Atomization energies prediction

Nguyen Qui Vinh Quang
International University, Vietnam National University, Ho Chi Minh City, Vietnam
{nguyenquivinhquang}@gmail.com

August 7, 2023

1 Introduction

Quantum mechanics is a foundation theory in physics that helps us understand how things like molecules, atoms, and even tinier particles behave. But doing the math for these calculations can be super tough because they are really complex. Thankfully, machine learning has come to the rescue, giving us a great way to estimate these tricky quantum calculations and figure out their properties.

In this report, we apply machine learning models to estimate atomization energies from the QM7 dataset. We experiment with several machine learning models, including traditional machine learning methods (linear regression, support vector regression, kernel ridge regression, multilayer perceptron) and graph learning methods(graph convolutional network[1] and graph attention network [3]). The performance of the models is evaluated based on the mean absolute error (MAE) and follows the cross-validation from previous work [2].

The structure of this report is organized as follows. In Section 2, we provide a detailed description of the QM7 dataset. Section 3 discusses the data pre-processing steps, while Sections 4 explores the traditional machine learning models and graph neural networks used in this study. In Section 5, we present the implementation details, numerical results, and an analysis of the experiments. Section 6 concludes the report. The code implementation will be provided during the interview.

2 Dataset

2.1 Data description

The molecular dataset consists of a maximum of 23 atoms, including C, N, O, S, and H elements. There are 7,165 molecules in this dataset, which are characterized by five attributes: Coulomb matrix (C), atomization energies (P), atomic charge (Z), Cartesian coordinates (R), and cross-validation splits (P).

The Coulomb matrix is defined as:

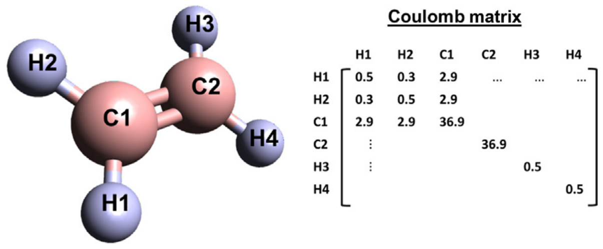
$$C_{ij} = \begin{cases} 0.5Z_i^{2.4} & \text{for } i = j \\ \frac{Z_i Z_j}{R_{ij}} & \text{for } i \neq j \end{cases} \quad (1)$$

In this equation, Z_i represents the nuclear charge of atom i, and R_i denotes its position. Notably, $C_{ij} \neq 0$ when atoms are present at positions i and j.

Furthermore, the dimensions of the arrays for C, T, P, Z, and R are $(7165 \times 23 \times 23)$, (7165), (5×1433), (7165×23), and $(7165 \times 23 \times 3)$, respectively.

2.2 Data analysis

This dataset includes a total of 7,165 molecules, but only 1,512 unique molecular formulas due to variations in Cartesian coordinates or Lewis structures. The most frequent molecular formula is C_6NH_3, occurring 112 times. For molecular formulas with multiple representations, atomic charges are maintained in the same order for consistency. Therefore, the invariant structure of each molecule does not need to be considered at this time.



$$\begin{array}{c|cccccc} & H1 & H2 & C1 & C2 & H3 & H4 \\ \hline H1 & 0.5 & 0.3 & 2.9 & \cdots & \cdots & \cdots \\ H2 & 0.3 & 0.5 & 2.9 & & & \\ C1 & 2.9 & 2.9 & 36.9 & & & \\ C2 & \vdots & & & 36.9 & & \\ H3 & \vdots & & & & 0.5 & \\ H4 & & & & & & 0.5 \end{array}$$

