Drug Discovery Using Graph Neural Networks for Regression: A Comprehensive Project Report

Saksham Saharia
23B1078

February, 2025

Contents

1 Introduction

In the modern era of drug discovery, the ability to predict molecular properties accurately is critical. Traditional computational methods have often struggled to capture the complex interactions within molecular structures, limiting their effectiveness. This project leverages Graph Neural Networks (GNNs) to tackle a regression problem: predicting a continuous molecular property, which is an essential step in drug design and optimization.

Molecules can naturally be represented as graphs, where atoms are nodes and chemical bonds are edges. This representation preserves the underlying structure and connectivity inherent in chemical compounds. By applying GNNs, which are adept at handling graph-structured data, the project seeks to improve upon traditional methods by providing a more nuanced prediction mechanism that directly leverages molecular structure.

This report documents the end-to-end process from data preprocessing to model training and evaluation, with detailed discussions on methodology, experimental results, and future research directions.

2 Background and Motivation

2.1 Drug Discovery Landscape

Drug discovery is a complex, multi-disciplinary field that integrates chemistry, biology, and computer science. Traditionally, the process involves screening vast libraries of compounds to identify those with potential therapeutic effects. However, experimental methods are time-consuming and costly. Consequently, computational methods have been developed to predict the properties of molecules before synthesizing and testing them in the lab.

2.2 Graph Neural Networks in Chemistry

Graph Neural Networks (GNNs) have emerged as a powerful tool in many domains due to their ability to model relationships between interconnected data points. In chemistry, molecules inherently exhibit a graph structure, making GNNs an ideal candidate for tasks such as property prediction, synthesis planning, and reaction prediction. The ability of GNNs to aggregate and transform node-level information allows them to capture both local and global structural information, which is crucial for understanding molecular properties.

2.3 Motivation for the Project

The primary motivation behind this project is to explore how GNNs can be effectively applied to regression tasks in drug discovery. By predicting continuous molecular properties, such as solubility or binding affinity, we can better prioritize compounds for further development. This work not only aims to build a robust model but also to provide insights into the preprocessing challenges and architectural choices required when working with molecular data.

3 Literature Review

The application of GNNs to molecular property prediction has seen significant advancements in recent years. Prior research has demonstrated the effectiveness of various GNN architectures including Graph Convolutional Networks (GCNs), Message Passing Neural Networks (MPNNs), and Graph Attention Networks (GATs).

3.1 Graph Convolutional Networks

GCNs extend the concept of convolution from grid-like data (such as images) to graph data. Kipf and Welling's work on semi-supervised classification laid the foundation for using convolution on graphs, which has been adapted for regression tasks in drug discovery.

3.2 Message Passing Neural Networks

MPNNs further refined the process by introducing a message-passing phase, where each node's features are updated based on information from its neighbors. This approach has been particularly effective in capturing the chemical context of atoms within a molecule.

3.3 Graph Attention Networks

GATs integrate attention mechanisms, allowing the model to weigh the importance of different nodes during aggregation. This has led to performance improvements in tasks where not all neighboring atoms contribute equally to the final property prediction.

The work by Mulugeta, which serves as the inspiration for this project, demonstrates a practical regression example using a GNN. By following this example, the project aims to replicate and extend these ideas, applying them in a comprehensive pipeline.

4 Data Processing and Graph Construction

4.1 Dataset Overview

For this project, an open-source molecular dataset such as QM9 was utilized. QM9 is a widely used dataset in computational chemistry, containing approximately 134,000 small organic molecules along with computed geometric, energetic, and electronic properties. The continuous property selected for regression was chosen based on its relevance to drug discovery and the challenge it poses to the model.

4.2 Molecular to Graph Conversion

The conversion of molecules into graph structures is a critical step. Each molecule is represented as an undirected graph where:

- **Nodes:** Represent atoms. Each node is annotated with features such as atomic number, hybridization state, aromaticity, and degree.
- **Edges:** Represent chemical bonds between atoms. Edges may include features like bond type (single, double, etc.) and whether the bond is aromatic.

Tools such as RDKit were employed for molecular parsing and feature extraction. The process involves:

1. **Molecule Parsing:** SMILES strings or molecular files (e.g., SDF) are converted into molecular objects.
2. **Feature Extraction:** Atom and bond properties are computed and stored.
3. **Graph Assembly:** Using libraries like NetworkX or directly interfacing with PyTorch Geometric, the molecule is structured into a graph that can be fed into the GNN.

4.3 Feature Normalization

Normalization of input features is essential to stabilize training and ensure that the model does not become biased by large variations in input scale. Both node and edge features were normalized using standard techniques such as min-max scaling or z-score normalization.

5 Model Architecture and Implementation

5.1 Graph Convolutional Layers

At the core of the implemented model are the graph convolutional layers. These layers perform localized aggregation of node features based on the graph structure. The general operation in a graph convolutional layer is given by:

$$\mathbf{h}_i^{(l+1)} = \sigma\left(\sum_{j \in \mathcal{N}(i)} W^{(l)} \mathbf{h}_j^{(l)} + b^{(l)}\right)$$

where:

- $\mathbf{h}_i^{(l)}$ is the feature vector of node i at layer l,
- $W^{(l)}$ and $b^{(l)}$ are learnable weights and biases,
- σ is an activation function (e.g., ReLU),
- $\mathcal{N}(i)$ denotes the neighborhood of node i.

