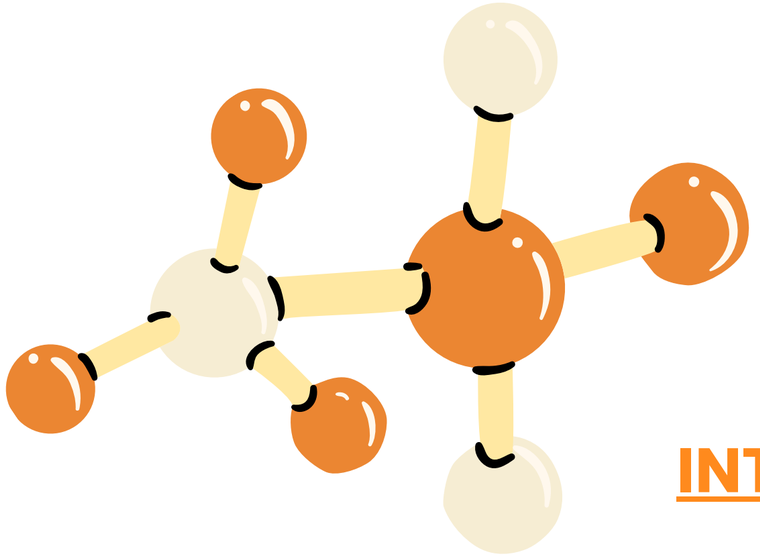


TRANSPOLYMER

G392

INTRODUCTION:

TransPolymer is a Transformer-based language model designed for polymer property prediction by using SMILES tokenizer and Masked Language Modeling (MLM) pretraining to enhance accuracy and efficiency.

SCOPE OF PROJECT:

TransPolymer uses AI to revolutionize polymer research , making it easier , faster , and more accurate to predict how different polymers will behave .

USERS:

- Material Scientists
- Polymer Chemists
- Computational Chemists
- R&D Engineers
- Data Scientist in Material Informatics
- Input and Output

Pain Points:

- Needs high-quality data
- External factors (temperature, aging) not always captured
- High computational power required
- Struggles with unseen polymers

01

TECH STACK:

02

- MERN Stack
- Python
- PyTorch
- Hugging Face Transformers
- RDKit and Scikit-learn
- NumPy, Pandas and Matplotlib.

Business Case:

03

- Cuts Costs & Saves Time
- Speeds Up Material Discovery
- Better Decisions with Data
- Gives Companies an Edge

04

Roles Involved:

- Researcher/User
- End Users(Polymer Engineers/Chemists)

Dataset and Data Source:

We created a custom dataset which contains 6 polymer properties. The dataset was created by referring the information extracted from GitHub repositories related to polymer science.

Cloud Deployment:

- Train & Store → Train on Cloud GPUs, store in S3/GCS.
- Deploy API → Use FastAPI/Flask, deploy via Docker (K8s, Lambda, Cloud Run).
- Frontend → React/Vue + JWT auth for secure access.
- Monitor & Scale → Prometheus + Grafana, auto-scale with K8s/Cloud Run.

Data Preprocessing :

- Verification (Check if the Data is Correct)
- Validation (Ensure Data is Meaningful & Usable)
- Normalization (Standardizing Data for Better Model Performance)

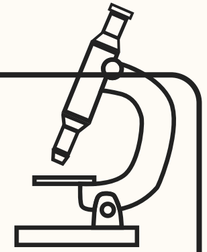
KEY LEARNINGS FROM OUR DEVELOPMENT JOURNEY:

Learned how Transformer models like ChemBERTa predict polymer properties from SMILES.

Gained experience building a full-stack ML app with Streamlit, Python, and MongoDB.

Improved skills in model evaluation, UI design, and deployment challenges.

TRACKING OUR MILESTONE COMMITMENTS:



Successfully implemented SMILES input, property prediction, and result storage.

No features were dropped; the system design was refined for better performance.

Reason for choosing ChemBerta:

- We chose ChemBERTa, a transformer-based model specifically pre-trained on SMILES, making it ideal for understanding molecular structures. It eliminates the need for manual feature engineering and captures complex chemical patterns, leading to more accurate polymer property predictions.

Comparison with other Models:

Traditional ML Models (e.g., Random Forest,SVM):

- Required manual feature engineering; lacked ability to capture sequence patterns in SMILES.

Graph Neural Networks (GNNs):

Effective for molecular graphs but required complex graph construction and were not well-suited for raw SMILES input.