Figure 1: This is an example of Coulomb matrix representation of C_2H_4 molecule

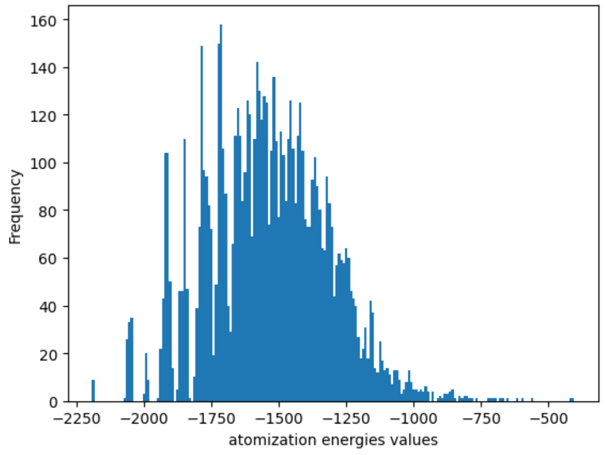


Figure 2: Atomization Energy Frequency

Figure 2 shows the frequency distribution of Atomization Energy, with minimum, maximum, and mean values of -2192.0, -404.88, and -1538.0377, respectively. The histogram indicates that Atomization Energy likely follows a normal distribution with some noise.

3 Pre-processing data

In this dataset, we have pre-processed the data to generate two distinct types of features:

1. Machine Learning Feature(T): This vector has a shape of $1 \times n$, where n signifies the total number of features. Each feature vector represents the attributes of each graph. It is specifically designed for machine learning approaches.
2. Graph Neural Network Features (Adjacency matrix M and Node feature vector F): The adjacency matrix M has a shape of (23×23) with values 0 or 1. An element $M_{ij} = 1$ if there exists an atom at positions i and j. The node feature vector F has a shape of $(1 \times t)$, where t represents the number of features of each node.

Eigenvalue: The eigenvalues of its adjacency matrix determine eigenvalues in a graph, and the collection of these eigenvalues is referred to as the graph spectrum. The spectral radius of a graph is the maximum absolute value of its eigenvalues. Additionally, the algebraic connectivity of a graph is denoted by the second smallest eigenvalue of its Laplacian matrix. Therefore, we use Eigenvalue as one of our features.

Eigenvector centrality: Eigenvector centrality is a measure of the influence of a node in a network. It assigns relative scores to all nodes in the network based on the principle that connections to high-scoring nodes contribute more to the score of the node in question than equal connections to low-scoring nodes.

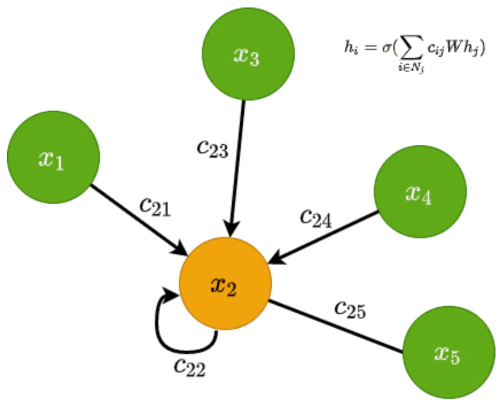
Atom coordinate: As discussed in Section 2.2, numerous molecules share the same molecular formula, implying they have identical atomic charges. The difference lies in the Cartesian coordinates of each atom within the molecule. Consequently, we believe that the Cartesian coordinates of each molecule should be taken into account as a feature. To this end, we utilize the mean and standard deviation of the atomic Cartesian coordinates in each molecule as a feature.

Sorted Coulomb matrix: One of the problems of the molecule matrix is the order of atoms is undefined in each molecule. Therefore, Sorted Coulomb Matrices will handle this problem litte bit. Sorted Coulomb Matrices will generate by choosing the permutation of atoms on the original Coulomb Matrices satisfies $\|C_i\| \geq \|C_{i+1}\| \, \forall i$ where C_i denotes the i^{th} row of the Coulomb matrix.

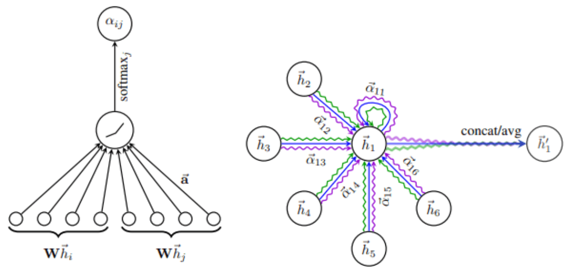
4 Method

4.1 Machine learning methods

Linear Regression (LR): We employ Linear Regression as our baseline model due to its simplicity and the assumption of a linear relationship between input features



(a) Message Passing in GNNs. Each neighboring node has an equal weight effect on the current node x_2.



