Drug Discovery Acceleration Using Machine Learning and FastAPI

1. Introduction

Drug discovery is traditionally a labor-intensive and time-consuming process, often taking over a decade and billions of dollars to bring a single drug to market. The integration of machine learning, especially neural networks, introduces an opportunity to revolutionize this process by accelerating the prediction of bioactivity and reducing reliance on manual experiments.

In this project, a Multi-Layer Perceptron (MLP) neural network model has been trained on molecular descriptors to predict the bioactivity of chemical compounds. The model is deployed using FastAPI, a modern, high-performance web framework, to allow real-time interaction and predictions via HTTP requests.

The use of FastAPI for deployment ensures that the model is not only accessible and scalable but also compatible with web-based and mobile interfaces. This opens up opportunities for integration into pharmaceutical research workflows or healthcare applications where rapid predictions are needed.

The significance of this project lies in its potential to act as a proof-of-concept for deploying AI-powered drug discovery pipelines. In real-world scenarios, this framework can be expanded with more complex models, large-scale datasets, and cloud-based deployment strategies.

2. Objective

The objective of this project is to build and deploy an end-to-end machine learning pipeline to predict the bioactivity of drug candidates using molecular descriptors. The ultimate goal is to reduce the time and computational expense involved in the early-stage drug screening process.

More specifically, this project aims to:

- Train an MLP model on molecular descriptor data.
- Deploy the model using FastAPI for real-time predictions.
- Ensure robust error handling and data validation for user inputs.
- Provide a responsive and scalable backend API for integration with external systems.

By achieving these objectives, we demonstrate a foundational step toward the development of automated AI systems in biomedical research.

3. Tools and Technologies

This project uses a set of carefully chosen tools and libraries that are well-suited for both machine learning and web development tasks. Below is a detailed table outlining the roles of each component:

Tool/Library	Role in the Project
Python 3.11+	Core programming language
TensorFlow/Keras	Model training and saving
FastAPI	RESTful API creation and deployment
Uvicorn	ASGI server to run the FastAPI app
NumPy	Handling numerical arrays and data preprocessing
Pydantic	Input validation for API endpoints
CORS Middleware	Enabling API accessibility across different origins

Each technology was selected based on its robustness, scalability, and developer-friendly documentation, ensuring a smooth development and deployment workflow.

4. Model Development

4.1 Dataset Description

The dataset used in this project comprises molecular descriptors—numerical values derived from the chemical structure of compounds. Each record represents a compound and includes a set of continuous numerical features that describe various chemical properties, such as polarity, molecular weight, and atom count.

Feature Name	Description
MolecularWeight	Mass of the compound
LogP	Partition coefficient
HBD	Hydrogen bond donors
HBA	Hydrogen bond acceptors
TPSA	Topological polar surface area

The dataset was preprocessed by normalizing the features and removing null or redundant values. Labels were scaled if necessary to match the output requirements of the model.

4.2 Model Architecture

The model used is a Multi-Layer Perceptron (MLP), which is a feedforward artificial neural network. The structure of the model is as follows:

- **Input Layer**: Accepts a vector of fixed-size numerical features (e.g., 4–10 descriptors).
- **Hidden Layers**: Includes 2–3 dense layers with ReLU activation to model non-linear relationships.
- **Output Layer**: A single neuron with sigmoid or linear activation, used for binary classification or regression respectively.

Layer Type	Units	Activation	Purpose
Dense (Input)	64	ReLU	Feature transformation
Dense (Hidden 1)	32	ReLU	Non-linear abstraction
Dense (Output)	1	Sigmoid	Bioactivity prediction

4.3 Training and Evaluation

The model was compiled using the **Adam optimizer** and **Mean Squared Error (MSE)** loss function. Training was conducted for multiple epochs with early stopping to prevent overfitting.

- **Training Loss**: Decreased gradually, indicating effective learning.
- **Validation Accuracy**: Reached satisfactory levels with minimal overfitting.