General LLMs (e.g., BERT, GPT):

Not trained on chemical data; lacked domain-specific understanding of molecular structures.

Deployment Approach:

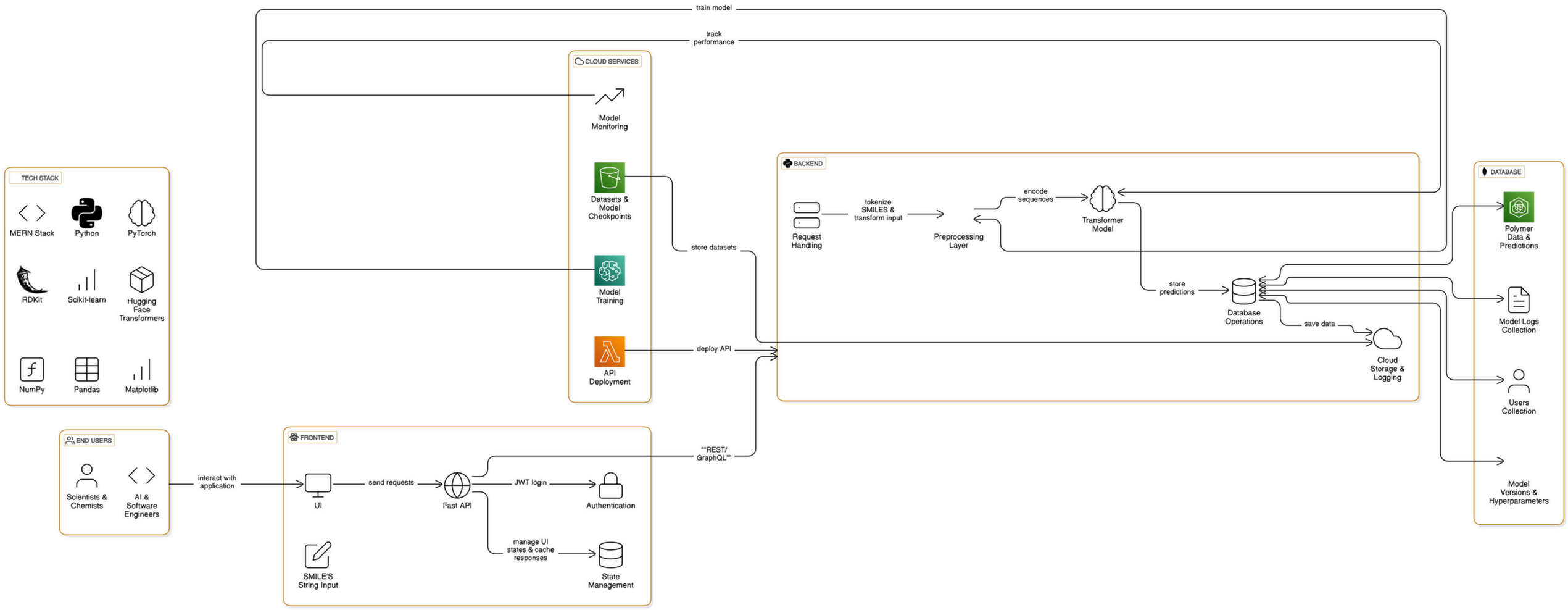
Deployed using Streamlit to combine frontend and backend in one app. ChemBERTa from Hugging Face provides SMILES embeddings, which are combined with RDKit descriptors and passed to a PyTorch transformer model for prediction. The app handles all inference and storage internally, without a separate API.

- **Challenges faced during project**
- **execution:**

- **Data Cleaning** – Handling invalid or inconsistent SMILES strings.
- **High Model Complexity** – Increased training time and resource usage.
- **Multi-Target Prediction** – Balancing accuracy across multiple property outputs.
- **Overfitting Risk** – Ensuring strong generalization on limited data.
- **Feature Fusion Issues** – Effectively combining embeddings, descriptors, and fingerprints