(b) Left: The attention mechanism. Right: An illustration of multi-head attention on its neighborhood

Figure 3: Comparing Message Passing between Graph Convolutional Networks and Graph Attention Networks

and the target variable. This model serves as a reference point for comparison with other, more sophisticated models.

Kernel Ridge Regression (KRR): We choose KRR for its ability to handle non-linear relationships between molecular structure and properties. KRR captures these complex relationships by implicitly mapping input features to a higher-dimensional space using a kernel function. Furthermore, KRR has been previously applied to this dataset as reported in [2]. We aim to reimplement KRR and test our hypothesis concerning the significance of atomic coordinates.

Decision Tree Regression (DTR): A Decision Tree Regression can be applied to the QM7 dataset due to its interpretability, ability to capture non-linear relationships, and automatic feature selection.

Multilayer Perceptron: One of the key features of Multilayer Perceptrons (MLPs) is their capacity to learn internal representations that can enhance model efficiency both statistically and computationally. We hypothesize that by carefully designing the architecture of the MLP, it can achieve satisfactory results on the QM7 dataset. This approach can potentially capture complex, non-linear relationships between molecular structures and properties, leading to a more accurate and robust model.

4.2 Graph learning methods

In this report, we investigate two popular graph-learning networks: Graph Convolutional Networks (GCNs) and Graph Attention Networks (GATs).

Graph Convolutional Networks: GCNs learn node representations by aggregating information from neighboring atoms through convolutions. This process allows features to propagate across the graph. The final node representations are pooled to obtain a fixed-size molecular representation feature vector. This feature vector is then used in a fully connected layer to predict the atomization energies of the molecule.

Graph Attention Networks: Unlike the simple averaging used in GCNs for aggregating neighbor information, GAT layers learn distinct attention coefficients for different neighbors. This attention mechanism enables GATs to determine the relative importance of neighbors when updating a node's features, allowing for more nuanced and context-aware representations.

5 Experiments

5.1 Evaluation Metrics

We employ the Mean Absolute Error (MAE) as our evaluation metric, utilizing the 5-fold cross-validation provided by the dataset. The final performance for each model is calculated as the average MAE across the 5 splits.

5.2 Implementation Details

We use the scikit-learn library for implementing LR, KRR, and DTR. For MLP, GCN and GAT, we employ the PyTorch library.

5.2.1 Parameters

For KRR, we choose the Radial Basis Function (RBF) kernel with both alpha and gamma set to $1e^{-4}$. For Decision Tree Regression, we set the minimum samples split and minimum samples leaf to 2 and 1, respectively.

For the MLP, we utilize 5 linear layers followed by ReLU activation functions. Before the last layer, we apply a sigmoid activation function. We train the MLP for 1000 epochs using the AdamW optimizer.

For the GCN, we employ three GCNConv layers. After each GCNConv layer, we apply a ReLU activation function. At the end, we use a global mean pool and linear layer to generate the prediction. For the GAT, we follow a similar structure as the GCN, but we replace the GCNConv layers with GAT layers and use only 2 GAT layers. Both models are trained using the AdamW optimizer.

To ensure a fair evaluation, we reinitialize these models for each fold during cross-validation.

5.2.2 Features implementation

Machine Learning Feature Vector: We concatenate the eigenvalues, eigenvector centrality, atom coordinates, and the flattened sorted Coulomb matrix to create the feature vector. Furthermore, we normalize the atomization energies by dividing them by their maximum absolute value (which is 2192.0).

Graph Neural Network Feature Vector: For the adjacency matrix M, we set each element $M_{ij} = 1$ if $C_{ij} \neq 0$; otherwise, $M_{ij} = 0$. For the node feature vector F, each element F_i is a feature vector created by concatenating the centrality and coordinates of atom i. We do not use the Coulomb matrix for node features because the centrality already captures its information. Additionally, we use the Coulomb matrix as our edge feature for the graph neural network.