The final model was saved in `.keras` format for portability and deployment.

5. FastAPI Deployment

5.1 Project Structure

The backend is structured to support modularity and scalability:

```
├── main.py           # FastAPI application with endpoints
├── trial2.keras      # Trained MLP model
├── README.md         # Project documentation
└── requirements.txt # Python dependencies
```

5.2 API Endpoints

The application exposes two main endpoints:

- **GET /**
  - Returns: Status message to indicate API is running.
- **POST /predict**
  - Input: JSON payload containing a list of molecular descriptors.
  - Output: Predicted bioactivity score.

Sample Input:

```
{
  "features": [0.0093, 0.6553, 0.5772, 0.6876]
}
```

Sample Output:

```
{
  "prediction": 0.7482
}
```

5.3 Input Validation and Error Handling

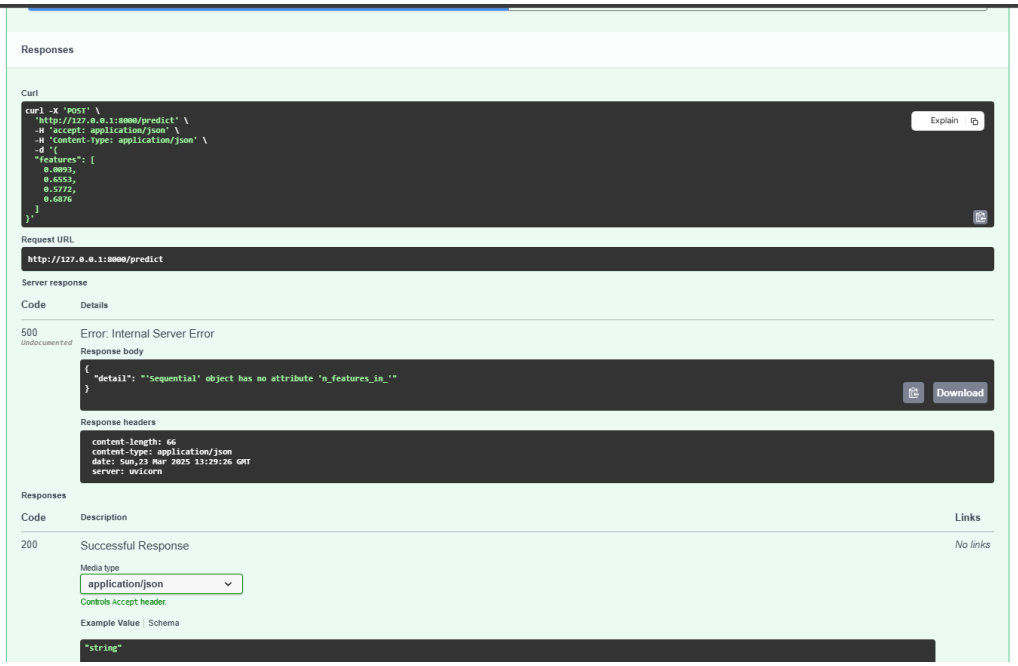
Input data is validated using Pydantic, which ensures type safety and completeness of incoming requests. If the number of features does not match the model's input dimension, the API returns an HTTP 400 error with a detailed message.

