Predicting Molecular Properties using Classical Machine Learning and Graph Neural Networks

Aryan Kayande
SURP-2025 Nomination Assignment

June 5, 2025

Abstract

In this report, we address the task of predicting the log solubility of molecules based on their SMILES representations. We employ both classical machine learning models and a Graph Neural Network (GNN) approach to compare their performance on this regression task. Extensive exploratory data analysis (EDA) was performed to understand the molecular fingerprint data. The final models were evaluated based on their regression metrics on a held-out test set.

1 Introduction

The solubility of a molecule is a crucial property in drug discovery and materials science. Accurately predicting this property from molecular structures can significantly reduce the time and cost of experimental testing. In this work, we utilize:

- Morgan Fingerprints derived from SMILES strings using RDKit.
- Classical regression models: Random Forest, XGBoost, and LightGBM.
- A Graph Neural Network model implemented with DeepChem.

2 Methodology

2.1 Data Preprocessing

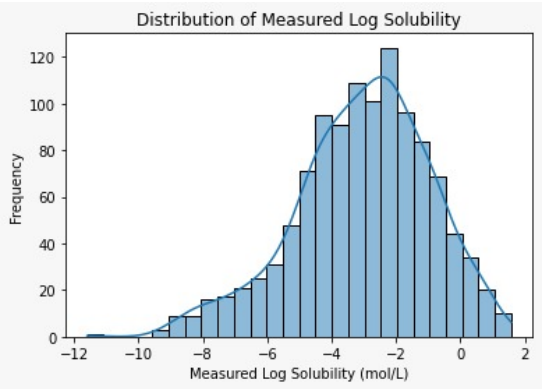
The dataset provided contained SMILES strings and their corresponding log solubility values. The preprocessing steps included:

1. Conversion of SMILES strings to Morgan Fingerprints using RDKit.
2. Transformation of the dataset into NumPy arrays suitable for scikit-learn and DeepChem models.
3. Train-test split: 75% for training and 25% for testing.

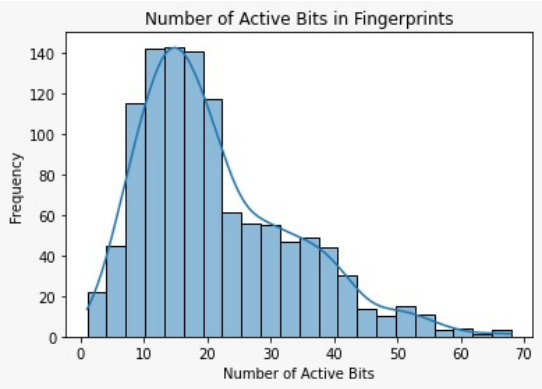
2.2 Exploratory Data Analysis (EDA)

Key insights from EDA included:

- Distribution of log solubility values.
- Sparsity and dimensionality characteristics of Morgan fingerprints.
- Correlation analysis to identify highly predictive fingerprint bits.



(a) Log solubility frequency distribution



(b) Number of active bits frequency distribution

Figure 1: EDA

2.3 Baseline Model Definitions

The following models were implemented using scikit-learn, XGBoost, and LightGBM:

- **Random Forest Regressor**: with hyperparameter tuning for number of estimators, max depth, and min samples split.
- **XGBoost Regressor**: with tuning for learning rate, max depth, and number of estimators.
- **LightGBM Regressor**: similarly tuned for learning rate, max depth, and boosting iterations.

Performance metrics recorded:

- Root Mean Squared Error (RMSE)
- R^2 Score

2.4 Graph Neural Network (GNN) Model

A simple GNN model was implemented using DeepChem's `GraphConvModel`:

- SMILES strings converted to molecular graphs using DeepChem's `ConvMolFeaturizer`.
- GraphConv layers with specified number of hidden features.
- Trained using Adam optimizer with appropriate learning rate and batch size.

3 Results

3.1 Performance Metrics

The models were evaluated on the test set using the following metrics:

Model	RMSE	R^2 Score
Random Forest	[1.421]	[0.696]
XGBoost	[1.279]	[0.726]
LightGBM	[1.117]	[0.733]
GNN (DeepChem)	[1.881]	[0.635]

Table 1: Performance comparison of different models on the test set.

4 Discussion

The performance results in Table 1 highlight that classical ensemble models demonstrated superior predictive capability compared to the Graph Neural Network (GNN) implemented via DeepChem for this molecular solubility prediction task. Among the baseline models, **LightGBM** achieved the lowest RMSE of **1.117** and the highest R^2 score of **0.733**, marginally outperforming both Random Forest and XGBoost. This indicates that LightGBM's gradient-based one-sided sampling and histogram-based optimisation strategies were well-suited for handling the sparse, high-dimensional fingerprint data derived from Morgan Fingerprints.

In contrast, the **GNN model underperformed relative to the baseline models**, with an RMSE of **1.881** and an R^2 score of **0.635**. This can likely be attributed to the relatively simple architecture of the GNN used, limited hyperparameter tuning, or potentially the dataset size not being sufficient to fully leverage the expressive power of graph-based models. Moreover, Morgan fingerprints inherently encode chemical substructures in a manner that is highly effective for tree-based ensemble methods, providing them with a natural advantage in this context.

Overall, the results suggest that while GNNs offer a conceptually powerful framework for modelling molecular graphs directly, classical machine learning models, particularly **LightGBM**, remain highly competitive baselines for structured molecular property prediction tasks using fingerprint-based features. Future work could explore deeper GNN architectures, advanced feature extraction strategies, or hybrid models to further improve predictive performance.

5 Challenges Faced

Firstly, I read about the different models that I needed to use, like XGBoost, Random Forests and GNNs (from ChatGPT, GeeksforGeeks and other similar sources). Then, I tried to code them out, which I found be pretty straightforward. The main issues that I faced were version mismatch on my device in Deepchem and Tensorflow *during the implementation of GNNs. I resolved that by upgrading/downgrading the versions of multiple libraries on my device. Another issue I faced was that the GNN I trained showed worse results than the baseline model used, which was counterintuitive. I checked

on the net and with ChatGPT, and what I figured out was that for smaller datasets (1200 points, like the one we used), it is quite possible that GNNs give worse results than the baseline models. I further tried out optimising the GNNs by hyperparameter tuning, etc. I again faced the version compatibility issue and currently I'm in that process of tuning and making the results better.