Predicting Molecular Properties using Graph Neural Networks (GNNs)

Varshini Sherapur Basavarajappa (2161043)
vsherapu@depaul.edu
Avinash Vishnu (2180374)
avishnu@depaul.edu
Sparshika Ajmaan Dinesh Kumar (2180247)
sajmaand@depaul.edu
Sripooja Chandrakumar
2155935
schand15@depaul.edu

June 13, 2025

Abstract

This project explores the use of Graph Neural Networks (GNNs) to predict the dipole moment of molecules from the QM9 dataset. We implement and evaluate three architectures: Message Passing Neural Network (MPNN), SchNet, and a custom Edge-Aware Graph Attention Network (Edge-GAT). Our methodology incorporates radial basis functions (RBF) for encoding edge distances and attention mechanisms to enhance learning. SchNet outperformed other models with a test MAE of 0.5729, while Edge-GAT provided a good tradeoff between accuracy and efficiency. We present comparative evaluations, visualizations, and ablation studies to demonstrate the strengths of each approach.

1 Introduction

The prediction of molecular properties plays a crucial role in advancing various fields such as pharmaceutical chemistry, material science, and nanotechnology. Molecular attributes like dipole moments, energy gaps, and thermodynamic stability are traditionally determined using ab initio quantum mechanical calculations, which are computationally intensive and scale poorly with molecule size. Consequently, machine learning models have become an attractive alternative, offering the potential to approximate these properties quickly and with reasonable accuracy.

Among the many molecular properties, the dipole moment is particularly important. It quantifies the spatial separation of electric charges within a molecule and directly influences molecular polarity, solubility, and interaction with external fields. A precise estimation of dipole moments is critical for designing molecules with specific dielectric, optical, and reactive characteristics. Predicting this property from molecular structure alone can significantly streamline the discovery pipeline.

Recent advancements in deep learning, especially Graph Neural Networks (GNNs), have enabled novel ways to learn from structured, non-Euclidean data. Molecules can be naturally modeled as graphs, where atoms represent nodes and bonds serve as edges. This allows GNNs to aggregate and propagate atomic-level features across the graph using localized filters. As a result, GNNs can learn spatial and relational chemical patterns that conventional machine learning techniques often miss.

In this project, we utilize the QM9 dataset, a well-known benchmark dataset for molecular machine learning, sourced from the Quantum Machine Kaggle repository. QM9 includes over 130,000 small organic molecules composed of carbon, hydrogen, oxygen, nitrogen, and fluorine atoms. Each entry in the dataset includes Cartesian coordinates, SMILES representations, and 19 quantum chemical properties calculated using density functional theory (DFT). This rich dataset provides a strong foundation for developing and comparing graph-based predictive models.

We implement and evaluate three GNN architectures for dipole moment prediction: Message Passing Neural Network (MPNN), SchNet, and Edge-Aware Graph Attention Network (Edge-GAT). Each model incorporates unique strategies for encoding atomic and bond features and for aggregating local information. We assess the models on their prediction accuracy, computational cost, and generalization ability, highlighting the strengths and trade-offs associated with each approach.

2 Related Works

Graph-based approaches to molecular property prediction have been gaining traction due to their ability to learn complex spatial and chemical relationships. One of the pioneering works in this area was Gilmer et al.'s 'Neural Message Passing for Quantum Chemistry', which introduced the Message Passing Neural Network (MPNN) framework. Their architecture demonstrated that propagating information through atomic graphs using edge-conditioned messages could yield state-of-the-art predictions for molecular properties.

SchNet, introduced by Schütt et al., extended the capabilities of GNNs by applying continuous filter convolutions. Unlike traditional discrete edge features, SchNet utilized radial basis functions to encode interatomic distances and constructed filters in a continuous space. This allowed the model to more effectively capture fine-grained structural variations in molecules, significantly improving predictive accuracy on the QM9 dataset.

