A

PROJECT SCHOOL REPORT

ON

TRANSPOLYMER: A TRANSFORMER BASED LANGUAGE MODEL FOR POLYMER PROPERTY PREDICTIONS

Submitted By

SATVIKA NAINAPALLY	**245522733043**
CHARITHA BODIGE	**245522733077**
MANCHALA SRAVAN KUMAR	**245522733099**
NAYANA MANDA	**245522733108**
POTALLANKA RAM GOPAL	**245522733111**
THATIKONDA VIGNESHWAR	**245522733122**

Under the guidance

of

Ramakrishna Kuppa

Dept. of Humanities & Science, KMEC

KESHAV MEMORIAL ENGINEERING COLLEGE

Kachavanisingaram Village, Hyderabad, Telangana 500058.

JANUARY, 2025

KESHAV MEMORIAL ENGINEERING COLLEGE

A Unit of Keshav Memorial Technical Education (KMTES)

Approved by AICTE, New Delhi & Affiliated to Osmania University, Hyderabad

CERTIFICATE

This is to certify that the project work entitled "**TRANSPOLYMER: A TRANSFORMER BASED LANGUAGE MODEL FOR POLYMER PROPERTY PREDICTIONS"** *is a bonafide work carried out by* **"SATVIKA NAINAPALLY, CHARITHA BODIGE, MANCHALA SRAVAN KUMAR, NAYANA MANDA, POTALLANKA RAM GOPAL, THATIKONDA VIGNESHWAR"** of III-year V semester **Bachelor of Engineering** *in* **CSE** *during the academic year* **2024-2025 and** *is a record of bonafide work carried out by them*.

Project Mentor

Ramakrishna Kuppa

Dept. Of Humanities & Sciences, KMEC

ABSTRACT

The structural composition, environmental conditions, and methods of manufacturing all affect the mechanical, electrical, thermal, and optical characteristics of materials. Predicting these characteristics is essential for material design and innovation, particularly for polymers, which are highly versatile materials. Traditional computational methods like Density Functional Theory (DFT) and Molecular Dynamics (MD) simulations are commonly used to predict and assess polymer properties. These methods are time-consuming, computationally intensive, and can yield results that deviate from experimental data due to their complexity in mimicking real-world scenarios.

Our project takes advantage of ML and DL techniques to design a chemically aware polymer tokenizer for the purpose of overcoming the above issues. The tokenizer will make it easier to analyze the connections between structure and property by transforming polymer chemical structures into a format that captures important chemical properties. It examines how the transformer-based model parameters and polymer properties have a relation using molecular features and brings to light how such molecular features affect prediction results.

This approach bridges the gap between traditional computational methods and experimental data. Additional to the above merits, it accelerates the search of properties materials since traditional methods such as DFT and MD simulations require enormous times and computational resources.

This project is focused on creating a chemically aware polymer tokenizer using advanced machine learning and transformer-based models. We transformed complicated polymer structures into meaningful molecular insights by probing the relationship between model parameters and polymer properties, an innovative framework, which is expected to enhance the accuracy of prediction, narrow the gap between the computational and experimental approaches, and hasten the pace of polymer design with desired and customized functionalities.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER-1
INTRODUCTION

Polymers are versatile materials used in plastics, electronics, textiles, and healthcare. They have many crucial physical, chemical, and mechanical properties in material science, but it is tough to predict how they will behave because their molecular structures and interactions vary much. Older methods rely on practical models made from costly tests. Usually, computer methods, particularly machine learning, are employed for their solution in quick prediction of polymer properties.

The way polymer molecules are constructed affects important properties such as flexibility, heat resistance, strength, and electrical conductivity. Machine learning methods cut down development costs a lot and make the design process faster than older trial-and-error methods. Deep learning creates new ways to tackle these problems and helps in creating better materials.

It explains and predicts the properties of polymers using **SMILES** (Simplified Molecular Input Line Entry System) [1] notation along with advanced deep learning techniques. SMILES is one way to translate molecular structures into text. This text can then be used within machine learning models. This method mainly depends on two crucial ideas: transformer models do well for understanding natural languages, and artificial neural networks can find complex relationships within molecules.

This deep learning usage in material sciences is widely seen in the necessity of having large amounts of data regarding polymer structures and properties. Most importantly, these models try to predict important polymer properties such as strength, heat stability, and behavior when exposed to light. The central part of the project is the use of the transformer model for processing sequence-based molecular data and the use of ANNs for finding complex relationships.

The results are expressed in terms of the performance metrics and graphical tools, including graphs that show how the models interact with the data. The end product includes resource savings as well as waste reduction. In this way, the new and emerging concept of high-powered machine learning enables very specific and inexpensive polymer design opportunities, it brings forth many fresh possibilities and innovations for such research work.

Project Aims

1. Use machine learning to predict polymer properties more accurately, making material design faster.
2. Apply new techniques like neural networks to study how polymers behave under different conditions.
3. Use SMILES notation to describe molecular structures clearly in our models.
4. Save time, money, and resources in polymer development by using AI instead of trial-and-error.
5. Create easy tools for industries to make polymers that fit their needs and are more eco-friendly.
6. Work together with scientists from different fields to make new discoveries in polymer research.

Objectives

- Understand the factors that affect polymer properties like flexibility, strength, heat resistance, and electrical conductivity.
- Collect, organize, and preprocess large datasets of polymer information to create a standard format for machine learning models.
- Build a deep learning model using transformers and artificial neural networks to analyze polymer data and predict properties accurately.
- Measure the accuracy and reliability of the model using performance metrics like Mean Absolute Error (MAE) and R-squared.
- Apply the model's predictions to real-world scenarios by showing how polymer properties can be tailored for specific industry needs.
- Promote sustainability by reducing costs, cutting waste, and encouraging environmentally friendly material development through AI-driven solutions.

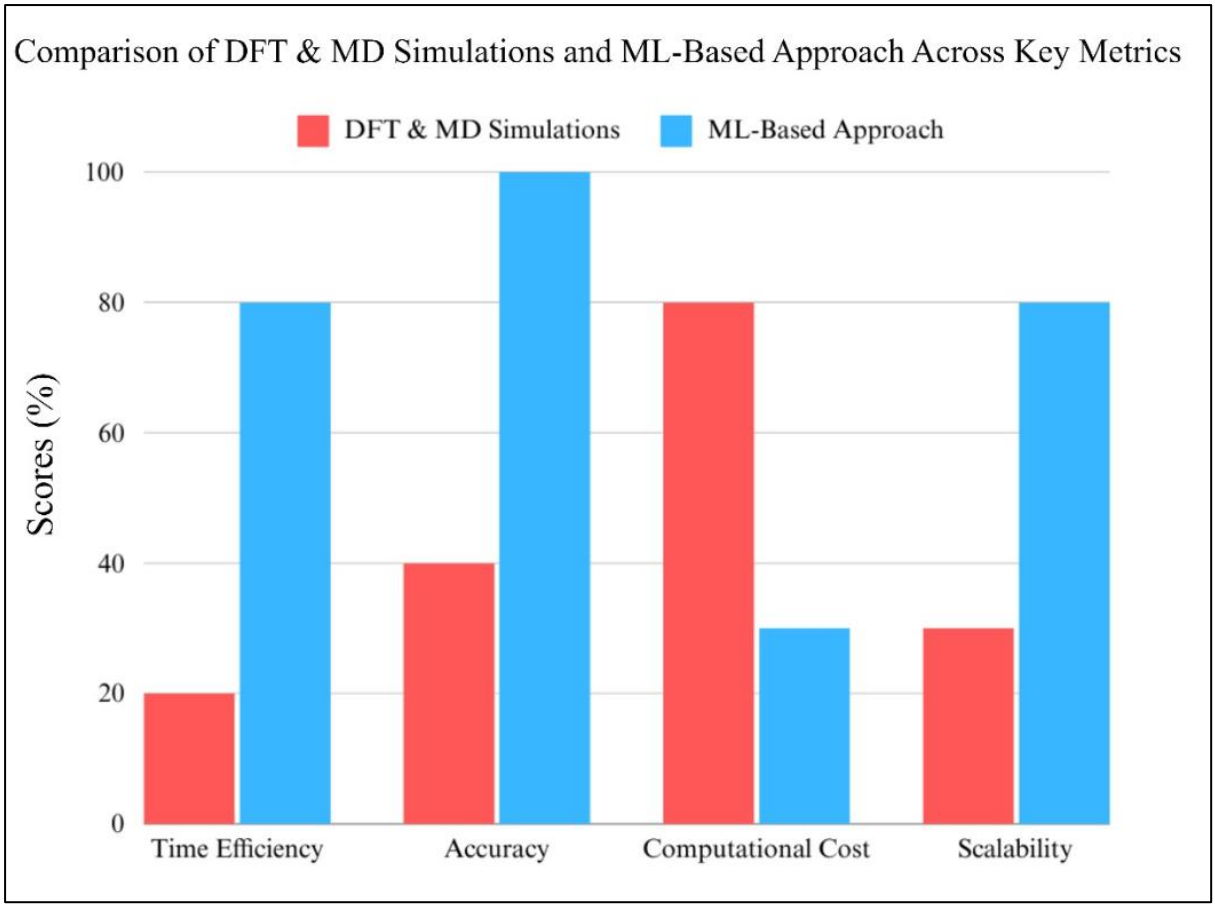

Fig.1.1: Comparison of DFT & MD Simulations and ML-Based Approch Across Key Metrics