5.2 Message Passing and Readout

In the message passing phase, each node collects "messages" from its neighbors. The aggregated messages are then used to update the node's representation. After several layers of message passing, a readout function aggregates node representations into a single graph-level embedding. Common readout functions include:

- Global Average Pooling,
- Global Max Pooling,
- Sum Pooling.

For this project, a sum pooling approach was used due to its ability to maintain information proportional to the number of atoms in a molecule.

5.3 Fully Connected Layers and Regression Output

Following the readout phase, the graph-level embedding is passed through one or more fully connected layers. These layers transform the aggregated representation into the final output—a continuous value representing the molecular property under investigation. Dropout layers and batch normalization were integrated to prevent overfitting and stabilize training.

5.4 Implementation Details

The model was implemented using PyTorch as the primary deep learning framework, with PyTorch Geometric handling the graph-specific operations. The modular design of the code allowed for experimentation with various architectures (e.g., changing the number of GNN layers or experimenting with different pooling mechanisms) with minimal adjustments.

6 Training and Experimental Setup

6.1 Loss Function and Optimizer

Given the regression nature of the task, the Mean Squared Error (MSE) loss function was selected:

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$

The Adam optimizer was used for training due to its adaptive learning rate properties, which are especially useful for complex models such as GNNs. Key hyperparameters included:

- **Learning Rate:** Tuned through experimentation.
- **Weight Decay:** Applied to regularize the model and prevent overfitting.
- **Batch Size:** Selected based on available GPU memory and model complexity.

6.2 Hyperparameter Tuning

A grid search strategy was initially adopted to explore a range of hyperparameters:

- **Number of GNN Layers:** Varied from 2 to 5.
- **Hidden Dimensions:** Tested with dimensions of 64, 128, and 256.
- **Dropout Rate:** Explored values between 0.2 and 0.5.

Validation performance was closely monitored to avoid overfitting and ensure that the model was learning generalizable features.

6.3 Training Process and Hardware

The training process involved multiple epochs until convergence, with early stopping based on validation loss. The experiments were conducted on a GPU-enabled environment to

accelerate the computations. Logging and checkpointing strategies were implemented to save model states and facilitate debugging.

7 Results and Analysis

7.1 Model Performance

The final GNN model achieved a competitive MSE on the validation set. Key observations include:

- **Convergence Behavior:** The training loss steadily decreased over epochs, and the validation loss reached a plateau, indicating convergence.
- **Generalization:** The model demonstrated good generalization on the hold-out test set, suggesting that the graph-based representation captures essential molecular features.

7.2 Comparative Analysis

When compared to baseline models (e.g., simple fully connected networks with molecular fingerprints as input), the GNN approach exhibited a lower MSE. This reinforces the hypothesis that leveraging the inherent graph structure of molecules provides richer, more predictive representations.

7.3 Visualization of Predictions

Visual tools such as scatter plots were used to compare predicted vs. actual molecular properties. The closeness of the points to the diagonal line in these plots provided a visual confirmation of the model's accuracy. Additionally, learning curves depicting training and validation loss over epochs further illustrated the training dynamics.

8 Challenges and Discussion

8.1 Data Preprocessing Complexities

One of the primary challenges encountered was the conversion of molecular data into graph structures. The variability in molecular sizes and the diversity of chemical features required careful preprocessing. Ensuring consistent node and edge feature representation across molecules was a non-trivial task.

8.2 Model Overfitting

Overfitting was observed in early experiments, particularly when using deeper networks without adequate regularization. The inclusion of dropout layers, along with a reduction in the number of layers, helped mitigate this issue. The balance between model complexity and generalization remains an ongoing challenge.

8.3 Computational Efficiency

GNNs are computationally intensive, especially when processing large datasets. Optimization techniques, such as mini-batch training and efficient graph sampling methods, were necessary to handle the computational load. Future work may explore more efficient implementations or approximate methods for scaling up.

8.4 Interpretability of Predictions

While the model produced accurate predictions, interpreting the learned representations remains challenging. Understanding which atomic or bond features contribute most significantly to the regression output is an area for further research, potentially involving techniques from explainable AI (XAI).

9 Conclusion

This project has successfully demonstrated the application of Graph Neural Networks to a regression problem in drug discovery. By representing molecules as graphs, the model was able to capture the complex interactions inherent in chemical compounds and provide accurate predictions of molecular properties.

Key achievements include:

- **Effective Graph Construction:** Transforming molecular structures into graph representations using RDKit and PyTorch Geometric.
- **Robust GNN Architecture:** Designing and implementing a GNN model that leverages convolutional layers, message passing, and pooling techniques.
- **Rigorous Evaluation:** Demonstrating improved performance over traditional methods through careful experimentation and analysis.

The work not only validates the utility of GNNs in computational chemistry but also opens up numerous possibilities for future research. Continued refinement and exploration of advanced techniques are expected to further bridge the gap between computational predictions and experimental validations in drug discovery.