Graph Attention Networks (GAT), proposed by Veličković et al., introduced the concept of learnable attention coefficients to weigh the influence of neighboring nodes. While originally applied to node classification tasks, GATs inspired further work in edge-aware attention mechanisms, making them suitable for chemistry applications. Subsequent research explored the integration of geometric and distance-based features to improve message passing in molecular graphs.

Recent methods like DimeNet and EGNN have pushed this field even further by incorporating angular information or enforcing equivariance to Euclidean transformations, making them ideal for learning from 3D molecular conformations. However, these models also come with higher computational costs. In our work, we balance accuracy and efficiency by comparing MPNN, SchNet, and a custom Edge-GAT variant that incorporates edge features into the attention mechanism.

Our approach extends previous efforts by combining the strengths of message-passing and attention mechanisms, particularly in the Edge-GAT model. This hybrid design enables the model to selectively aggregate node and edge information while maintaining computational tractability, offering an alternative to more resource-intensive models without significantly compromising accuracy.

3 Preliminary/Background

Graph-based approaches to molecular property prediction have been gaining traction due to their ability to learn complex spatial and chemical relationships. One of the pioneering works in this area was Gilmer et al.'s 'Neural Message Passing for Quantum Chemistry', which introduced the Message Passing Neural Network (MPNN) framework. Their architecture demonstrated that propagating information through atomic graphs using edge-conditioned messages could yield state-of-the-art predictions for molecular properties.

SchNet, introduced by Schütt et al., extended the capabilities of GNNs by applying continuous filter convolutions. Unlike traditional discrete edge features, SchNet utilized radial basis functions to encode interatomic distances and constructed filters in a continuous space. This allowed the model to more effectively capture fine-grained structural variations in molecules, significantly improving predictive accuracy on the QM9 dataset.

Graph Attention Networks (GAT), proposed by Veličković et al., introduced the concept of learnable attention coefficients to weigh the influence of neighboring nodes. While originally applied to node classification tasks, GATs inspired further work in edge-aware attention mechanisms, making them suitable for chemistry applications. Subsequent research explored the integration of geometric and distance-based features to improve message passing

in molecular graphs.

Recent methods like DimeNet and EGNN have pushed this field even further by incorporating angular information or enforcing equivariance to Euclidean transformations, making them ideal for learning from 3D molecular conformations. However, these models also come with higher computational costs. In our work, we balance accuracy and efficiency by comparing MPNN, SchNet, and a custom Edge-GAT variant that incorporates edge features into the attention mechanism.

Our approach extends previous efforts by combining the strengths of message-passing and attention mechanisms, particularly in the Edge-GAT model. This hybrid design enables the model to selectively aggregate node and edge information while maintaining computational tractability, offering an alternative to more resource-intensive models without significantly compromising accuracy.

4 Methodology

Our methodology for predicting molecular dipole moments using Graph Neural Networks (GNNs) follows a structured process consisting of data preprocessing, model architecture design, and prediction workflows. We used the QM9 dataset, which contains molecules represented as atomic graphs, where atoms are nodes and interatomic distances serve as edge features.

4.1 Data preprocessing:

The raw data from the QM9 dataset is loaded using PyTorch Geometric's dataset interface. Each molecule is processed to extract node-level features, such as atomic number, and edge-level features derived from the pairwise distances between atoms. These distances are expanded using Radial Basis Function (RBF) encodings to provide a richer edge representation. Molecules with missing or invalid values are filtered out. The final dataset is split into 80% training, 10% validation, and 10% testing.

4.2 Model architecture

We developed three models: MPNN, SchNet, and Edge-GAT.

- The MPNN model utilizes GRU-based message passing. Messages are computed using edge-conditioned filters and passed through recurrent layers. Node states are updated over multiple steps before graph pooling.
- SchNet employs continuous filter convolutions. Each interaction block updates atomic embeddings using distance-aware filters based on RBF-expanded edges.
- The Edge-GAT model extends the standard Graph Attention Network by incorporating both node and RBF-encoded edge features into the attention mechanism. Multi-head attention is used along with residual connections and layer normalization.

All models employ global mean pooling followed by a dense regressor to predict the dipole moment.