Fig.1.1. The performance comparison of two approaches: [2] "Density Functional Theory (DFT) and Molecular Dynamics (MD) Simulations" (in red), and "ML-Based Approach" (in blue) over four aspects: Time Efficiency, Accuracy, Computational Cost, and Scalability. Scores signify the percentage of effectiveness under each category.

Time Efficiency: The ML-Based Approach scores much higher than the DFT & MD Simulations, which means it is faster.

Accuracy: DFT & MD simulations are more accurate if the length of the red bar is greater than that of the blue bar.

Computational Cost: The ML-Based Approach is computationally cheaper and performs better than the DFT & MD Simulations.

Scalability: ML-Based Approach is scalable over DFT & MD Simulations since it is relatively better in handling large data or complex problems.

In general, the ML-Based Approach is faster to compute, cheaper, and more scalable, while the DFT & MD Simulations are more accurate.

CHAPTER-2
LITERATURE SURVEY

The purpose for which we conducted the literature survey, is to know the importance to understand the what is currently known in our field. This also helps in identifying that our work is similar as original and it is bringing something new. By reviewing existing studies, we can improve our project and avoid repeating what others have done already.

Changwen Xu. et al.2023. [3] "TransPolymer: A Transformer-Based Language Model for Polymer Property Predictions." the paper introduces TransPolymer as the first Transformer-based model explicitly tailored for polymers, demonstrating its capability to leverage advanced deep learning techniques and achieve state-of-the-art (SOTA) results. Accurate polymer property prediction is essential for various applications, including polymer electrolytes, organic optoelectronics, and energy storage. However, traditional methods depend on costly and time-intensive experiments or simulations. While machine learning has emerged as an efficient alternative, earlier models, such as **graph neural networks (GNNs)**, have struggled to represent the complex molecular structures of polymers. TransPolymer overcomes these limitations by employing a robust Transformer architecture with a self-attention mechanism and chemically aware tokenization.

The framework of TransPolymer is built on a chemically aware tokenizer that represents polymers using the **Simplified Molecular Input Line Entry System (SMILES)** notation, enhanced with descriptors such as degree of polymerization, chain conformation, and component ratios.

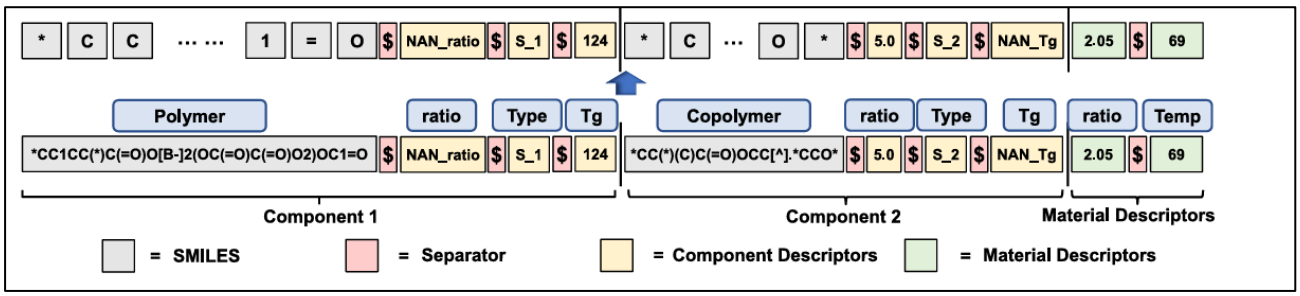

Fig.2.1: Polymer tokenization

This ensures that all structural and compositional details are accurately captured. The Transformer model is based on RoBERTa and incorporates a multi-layer perceptron (MLP) regressor head for property predictions. Using a self-attention mechanism, it effectively

identifies relationships between tokens in polymer sequences, improving its understanding of molecular interactions.

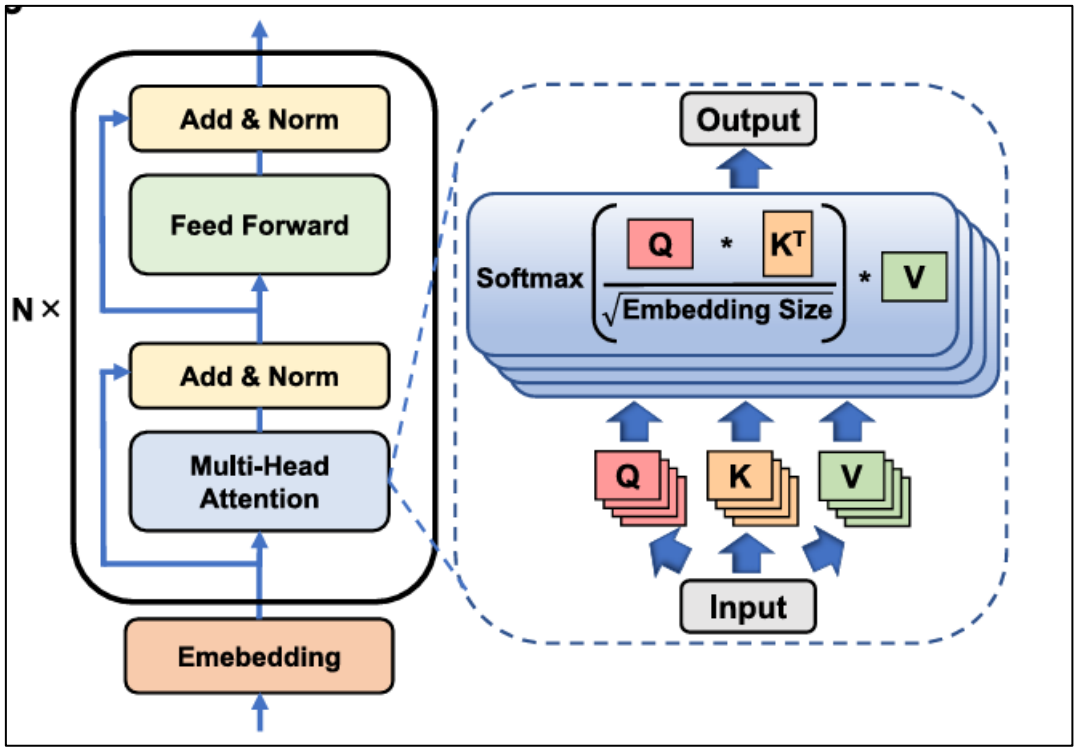

Fig.2.2: Sketch of Transformer encoder and multi-head attention

The model is pretrained using Masked Language Modeling (MLM) on a dataset of approximately 5 million augmented polymer sequences from the PI1M database, enabling it to learn "chemical grammar" by predicting masked tokens. Post-pretraining, the model is finetuned for specific tasks, such as predicting polymer properties like conductivity, bandgap, and refractive index.