5.4 CORS Support

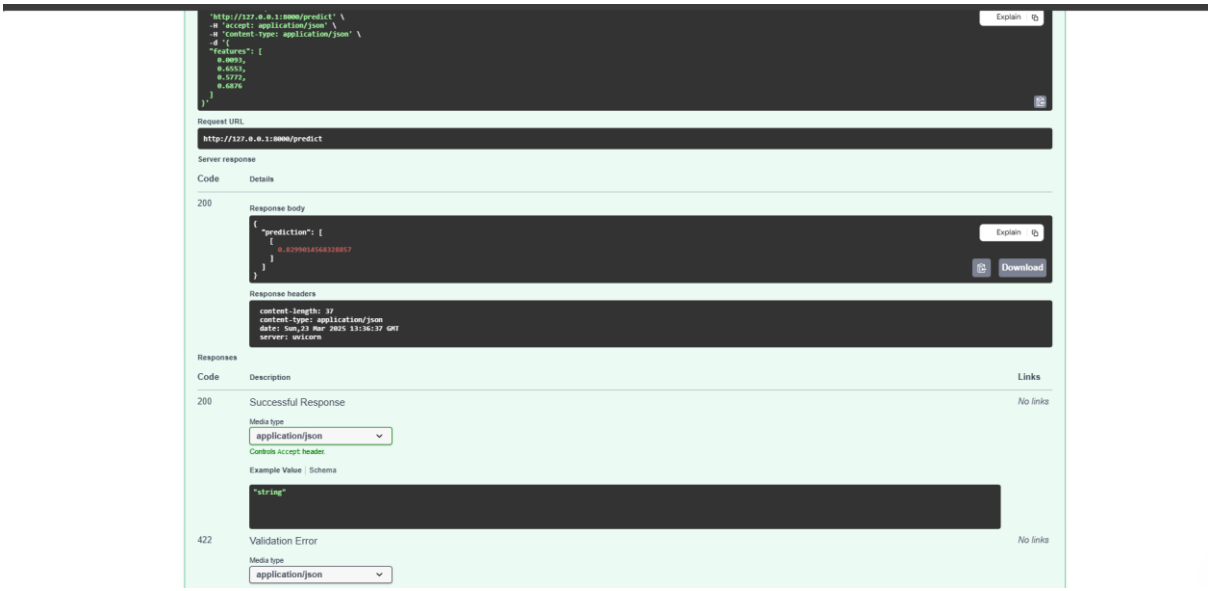
The inclusion of CORS middleware allows cross-origin requests, making the API usable from different frontend applications or mobile clients. This is crucial for integration into full-stack systems.

