenses for documentation, presentations, and potential third-party APIs for molecular data.
Total Estimated Cost	400	

Table 8.1: Preliminary Budget Estimates

8.3 Engineering Time Allocation

The project timeline is broken into milestones, with estimated human-hours allocated for each phase:

Milestone	**Estimated Hours**	**Description**
Initial Planning and Research	10	Literature review, requirements gathering, and project planning.
Data Preprocessing Pipeline	10	Implementing tools for molecular encodings and data validation.
GNN Development	30	Designing, training, and testing graph neural networks.
PINN Development	30	Incorporating physics-based equations into a predictive model.
KAN Implementation	30	Developing kernel attention networks.
GUI Development	20	Creating a user-friendly interface for input, prediction, and visualization.
System Integration and Optimization	10	Combining all components and optimizing performance.
Testing and Documentation	30	Usability testing, system evaluation, and preparing project documentation.
Total Estimated Hours	**170**	

Table 8.2: Estimated Human-Hours Allocation

8.4 Future Adjustments

This budget represents an initial estimate and will be updated as the project evolves. Actual expenditures and time allocations will be recorded and compared to these estimates to ensure accountability and adaptability.

Bibliography

[1] Unknown, "A sample schematic of a pinn architecture for the linear transport equation," https://www.researchgate.net/figure/A-sample-schematic-of-a-PINN-architecture-for-the-linear-transport-equation-u-t-u-x_fig5_344783581, Accessed 2024, accessed on 2024-12-03.

[2] S. M. College, "10: Experimental determination of the gas constant (experiment)," https://chem.libretexts.org/Ancillary_Materials/Laboratory_Experiments/Wet_Lab_Experiments/General_Chemistry_Labs/Online_Chemistry_Lab_Manual/Chem_10_Experiments/10%3A_Experimental_Determination_of_the_Gas_Constant_%28Experiment%29, 2020, accessed on 2024-12-03.

[3] S. Genheden, A. Reymer, P. Saenz-Méndez, and L. A. Eriksson, "Computational chemistry and molecular modelling basics," in *Computational Tools for Chemical Biology*, S. Martín-Santamaría, Ed. Royal Society of Chemistry, 2017, pp. 1–38. [Online]. Available: https://books.rsc.org/books/edited-volume/630/chapter/312482/Computational-Chemistry-and-Molecular-Modelling

[4] Microsoft_Research, "Visnet: A general molecular geometry modeling framework for predicting molecular properties and simulating molecular dynamics."

[5] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[6] Z. Liu, Y. Wang, S. Vaidya, F. Ruehle, J. Halverson, M. Soljačić, T. Y. Hou, and M. Tegmark, "Kan: Kolmogorov-arnold networks," *arXiv preprint arXiv:2404.19756*, 2024.

Appendix A

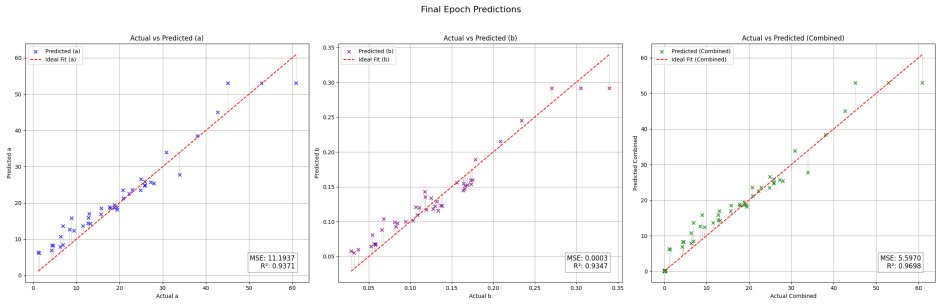

Figure A.1: Prediction Results with Initial Embeddings

Figure A.2: Equation of State Coefficient Predictor Logo (Van der Waals Coefficient Predictor)