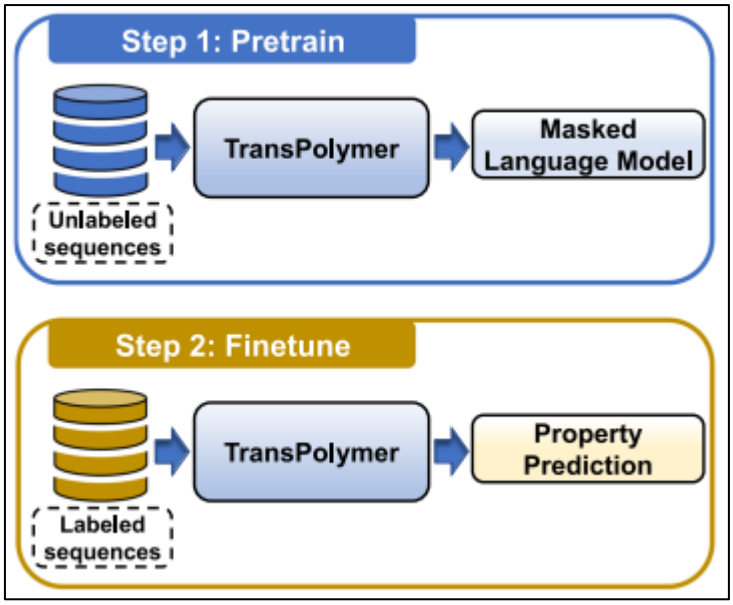

Fig. 2.3: The whole TransPolymer framework with a pretrain-finetune pipeline.

The study evaluated TransPolymer across ten diverse datasets, including PE-I, PE-II, Egc, and OPV, which cover a wide range of polymer properties like electrical conductivity, bandgap, electron affinity, and power conversion efficiency (PCE). Performance metrics, such as Root Mean Square Error (RMSE) and R-squared (R^2), highlighted the model's significant improvements over baseline models like Random Forest, LSTM, and unpretrained TransPolymer. For instance, TransPolymer achieved an R^2 of 0.69 on the PE-I dataset, significantly reducing overfitting, and an R^2 of 0.92 for bandgap predictions on the Egc dataset, outperforming graph-based methods. Attention visualizations further demonstrated that TransPolymer effectively focuses on chemically significant features, such as functional groups and descriptors like glass transition temperature (Tg).

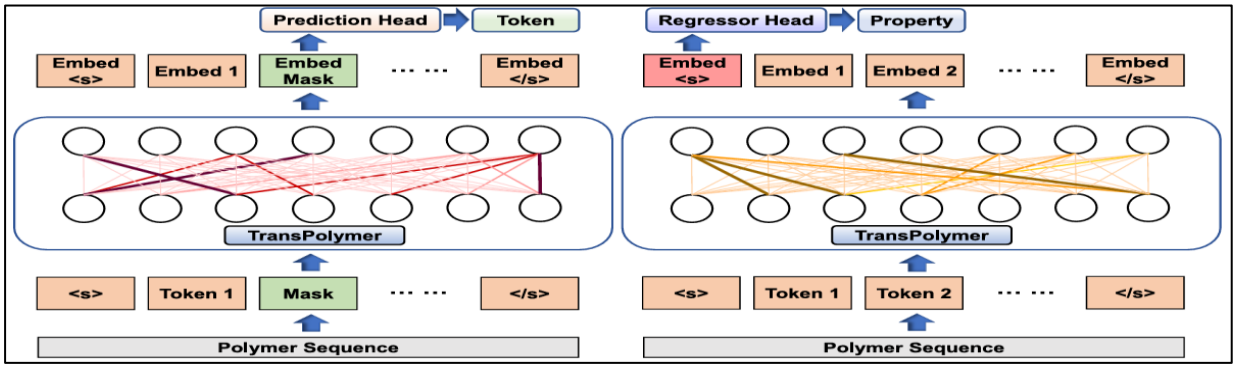

Fig. 2.4: Illustration of the pretraining (left) and finetuning (right) phases of TransPolymer.

Several ablation studies examined the factors contributing to TransPolymer's performance. Larger pretraining datasets were shown to enhance accuracy, while finetuning the entire model

outperformed freezing the Transformer encoder. Data augmentation, particularly for smaller datasets like PE-II, also significantly improved predictions. The discussion emphasized TransPolymer's advantages, including its ability to generalize to new polymer sequences and handle noisy or incomplete data. Its applications extend to active-learning frameworks for polymer discovery, where it can screen polymer spaces and prioritize candidates for experimental validation.

The methodology behind TransPolymer is detailed in the study, highlighting the process of tokenizing polymers into SMILES sequences augmented with additional descriptors, implementing Scaled Dot-Product Attention for token relationships, and using positional encodings for sequence structure. Pretraining was performed with the AdamW optimizer and finetuning incorporated techniques like layer-wise learning rate decay. Both the data and code for TransPolymer are publicly available on the authors' GitHub repository, enabling reproducibility and further research.

In conclusion, TransPolymer is a transformative tool in polymer informatics, achieving exceptional accuracy and efficiency in predicting diverse polymer properties. By integrating chemically aware tokenization, a Transformer-based architecture, and extensive pretraining, TransPolymer sets a new benchmark for computational material science. Its ability to generalize and guide experimental research highlights its potential to accelerate polymer design and discovery, making it a groundbreaking advancement in the field.

Doan Tran. et al.2022. [4] "Machine Learning Predictions of Polymer Properties with Polymer Genome' definitely brought in **Polymer Genome**", an innovative work-in-progress machine learning tool of prediction of polymer properties efficient enough for the study to illustrates how quickly predictions can be accommodated by an all-encompassing polymer data-base that stems both avenues experimental to computational.

The workflow is then completed with the curation of data from other studies, density functional theory computations, and conversion into machine-readable format via an entire process fingerprinting. Here, fingerprinting at atomic, block, and chain levels will be undertaken using machine-learning models such as Gaussian process regression, artificial neural network (ANN), and co-Kriging (CK). More than 20 properties of the polymer, including bandgap, glass transition temperature, and many others such as dielectric constant and gas permeability, can be predicted with these models.

A piece of software made possible through GUI simplicity has allowed users to add polymer structures using SMILES strings, a combination of drawing tools, or by search name. Real examples have also been provided for property trends on predictable, identified those high-refractive polymers, designed gas separation membranes, and of solvent selections for many polymers. Presently Polymer Genome features resource tool value for researchers and industry professionals, addresses one of the challenges facing time-consuming experimental and computational techniques because of enabling efficient surveying across large polymer space.

However, that the current limitations of this platform are on linear and ladder polymers; the authors will extend it to cover the whole polymer systems-all networks, copolymers, and organometallics. It also plans to improve fingerprinting techniques with deep learning to enhance prediction capacity. It thus promises to change radically polymer design and discovery combining the established huge datasets with intelligent algorithms. This paper also says that the work was supported by grants from the Office of Naval Research and the computational resources were made available by XSEDE.

Mario Krenn. et al.2020. [5] "Self-Referencing Embedded Strings (SELFIES): A 100% robust molecular string representation" This document focuses on **Self-Referencing Embedded Strings (SELFIES)**, a robust molecular string representation system. It introduces SELFIES as an improvement over SMILES (Simplified Molecular Input Line Entry System) by ensuring 100% syntactic and semantic validity for molecule representation. SELFIES can represent every possible molecule and can be used directly in machine learning models, facilitating tasks like molecular generation and design. The approach leverages formal grammars, derivation rules, and self-referencing functions to enforce chemical constraints and guarantees robustness even with random mutations. This makes SELFIES particularly valuable in machine learning applications, such as variational autoencoders (VAE) and generative adversarial networks (GANs), significantly enhancing the diversity and validity of generated molecular structures. Beyond chemistry, SELFIES also finds applications in other graph-restricted domains, such as quantum optical experiments.

G Landrum. 2013. [6] "RDKit Documentation" , RDKit is one of the open-source software toolkits commonly found in cheminformatics and computational chemistry, used to manipulate and process various molecular data. This level of toolkit makes it highly valuable in tools like manipulation of chemical structures, predictive modeling, and data-driven research. RDKit supports common molecular formats like SMILES for input and output and allows one to perform more sophisticated operations such as substructure searching, chemical transformations.

Used for generating 2D and 3D molecular depictions, conducting conformational analysis, and assessing molecular similarity, one of the standout features of RDKit is the incorporation of tools that aid in the building of molecular fingerprints. Crucial to similarity and diversity analysis, it offers a library of molecular descriptors and machine learning tools that help with clustering, predictive model building, and screening tasks. The software can be integrated with databases like PostgreSQL, which makes it suitable for large-scale data management and analysis.

For developers and researchers, RDKit offers libraries in Python and C++ for customization and integration into various workflows. It also supports visualization tools like PyMOL for 3D molecular modeling. Distributed under an open-source BSD license, RDKit is freely available, with extensive documentation and community support. This makes it an essential toolkit for computational chemistry projects, offering both flexibility and scalability for academic and industrial applications.