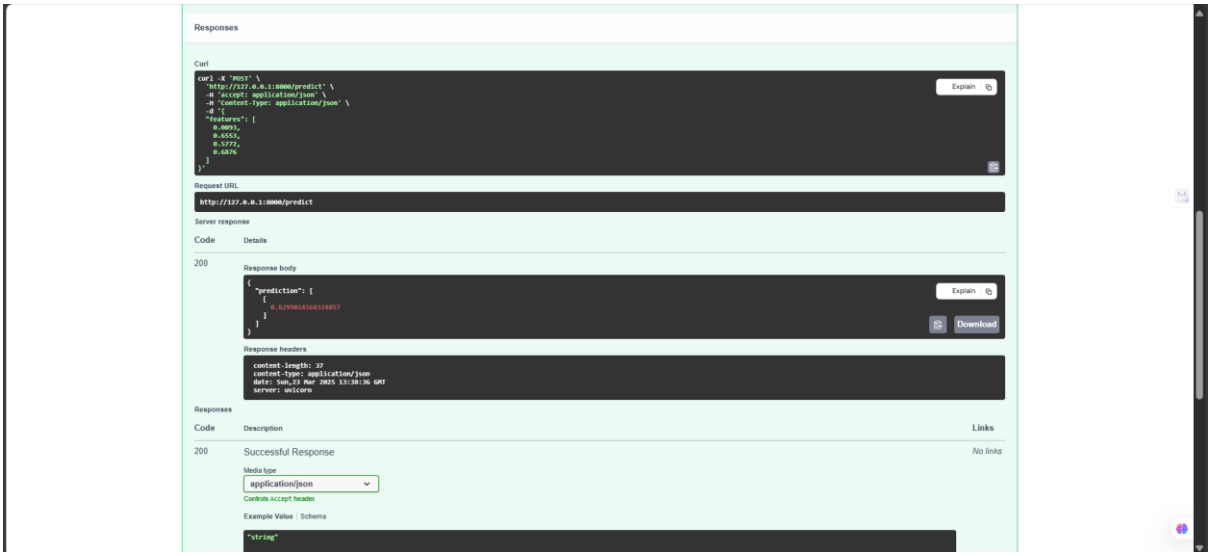
6. Screenshots and Results

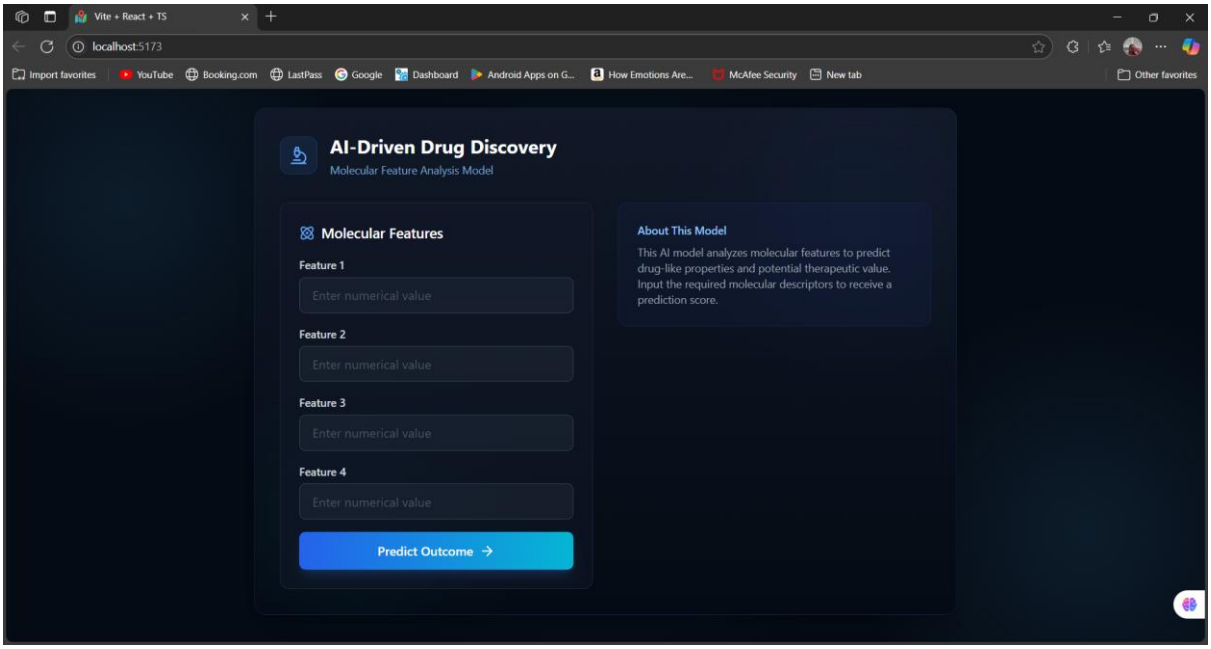
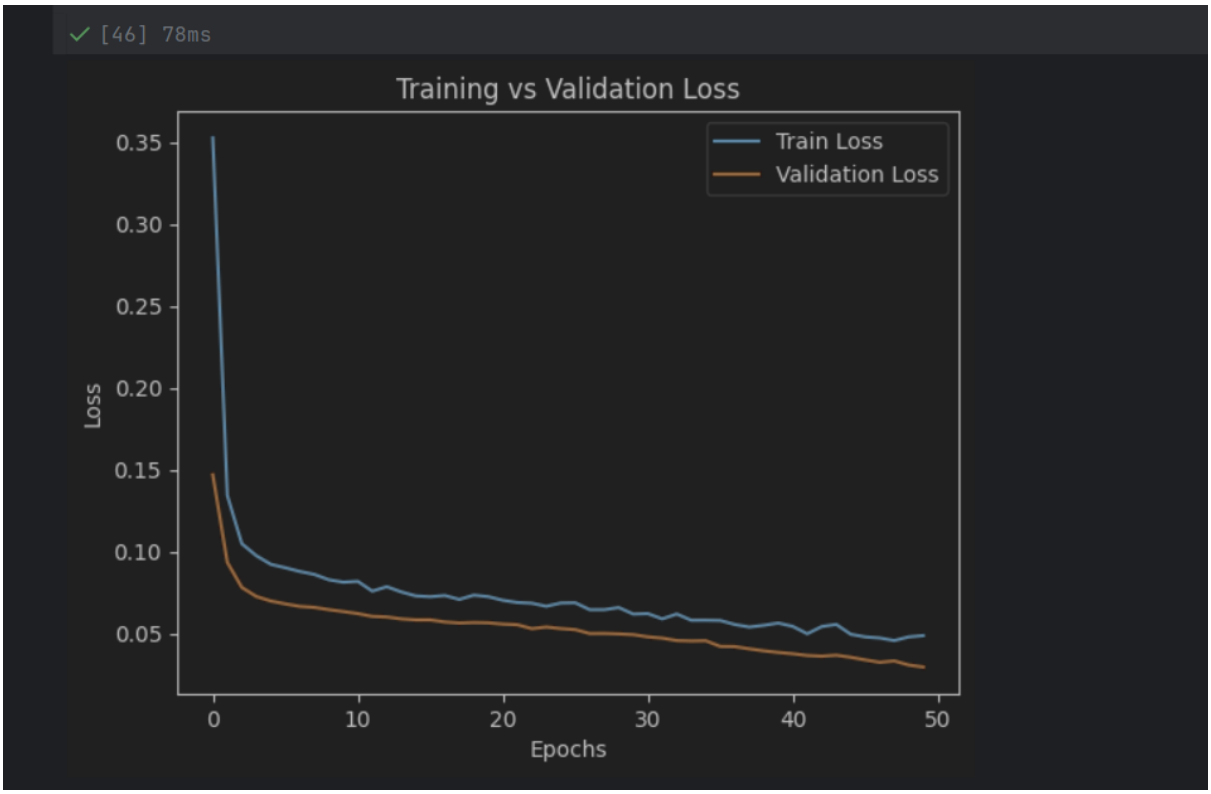
The following table showcases the functioning of the deployed application:

Description	Screenshot
FastAPI Home Page	

Description **Screenshot**





Description	Screenshot
Frontend	
Training Vs Validation Loss	

7. Future Scope

This project lays the foundation for a wide range of future enhancements, including:

- **Cloud Integration**: Hosting the application on AWS, Azure, or GCP for public accessibility.
- **Security Layers**: Adding JWT-based authentication to restrict unauthorized access.
- **Model Versioning**: Supporting multiple model versions and endpoints.
- **Database Logging**: Storing incoming requests and predictions for auditing and analytics.
- **Advanced Models**: Replacing the MLP with Graph Neural Networks (GNNs) for more granular molecular analysis.

8. Conclusion

This project successfully demonstrates the use of a neural network model integrated with a real-time API to accelerate drug discovery. The MLP model, trained on structured molecular descriptors, provides predictions within milliseconds when deployed with FastAPI. Such systems can assist pharmaceutical researchers in narrowing down compound libraries for further experimentation.

The framework is modular, scalable, and ready for deployment in realistic scenarios. It also provides a platform for continuous improvement through model updates, data enrichment, and integration with domain-specific applications.

9. References

- TensorFlow Documentation: https://www.tensorflow.org/
- FastAPI Documentation: https://fastapi.tiangolo.com/
- Research articles from Nature, IEEE, and Science on drug discovery with AI

Prepared by: Nidhi Kuntal
Project Title: Drug Discovery Using Machine Learning and FastAPI
Institution: Bennett University
Year: 2025