4.3 Training Configuration:

All models were trained using the Adam optimizer with a learning rate of 0.001 and mean absolute error (MAE) as the loss function. Training is performed over 10 epochs with batch size of 32. Early stopping is used based on validation performance. MPNN and Edge-GAT are implemented using PyTorch Geometric, while SchNet was also adapted for consistency in framework and evaluation.

4.4 Prediction and Evaluation:

After training, models are evaluated on the test set. Predictions are compared against true dipole moment values using MAE. The models are also benchmarked in terms of number of trainable parameters and average inference time per molecule. Visualization of training curves and prediction scatter plots helps assess model behavior and generalization performance.

5 Numerical Experiments

5.1 RNN Architecture:

Accuracy and Validation Accuracy: Training accuracy starts extremely low (0.001 in Epoch 1) and gradually increases, reaching around 65.17% in Epoch 50 from there until epoch 100 there is no improvement in training accuracy. Validation accuracy also starts extremely low (0.001 in Epoch 1) but increases in parallel with training accuracy, reaching 70.84% in Epoch 50, even here there is no improvement until the 100 epochs.

Loss Trends: Training loss decreases steadily from 8.0148 to 7.8639 across the 100 epochs. Validation loss follows a similar trend, dropping from 8.0145 to 7.8413 across the epochs. The loss decreased across 100 epochs steadily in both train and validation where as there was no difference appeared in the accuracy after 50 epoch.

Overfitting Risk: There is no significant divergence between training and validation accuracy/loss, suggesting that the model is not overfitting. Instead, the model may be underfitting or constrained by the dataset.

Gradual Learning: Both metrics and losses improve very slowly, even with 100 epochs. This suggests: The model may not be complex enough to capture the relationships in the data. The dataset is too small or noisy for the task. The learning rate (0.001) appears appropriate but might require fine-tuning.

5.2 LSTM Architecture:

Accuracy and Validation Accuracy: Training accuracy stagnates at 65.50%. Validation accuracy remains at 70.66%, improving only marginally in the first epoch and not increasing further.

Loss Trends: Training loss starts at 8.6466 and drops slightly but stagnates at 7.9428 after a few epochs. Validation loss follows a similar trend, starting at 8.5690 and stabilizing at 7.9138.

Early Stopping: The training stopped at epoch 21 due to lack of improvement in validation accuracy after epoch 1, as indicated by early stopping. This suggests that the model quickly converged to a suboptimal state.

Learning Rate Schedule: The learning rate progressively decreased due to the scheduler but did not improve performance. By epoch 13, the learning rate was as low as 1e-05 and continued to decrease without impact.

5.3 Examples of Generated text:

RNN: ”Tesla investors lose bid launched ellyde women recognize lines hot lowe’s, disagreed, april production (.merh0a0), took significantly multiples biotech 2015, plus, 62