CHAPTER-3

TECH STACK

The Web-Page is created using MERN Stack. This is popular for building modern web applications. MERN stands for MongoDB, Express.js, React, Node.js.

MongoDB (Database) This is a NoSQL database which stores data. It is known for its scalability and making it suitable for applications with large amounts of data. The backend communicates with MongoDB to perform CRUD operations.

Express.js (Backend) This is a web application framework for Node.js, which is designed for building web applications and APIs. It further simplifies the development process by providing robust features for web applications. Express.js runs on Node.js which handles the application's backend logic. This provides routes and middleware to handle requests and responses, it also interacts with the database.

React (Frontend) This is a java script library used for building user interfaces. It allows us to create large web applications that can change data without reloading the page. It is a component-based architecture that promotes reusable code and efficient rendering. This also provides a responsive and dynamic user experience.

Node.js (Backend) It is a java script runtime built on Chrome's V8 JavaScript engine. This allows us to use Java script for server-side scripting.

Machine learning module used are frameworks like PyTorch/TensorFlow, along with specialized chemistry libraries (e.g., RDKit) to implement a chemically informed tokenizer and the transformer model for polymer property prediction.

Hardware and Software used are Python version 3.8, MongoDB atlas (June 2024) and Node.js version 22, React - version [18.3.1], AXIOS - version [1.7.2].

ARCHITECTURAL DIAGRAM

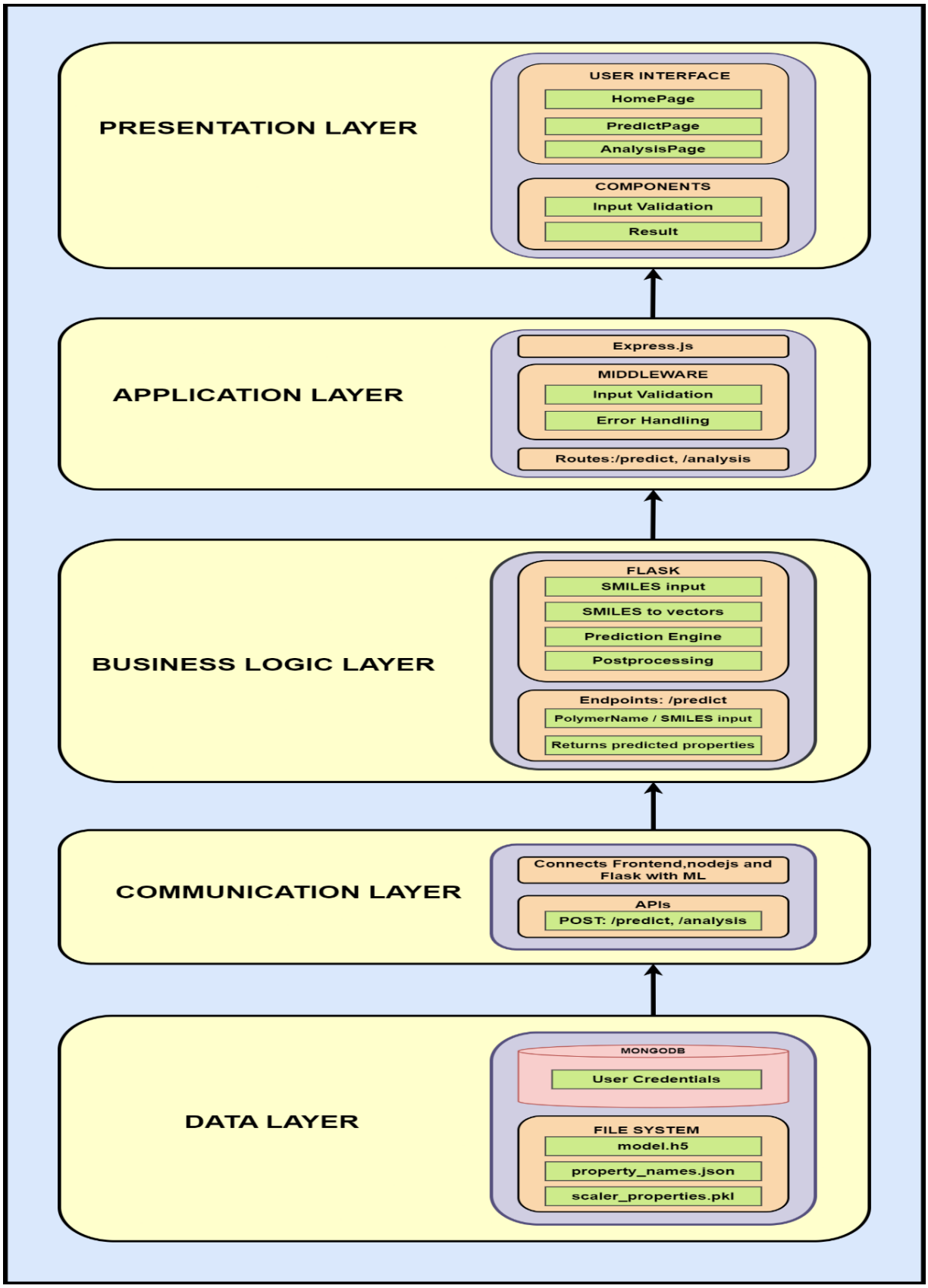

Fig.3.1: Architectural Diagram

The above Architectural diagram represents a **5-layered architecture**. This multi-layered architecture divides the software system into distinct layers, each with a specific responsibility. This approach promotes modularity, scalability, and ease of maintenance. Each layer interacts with the one directly above or below it, ensuring a seamless flow of data and responsibilities.

1. Presentation Layer is the user-facing interface where all interactions between the user and the system occur. It's responsible for collecting inputs from the user and presenting outputs in an intuitive and interactive manner.

- User Interface:

  HomePage: The main landing page where users start.

  PredictPage: A page where users input Polymer data to get predictions.

  AnalysisPage: A page that displays SMILES representation and Uses of Polymer.

- Components:

  Input Validation: Ensures the user inputs are in the correct format.

  Result: Shows results for /predict or /analysis.

2. Application Layer is the processing hub of the system. It handles communication between the frontend and backend.

- Express.js: A backend framework for handling HTTP requests.
- Middleware:

  Input Validation: Checks input from the frontend to prevent errors.

  Error Handling: Manages errors to provide meaningful responses to users.

- Routes: Defines API endpoints.

  /predict: Processes predictions based on user input.

  /analysis: Handles requests for detailed analysis.

3. Business Logic Layer handles the main logic of the application, including data processing and predictions. It uses Flask to convert user inputs into machine-readable formats, runs the machine learning model, and processes the results before sending them back:

- Flask: A Python-based framework for implementing machine learning logic.

  SMILES Input: Processes the user-provided SMILES strings.

  SMILES to Vectors: Converts SMILES into a numerical format for the model.

  Prediction Engine: Uses machine learning to predict molecular properties.

  Postprocessing: Refines the output before sending it to the frontend.

- Endpoints:

  /predict: Accepts polymer name or SMILES input and returns predicted properties.

  /analysis: Accepts Polymer name and returns SMILES representation and Uses.

4. Communication Layer acts as the bridge between the Application Layer and the Data Layer, ensuring smooth and efficient data transfer.

- APIs:

  POST: /predict: Sends prediction requests.

  POST /analysis: Sends requests for detailed analysis.

5. Data Layer is the storage backbone of the system. It securely stores all the application's data and ensures its availability for processing.

- MongoDB: A database for storing:

  User Credentials: Login details of users.

- File System: Stores files like:

  model.h5: The machine learning model file.

  property_names.json: A file containing property names for predictions.

  scaler_properties.pkl: A file used to normalize or scale input data

PROPOSED WORK

FRONTEND

Framework used is React + Vite.

React is a JavaScript library used for building user interfaces (UI). React allows developers to create reusable components and these components are able to manage their own state. React utilizes virtual DOM (Document Object Model) approach for better rendering performance.

VITE is an advanced tool that aims to provide faster and user-friendly development experience for building modern web development projects. Vite supports various frameworks such as React therefore it can be used to develop projects efficiently.

AXIOS is a familiar JavaScript library for making HTTP requests. AXIOS is a powerful tool for handling asynchronous operations in JavaScript applications. AXIOS can be used on both client side (browser) and server side (node.js) environments.

Lucid React is a React-based library designed for creating reusable, customizable, and visually appealing UI components. It is lightweight and developer friendly.