- **changes made from the initial**
- **architecture diagram:**

AT FIRST WE THOUGHT OF USING MERN STACK IN OUR PROJECT BUT LATER WE'VE DEVELOPED THE FRONTEND USING STREAMLIT BECAUSE THE REACTJS AND NODEJS ARE A BIT COMPLEX COMPARED TO THE STREAMLIT. AND AS FOR THE API CONNECTIONS AND DEPLOYMENT, WE DID THE DEPLOYMENT OF OUR PROJECT IN THE HUGGING FACE FREE SPACES SO THERE WAS NO NEED OF ANY API CONNECTIONS


train model
track performance
CLOUD SERVICES
Model Monitoring
Datasets & Model Checkpoints
Model Training
API Deployment
store datasets
deploy API
TECH STACK
MERN Stack
Python
PyTorch
RDKit
Scikit-learn
Hugging Face Transformers
NumPy
Pandas
Matplotlib
BACKEND
Request Handling
tokenize SMILES & transform input
Preprocessing Layer
encode sequences
Transformer Model
store predictions
Database Operations
save data
Cloud Storage & Logging
DATABASE
Polymer Data & Predictions
Model Logs Collection
Users Collection
Model Versions & Hyperparameters
END USERS
Scientists & Chemists
AI & Software Engineers
interact with application
FRONTEND
UI
send requests
Fast API
JWT login
Authentication
manage UI states & cache responses
State Management
SMILE'S String Input
""REST/ GraphQL""

🧪 Polymer Prediction System Architecture

🚀 **Deployed on Hugging Face Spaces** - Streamlit Frontend with Integrated ML Pipeline

🖥️ Presentation Tier

Streamlit Web Interface
Interactive user interface for SMILES input and molecular visualization

Streamlit, HTML/CSS, JavaScript

SMILES Input Handler
Validates and processes molecular structure inputs

RDKit, Streamlit Forms

Visualization Engine
Renders molecular structures and prediction results

Matplotlib, Plotly, RDKit

Results Dashboard
Displays polymer property predictions and confidence scores

Streamlit Components

→

🤖 ML Processing Tier

Feature Extraction Service
ChemBERTa embeddings and molecular descriptors generation

Transformers, RDKit, PyTorch

Feature Engineering Pipeline
Combines embeddings with molecular features and applies PCA

Scikit-learn, NumPy, Pandas

Transformer Model Engine
2-layer transformer encoder with attention mechanism

PyTorch, Transformers

Regression Head
Final prediction layer with confidence estimation

PyTorch, Statistical Models

Model Management
Model loading, versioning, and performance monitoring

Hugging Face Hub, MLflow

→

🗄️ Data Tier

MongoDB Database
Stores user information and prediction history

MongoDB Atlas

User Management
Authentication and user session management

MongoDB, Session Tokens

Prediction Cache
Caches recent predictions for improved performance

Redis, MongoDB

Training Data Repository
Polymer dataset storage and version control

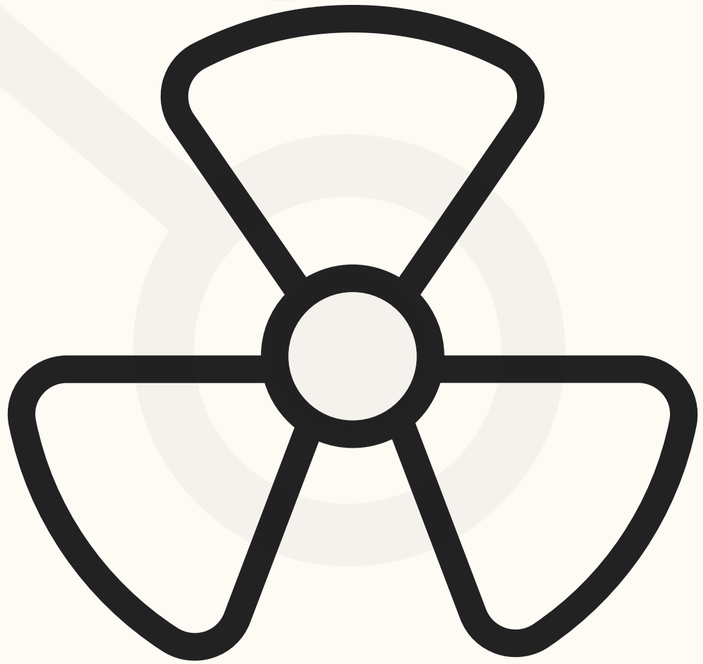
MongoDB GridFS, Hugging Face Datasets

Analytics & Logging
Usage analytics and system performance logs

MongoDB, Streamlit Analytics

CONCLUSION

We built a robust ensemble regression model to predict polymer properties from SMILES strings using ChemBERTa embeddings, RDKit descriptors, and Morgan fingerprints. By combining transformer-based learning, feature fusion, and randomized SMILES augmentation, our model achieved high predictive accuracy across multiple properties. This pipeline offers a powerful tool for data-driven polymer design and discovery



THANK YOU