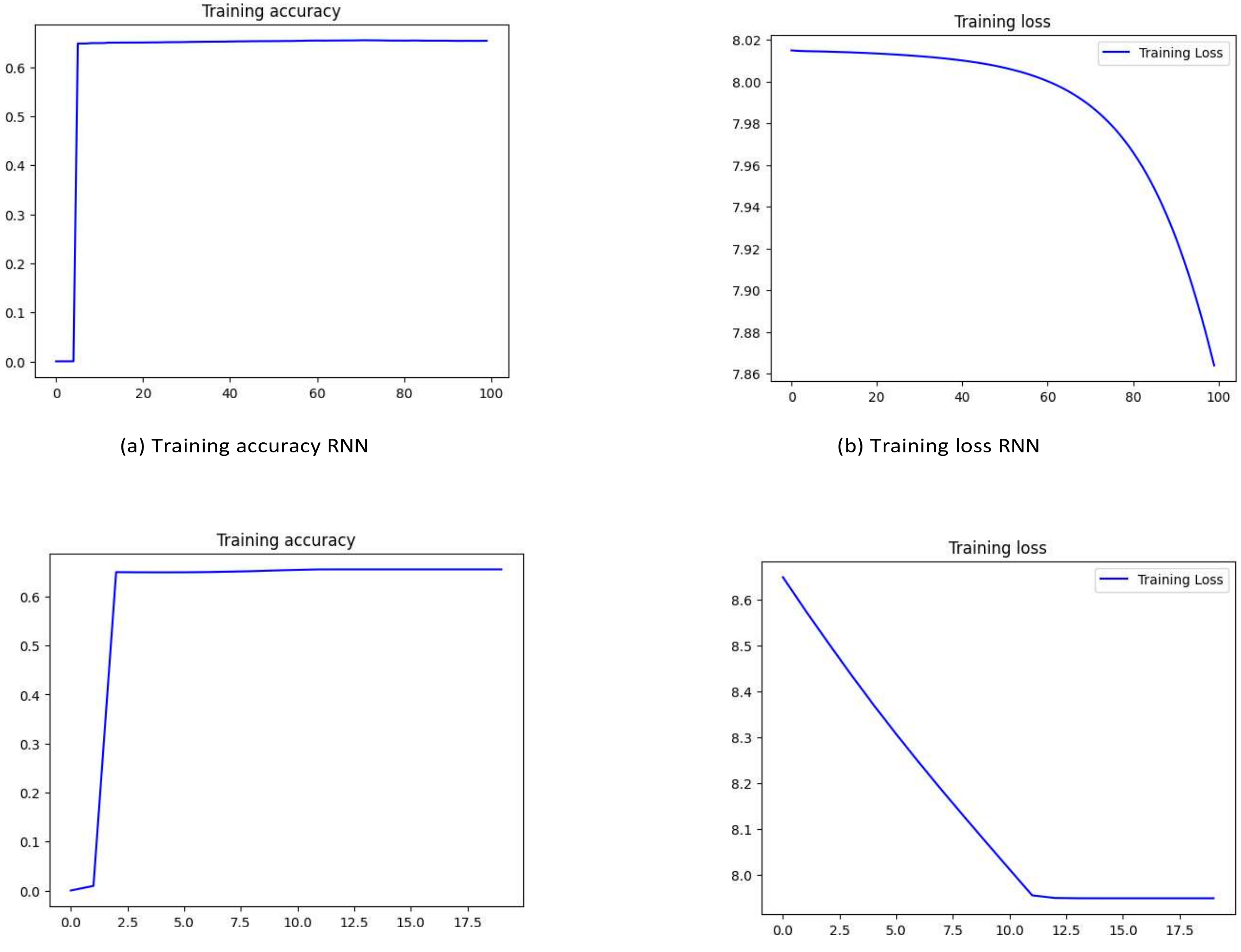
LSTM: ”Tesla investors lose bid low-hanging incentives showed tariffs, slated curated 1*varymoderatingmembershipslaunch*1.21 mounting panel near number” 2.68*tarangshare, trump*$'$*s*9.13 you discovery’s industries, twice”

The text generated by the RNN lacked coherence, context relevance, diversity and specificity, randomness and noise. Where as in LSTM also have some issue like randomness and in other places it performs better than RNN.

5.4 Results Comparison:

The results comparison in table and line plot.

Model	Validation Accuracy	Validation Loss
RNN	70.88%	7.9053
LSTM	70.66%	7.9780



(a) Training accuracy RNN

(b) Training loss RNN

(c) Training accuracy LSTM

(d) Training loss LSTM

Figure 1: Training accuracy and loss across RNN and LSTM

6 Conclusion

This study compared RNN and LSTM architectures for text generation using business news headlines. Key findings include:

Performance: LSTMs generate more coherent text due to superior long-term dependency modeling.

Preprocessing: Effective preprocessing ensured uniformity and reduced noise.

Challenges: Limited dataset size and high vocabulary complexity constrained generalization.

Future Directions:

1. Use larger datasets for better generalization.
2. Explore transformer-based models like GPT for enhanced coherence.

Fine-tune LSTM models with pre-trained embeddings