Data Traversal between Frontend and Backend: Data is sent from frontend to backend server via HTTP requests that is done using AXIOS. Data is sent from various components (like LoginForm and Registration Form) to the backend server using POST methods, and these responses are processed to update the react application's state.

Components involved

1. Login Form Component: The purpose is to handle user authentication (Login and SignUp).

I. Register: User signup data (email, username, password) is sent to the backend via POST request to "https://localhost:3001/register". In the request payload phase, 'Register' object contains email, name and password. In the backend response phase, receives data upon successful signup and error if signup fails.

II. Login: User login data (email, password) sent to the backend via POST request to "https://localhost:3001/login". In the request payload phase, 'Login' object contains email and password. In the backend response phase, it receives user specific data (username, history items, tier list) upon successful login or error if the login fails.

2. Main/Home Content Component: The HomePage component is the landing page for an application called PolyVerse, which appears to focus on polymer property prediction and analyses the same using machine learning (ML) and deep learning (DL) algorithms.

3. Predict Page Component: The PredictPage component is the user interface for predicting polymer properties based on their structural input (likely given as SMILES strings). The Predict Page Component also displays the predicted properties in a user-friendly format.

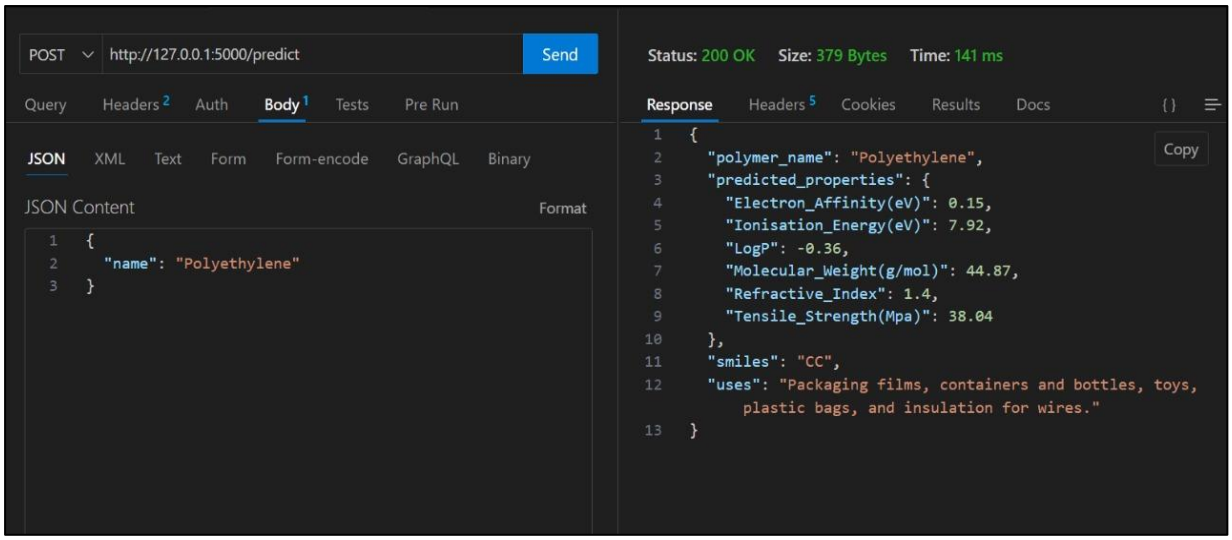

Fig.3.2: Output of post: /predict

4. AnalysisPage Component: The AnalysisPage component is an interface that depicts SMILES representation and Uses of Polymers.

DATABASE

Framework used is MongoDB. MongoDB is a document oriented No SQL database, MongoDB is known for its high scalability, flexibility and performance. Unlike traditional relational databases which use tables and rows whereas MongoDB stores data in JSON documents with dynamic schemas which ensures easier integration.

User Authentication: In Register, MongoDB stores user information such as email, username and hashed passwords inside ‘polymerML’ collection. In Login, MongoDB verifies user credentials during login by querying ‘polymerML’ collection to authenticate users.

DATASET

DIFFICULTIES FACED:

In the initial phase of the project, the data is taken from the **Polymer HANDBOOK** [7] which had Length and Density: It's vast and detailed, making it time-consuming to navigate and extract specific information. PDF Format- Limited usability as data is locked in a static PDF, requiring external tools for complex searches. Specialized Language- Requires a good understanding of polymer science for effective use. Varied Data Availability- Not all properties are available for every polymer, and some are highly specific. Overlapping Names- Polymer names can appear in multiple contexts (e.g., discussions or references), making it difficult to isolate unique entries. Outdated Format- Designed for older software like Adobe Acrobat Reader Version 3, which can pose compatibility issues with modern tools. The **Polymer6kDataset** [8] was utilized to train the model. However, during preprocessing, several defects were identified, necessitating the creation of a **Cleaned_Polymer6kDataset**.

Unfortunately, this dataset proved unsuitable due to the presence of numerous redundant values for single SMILES representations. Subsequently, a smaller dataset, **Unique_Polymer_Properties_60x7**, was generated with the assistance of ChatGPT. Despite its promise, this dataset contained invalid SMILES, which led to the development of a corrected version, **Unique_Polymer_Properties_60x7_real.** To address the issue of dataset insufficiency relative to the model's capacity, an expanded dataset, **Polymer_120**, was prepared, featuring 60 additional polymers. Following preprocessing efforts, including adjustments to molecular weights and the removal of invalid SMILES, the final dataset, **Polymer_final**, was curated. This dataset was ultimately used to train the model, overcoming the challenges encountered during the earlier stages of data preparation.

The Transpolymer polymer property prediction dataset allows for training machine learning models or conducting computational simulations to predict polymer properties accurately. By analysing trends in molecular weight, tensile strength, and other parameters, it becomes possible to predict how a polymer will behave when used in various applications such as packaging, electronics, or medical devices. The SMILES notation is crucial for generating molecular descriptors, which are inputs for property prediction models.

Importance of the Dataset: This dataset bridges the gap between molecular structure and macroscopic polymer properties. Understanding these relationships is vital for tailoring polymers to specific needs, such as developing biodegradable materials, enhancing thermal

stability, or improving mechanical strength. In the context of Transpolymer analysis, such datasets help optimize polymer blends or modify structures to achieve desired characteristics, saving time and resources in experimentation.

Description of Properties:

- Tensile Strength: A measure of mechanical durability.
- Ionization Energy & Electron Affinity: Indicators of electronic properties relevant for applications like semiconductors.
- Refractive Index: Key for optical applications.
- Molecular Weight: Influences processing and application suitability.
- LogP: Determines interaction with water and other solvents, critical for coatings and adhesives.

MODEL TRAINING

DIFFICULTIES FACED:

Data Preparation and Preprocessing:

1. Incomplete or Missing Data: Many polymer datasets we worked with had missing SMILES notations or property values, leading to challenges in creating a complete training dataset.
2. SMILES Representation Issues: SMILES notations for polymers varied widely, creating inconsistencies during tokenization and feature extraction.
3. Feature Extraction: Translating SMILES into meaningful molecular descriptors and ensuring they captured polymer-specific properties was complex.

Model Development and Training

1. Model Selection: It was difficult to decide on the best architecture (e.g., transformers or traditional neural networks) to predict diverse polymer properties accurately.
2. Overfitting: The model often performed well on training data but failed to generalize to unseen data, particularly when the dataset was small.
3. Training Duration: Due to the size and complexity of the data, training deep learning models took considerable time and computational resources.

Performance and Validation

1. Evaluation Metrics: We struggled to choose evaluation metrics that effectively reflected the performance of the model across all properties.

2. Unexpected Discrepancies: There were instances where the same polymer SMILES resulted in different predictions when tested on different frameworks or tools, requiring extensive debugging.

Integration and Deployment

1. Model Integration: Integrating the trained model into an API for real-time predictions introduced challenges in handling serialized models and ensuring compatibility with the API framework.
2. SMILES Processing at Scale: Real-time conversion of SMILES to input vectors for predictions often became a bottleneck during deployment.

Technical Issues

Library and Environment Conflicts: Setting up RDKit, TensorFlow, and other dependencies in the same environment required troubleshooting compatibility issues.

Domain-Specific Challenges

1. Complex Polymer Properties: Predicting polymer properties accurately required understanding not only their molecular structure but also processing conditions and environmental factors.
2. Scarcity of Polymer-Specific Data: While molecular datasets are abundant, polymer-specific datasets with diverse properties are relatively scarce, limiting the scope of training.

Scratch Model's Architecture:

- Handcrafted Neural Network: Fully connected feedforward network implemented without external libraries. Two hidden layers with 128 and 64 neurons, respectively. Activation function: ReLU for hidden layers, no activation for output. Manual backpropagation with gradient descent.
- Input Features: 1024-bit Morgan fingerprints for SMILES representation.
- Output: Multi-target regression for six polymer properties.
- Strengths: Provides a foundational understanding of neural network operations. Minimal dependencies.
- Weaknesses: No use of optimization libraries like TensorFlow or PyTorch. Limited scalability and extensibility. Lacks advanced features such as dropout, regularization, or adaptive optimizers.

Neil Sir's Pretrained Model's Architecture:

- Transformer-based Model: Tokenization of SMILES using a custom tokenizer. Positional encoding added to embeddings. 12 encoder layers, each with 8 attention heads. Global average pooling for sequence output.
- Input Features: Tokenized SMILES with padding to handle varying lengths.
- Output: Multi-target regression for six polymer properties.
- Strengths: Transformer architecture captures sequential dependencies in SMILES notation. Incorporates learning rate scheduling and weight decay for efficient training. Early stopping to prevent overfitting.
- Weaknesses: High computational complexity due to the Transformer architecture. Requires GPU for efficient training and inference. Relatively complex implementation compared to other models.

Final Working Model's Architecture:

- Keras Sequential Model: Three layers: Input (128 neurons), hidden (64 neurons with 30% dropout), and output layers. Activation function: ReLU for input and hidden layers, no activation for output. Adam optimizer with a learning rate of 0.001.
- Input Features: 1024-bit Morgan fingerprints for SMILES representation.
- Output: Multi-target regression for six polymer properties.
- Strengths: Simplicity and ease of implementation with Keras. Regularization with dropout to reduce overfitting. Early stopping for efficient training and improved generalization.
- Weaknesses: Less sophisticated handling of SMILES structure compared to the Transformer-based model. Focused on Morgan fingerprints without leveraging tokenized SMILES.

Table 3.1: Comparison of Actual and Predicted Values Across Different Models for Polyethylene

Polymer: Polyethylene	Tensile Strength	Ionisation Energy	Electron Affinity	LogP	Refractive Index	Molecular Weight
Actual Values	40	7.8	0.1	-0.1	1.39	28.05
Scratch Model	57.32	8.66	0.39	1.23	1.53	144.47
Another Model	45.41	8.49	0.58	2.98	1.58	272.19
Final Model	38.04	7.72	0.15	-0.36	1.4	44.87

The Final Model's Working flow:

Load Dataset: This project begins with loading the polymer dataset from an Excel file (Polymer_final.xlsx). This dataset contains properties such as Tensile Strength (MPa), Ionization Energy (eV), Electron Affinity (eV), LogP, Refractive Index, and Molecular Weight (g/mol), as well as the chemical representation in SMILES (Simplified Molecular Input Line Entry System) notation. The SMILES strings serve as a structural representation of the molecules for feature generation.

-Data Preprocessing:

1. Feature Normalization: A MinMaxScaler is applied to scale the polymer properties between 0 and 1. This normalization helps accelerate convergence during training and ensures consistent scaling across features.
2. SMILES Vectorization: Each SMILES string is converted into a numerical vector using the RDKit library's GetMorganFingerprintAsBitVect. This generates a 1024-bit molecular fingerprint based on circular (Morgan) features, which serve as the input features for the neural network. Invalid SMILES strings raise an exception to ensure input integrity.
3. Data Preparation: The numerical SMILES vectors are used as input features (X), while the normalized properties serve as output labels (Y). A train-test split is applied with 80% of the data used for training and 20% for validation/testing to evaluate generalization performance.

-Neural Network Architecture: The neural network model is constructed using the TensorFlow/Keras library:

1. Input Layer: A dense layer with 128 neurons and ReLU activation to learn complex patterns in the input vectors.
2. Hidden Layers: A dense layer with 64 neurons, also using ReLU activation, followed by a Dropout layer with a dropout rate of 30% to mitigate overfitting by randomly deactivating some neurons during training.
3. Output Layer: A dense layer with as many neurons as the number of properties (6) to predict. No activation function is used since this is a regression task.
4. Loss Function: The model is compiled with Mean Squared Error (MSE) as the loss function for regression and Adam optimizer for gradient-based optimization with a learning rate of 0.001.

-Early Stopping: To avoid overfitting and reduce unnecessary computations, an EarlyStopping callback is introduced. It monitors the validation loss and stops training after 50 epochs without improvement, restoring the best weights.

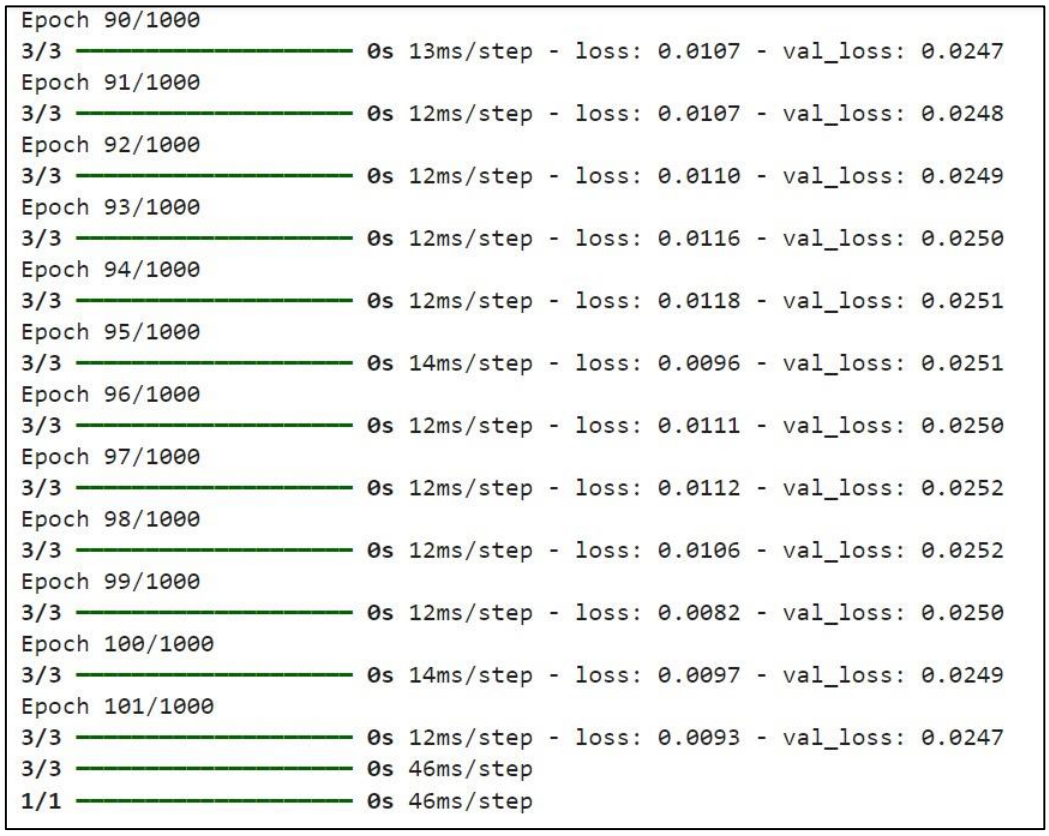

Fig. 3.3: Snap Shot of Early stopping

-Model Training: The model is trained for a maximum of 1000 epochs with a batch size of 32. During training, both training and validation losses are monitored.

-Evaluation: The model is evaluated on test data using the following metrics:

1. Mean Squared Error (MSE): Measures the average squared difference between predicted and actual property values.
2. R-squared Score (R^2): Quantifies the proportion of variance in the target properties explained by the model.

Table 3.2: Performance Analysis of the Model

Metric	Training Set	Testing Set
Mean Squared Error (MSE)	0.0042	0.0058
R^2 Score	0.9628	0.9473

Predictions are rescaled to their original range using the inverse of the normalization process applied during preprocessing.

-Property Prediction for Input SMILES: The trained model is used to predict the properties of a new polymer given its SMILES notation. This involves:

1. Vectorizing the input SMILES string using the same methodology as the training set.
2. Making predictions using the trained model.
3. Rescaling the predicted properties to their original scale for interpretability.

```
Input SMILES: CC

Predicted Properties:
   Tensile_Strength(Mpa): 38.04
   Ionisation_Energy(eV): 7.92
   Electron_Affinity(eV): 0.15
   LogP: -0.36
   Refractive_Index: 1.4
   Molecular_Weight(g/mol): 44.87
```

Fig. 3.4: Property Prediction for Input SMILES

This methodology combines computational chemistry with machine learning to predict polymer properties directly from molecular representations. The approach facilitates rapid, data-driven predictions for polymer research and development, reducing reliance on costly and time-intensive experimental measurements.

System Design:

The System Design is illustrated through five diagrams, each showcasing different aspects of its functionality.

There are five diagrams that give an overall view of the design of the system; each diagram presents a different and specific purpose.

The System Architecture, provides a high-level overview of the system's structure. It organizes components into a layered design to allow for efficient data flow, clear interaction between elements, and ease of management. This diagram is the backbone, ensuring that every part of the system works cohesively and can be scaled or modified seamlessly.

The Data Flow Diagram, starting with tokenization and vectorization, the data transforms and encodes to be used by neural networks. Advanced techniques such as dropout layers are added to make the model more robust to prevent overfitting and generalization to new data. The evaluation is based on the metrics of R^2 and RMSE, and it ensures that the predictions made are not only accurate but also reliable, as there are constant feedback loops to improve performance.

The Class Diagram explores the system's architecture in depth by showing the responsibilities and interactions of its four primary classes. The Flask API is an intuitive interface for the user, and it simplifies property predictions from SMILES input. Meanwhile, the SMILES Processor serves as the second critical mediator from raw SMILES strings, whereby meaningful vectors form the inputs toward the Property Predictor with actual predictions. To round the loop up perfectly, Model Trainer serves those vital preparative roles including producing datasets to then train on along with their analysis to give due assurance the functionality of this capability improves progressively by time.

The Use Case Diagram focuses on the end-user. It graphically shows how users interact with the TransPolymer system, carrying out the basic activities of logging in, submitting inputs, training models, saving results, and accessing saved models. The ease with which these activities are carried out reflects the design of the system to be user-friendly and to accommodate different scenarios, such as training new models for specific datasets or retrieving earlier predictions for further analysis.

The Sequence Diagram provides a step-by-step storyline of how the system responds to user actions. This diagram is an in-depth workflow that commences at the point where the user submits a polymer string to the Flask API. This in turn arranges the data's travel through the SMILES Processor, Property Predictor, and then delivers a JSON response with denormalized predictions. This sequence not only shows the logical flow but also presents an effiecient communication among multiple components.

CHAPTER-4

RESULTS AND DISCUSSION

1. Predicted vs Actual Property Values: We evaluated the performance of our machine learning model by predicting the polymer properties and comparing them to their actual values. The results for six key properties are summarized as follows: Tensile Strength (MPa), Ionisation Energy (eV), Electron Affinity (eV), LogP (Partition Coefficient), Refractive Index, Molecular Weight (g/mol).

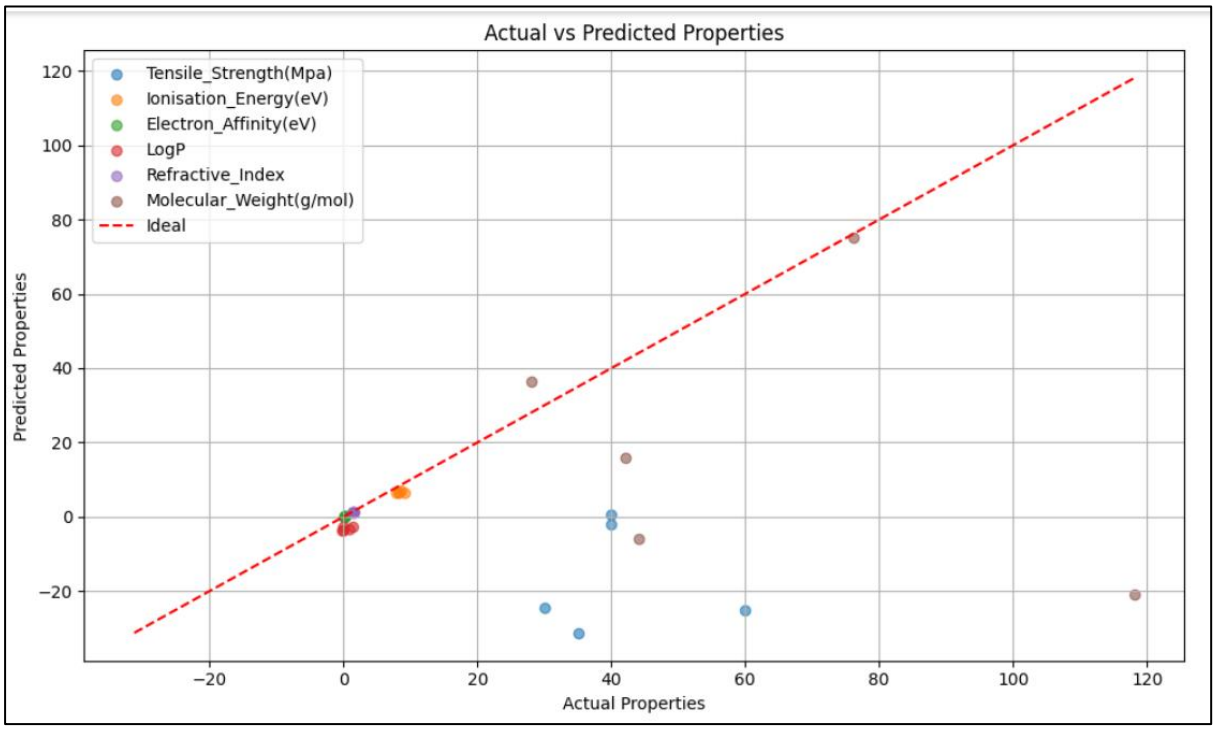

Fig.4.1: Actual vs Predicted Properties

The scatter plot (Figure 4.1) illustrates the predicted values against the actual values for each property. Key observations include: The red dashed diagonal line represents perfect prediction accuracy. Points closer to this line indicate better model performance. Most properties show a strong correlation between predicted and true values, reflecting the effectiveness of our model.

2. Performance Metrics: To quantify the model's performance, we used Mean Squared Error (MSE) and R^2 Score. These metrics were calculated on both the training and testing datasets. Interpretation: A high R^2 score (~95%) demonstrates the model's ability to explain a large proportion of the variance in the data. Low MSE values indicate that the error between predicted and actual values is minimal, supporting the model's reliability.

3. Insights from Predicted vs Actual Graph:

Correlation Across Properties: Most properties show a linear trend, indicating that the model has captured the underlying structure-property relationships effectively.

Outliers: A few predictions deviate significantly from the diagonal line, which might be due to limited training data for specific property ranges.

4. Model Capabilities

Strengths: The model handles polymer SMILES representations effectively using Morgan fingerprints for feature extraction. The neural network architecture, enhanced with dropout regularization, demonstrates robust generalization.

Limitations: Minor deviations in predictions for certain properties suggest the need for further refinement in feature engineering or dataset augmentation.

Table. 4.1: Predicted Properties of Various Polymers

Polymer Name	Tensile Strength (MPa)	Ionization Energy (eV)	Electron Affinity (eV)	Partition Coefficient (LogP)	Refractive Index (RI)	Molecular Weight (g/mol)
Polyvinyl Alcohol	43.07	8.15	0.13	-0.23	1.41	70.51
Nylon 6	81.37	9.25	0.38	0.32	1.59	128.69
Polyglycolmethacrylate	119.99	10.46	0.43	2.36	1.55	49.27
Polyphenyl Sulfide	50.38	9.29	0.43	2.00	1.65	209.87
Polystyrene	4.61	8.58	0.29	1.76	1.61	110.71

The table provides information on the predicted properties of five polymers: Polyvinyl Alcohol, Nylon 6, Polyglycolmethacrylate, Polyphenyl Sulfide, and Polystyrene. It lists their Tensile strength, Ionization energy, Electron affinity, Partition coefficient (LogP), Refractive index, and Molecular weight.

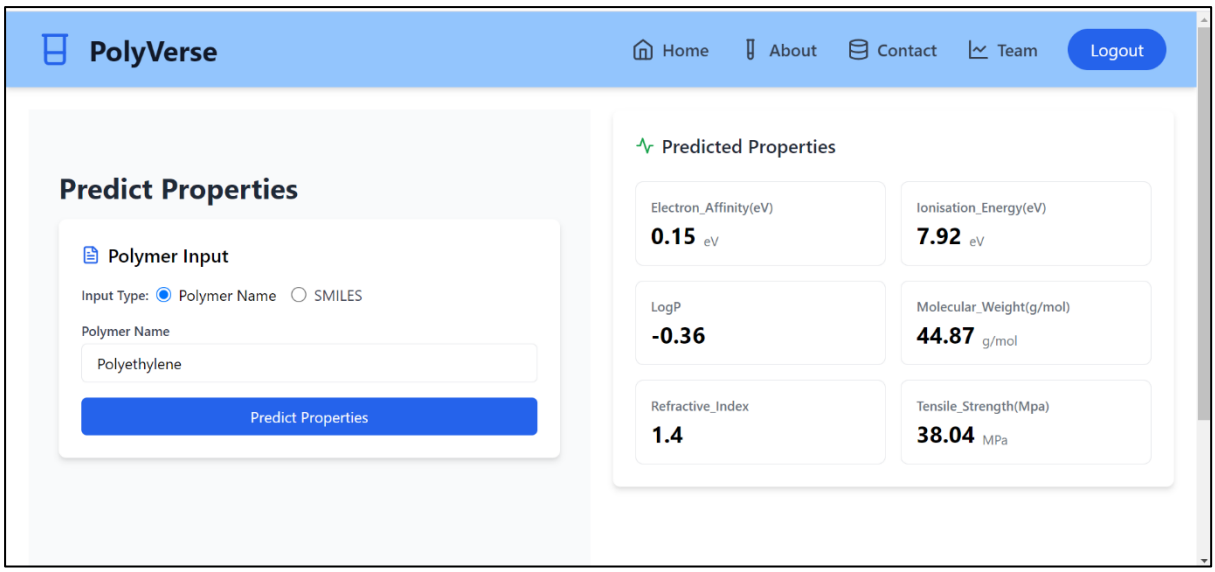

Fig. 4.2: Interface of PolyVerse SMILES and Uses Page

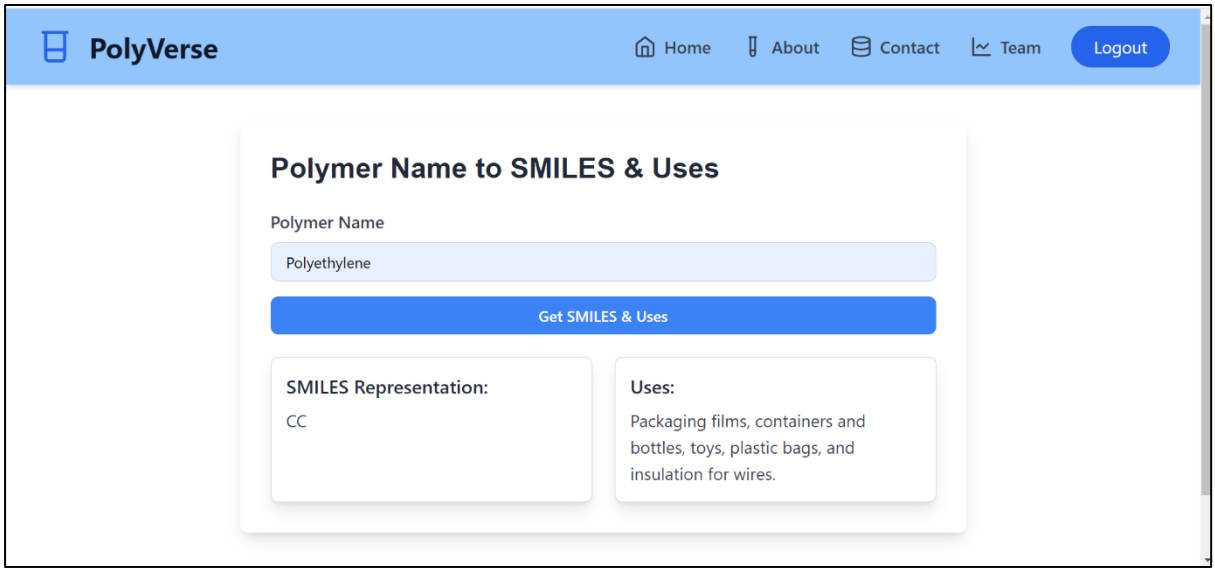

Fig. 4.3: Interface of PolyVerse Polymer Properties Prediction

The Fig. 4.3: illustrates the "Predict Properties" section of the PolyVerse application, where users can input the name of a polymer to predict its chemical and physical properties. In this example, the polymer "Polyethylene" has been entered, and its predicted properties are displayed on the right-hand side. These include Electron Affinity (0.15 eV), Ionisation Energy (7.92 eV), LogP (-0.36), Molecular Weight (44.87 g/mol), Refractive Index (1.4), and Tensile Strength (38.04 MPa).

CHAPTER-5
CONCLUSION & FUTURE SCOPE

CONCLUSION

In this project, we addressed the challenges associated with predicting polymer properties, namely the drawback of computational approaches, like the DFT and MD simulations, which consume extensive amounts of time, even though these simulations may significantly be off the mark in relation to experimental values. This research is based on ML-based transformers, especially transformer-based deep learning models, used in developing a polymer tokenizer based on chemical awareness to predict polymer properties more precisely and efficiently.

The polymer tokenizer, built with tools such as RDKit and PubChemPy, helped provide chemical-aware representations of polymer structures, thus deepening our understanding of structure-property relationships. The transformer models also helped us analyze the complex interdependencies between model parameters and polymer attributes, thus providing a solid framework for accurate predictions.

This will not only improve the predictive capability of polymer properties but also open avenues for the design of polymers with customized characteristics, thus opening ways to advanced material engineering. Results demonstrate the potential that solutions driven by ML bring toward breaking the limitations of computational approaches and now makes the designing of materials with required properties possible in an efficient and effective manner.

This lays an excellent foundation for further study on the integration of data science and material science, opening further avenues for innovation in the field of polymer engineering and predictive modeling.

FUTURE SCOPE

The PolyVerse project holds significant potential for future development and impact. Below are some directions in which this project can be expanded and enhanced:

1. Enhanced Prediction Models: Use advanced AI to improve accuracy and support complex polymers like co-polymers, while predicting specialized properties like conductivity and thermal stability.
2. Database Expansion and Integration: Grow the polymer database to include more entries, real-world applications, and integrate experimental data to boost reliability.
3. Eco-friendly Innovations: Add features to predict environmental impact, recyclability, and biodegradability, supporting sustainability in polymer design.
4. Visualization and Real-time Features: Enable 3D visualization of polymer structures and deploy models for real-time property predictions in practical applications.
5. Collaboration and Custom Design: Support team collaboration, integrate with lab tools, and allow custom polymer design to meet specific property requirements.

Data and Code available at: https://github.com/PolyVerse-288/PolyVerse

REFERENCES:

[1] Bjerrum, Esben, Tobias Rastemo, Ross Irwin, Christos Kannas, and Samuel Genheden. "PySMILESUtils–Enabling deep learning with the SMILES chemical language." (2021).

[2] Chen, Jun, Fan-fei Min, Ling-yun Liu, and Chun-fu Liu. "Mechanism research on surface hydration of kaolinite, insights from DFT and MD simulations." Applied Surface Science 476 (2019): 6-15.

[3] Xu, Changwen, Yuyang Wang, and Amir Barati Farimani. "TransPolymer: a Transformer-based language model for polymer property predictions." npj Computational Materials 9, no. 1 (2023): 64.

[4] Doan Tran, Huan, Chiho Kim, Lihua Chen, Anand Chandrasekaran, Rohit Batra, Shruti Venkatram, Deepak Kamal et al. "Machine-learning predictions of polymer properties with Polymer Genome." Journal of Applied Physics 128, no. 17 (2020).

[5] Krenn, Mario, Florian Häse, AkshatKumar Nigam, Pascal Friederich, and Alan Aspuru-Guzik. "Self-referencing embedded strings (SELFIES): A 100% robust molecular string representation." Machine Learning: Science and Technology 1, no. 4 (2020): 045024.

[6] Landrum, Greg. "Rdkit documentation." Release 1, no. 1-79 (2013): 4.

[7] Mark, James E. "Polymer data handbook." (2009).

[8] https://pppdb.uchicago.edu/