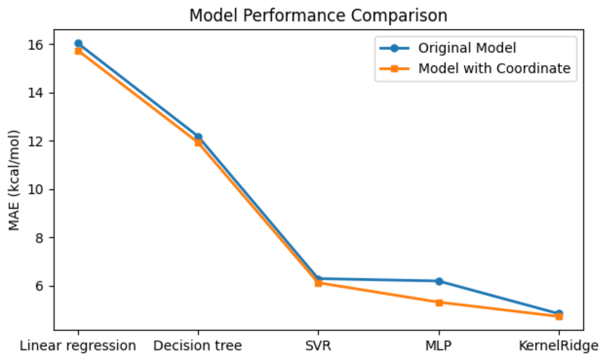


Figure 4: Average loss values for test models with coordinate and without coordinate features.

5.3 Experiments Results

5.3.1 Machine learning methods

Figure 4 illustrates the average loss of LR, DT, Support Vector Regression(SVR), MLP and KRR. The KRR model demonstrates the best performance, with a loss of merely 4.72585. The second-best model is the MLP, which achieves a loss of 5.31. However, it is important to note that the training time for MLP is significantly longer at 1691 seconds, while KRR takes only 15 seconds to train. Additionally, the figure highlights that including the atom coordinates feature leads to a reduction in the error for all models in the validation process.

In Table 1, we present an ablation study comparing the performance of the KRR model using different feature combinations. The results indicate that the introduction of the Sorted Coulomb matrix dramatically reduces the mean absolute error from 8.911 to 4.881. Furthermore, incorporating the Atom coordinate feature enables the model to achieve the lowest MAE, which supports our hypothesis discussed in Section 3. It is observed that as the number of features increases, the execution time also becomes longer. However, at 15 seconds, KRR remains the fastest model in this experiment.

Features	MAE	Time(s)
Only Coulomb matrix	8.911	9.667
+ Sorted Coulomb matrix	4.881	9.572
+ Eigenvalue	4.884	9.933
+ Eigenvector centrality	4.841	10.4696
+ Atom Coordinate	4.726	15.2015

Table 1: Ablation study of the feature combinations for the KRR model

5.3.2 Graph learning methods

Figure 5 presents the performance of GAT and GCN models in our experiment. GAT achieves a lower loss (39.932) compared to GCN (54.246). Throughout the experiment, GAT consistently outperforms GCN, indicating that the attention mechanism enhances the robustness of node features in this dataset. Another advantage of GAT over GCN is its ability to utilize edge features, which GCN does not support.

However, the performance of both graph learning methods is inferior to the machine learning models tested in our experiments. One reason for this discrepancy could be the limited number of features available for each node, making it difficult to predict atomization energy accurately. Another reason could be the insufficient training data, as graph learning with convolution requires a large amount of training data, but our dataset only contains 5732 molecules for training. Therefore, further investigation is needed to evaluate the performance of graph learning methods with more training data and additional node features.

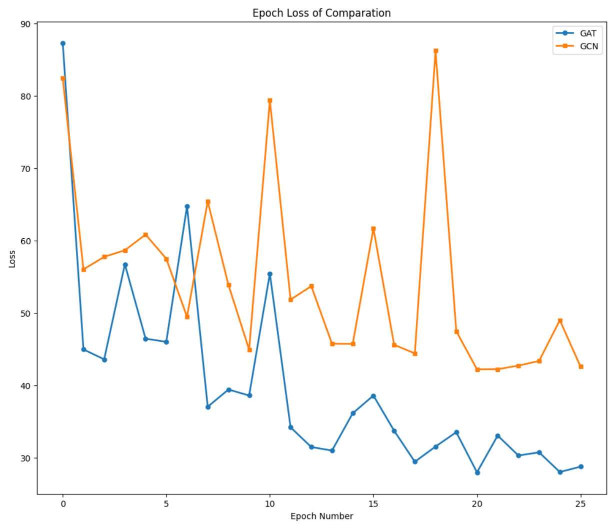


Figure 5: Comparison of average loss values for the validation set during training between GAT and GCN

Method	MAE
Linear regression	15.734
Decision tree	11.926
Support Vector Regression	6.119
Multilayer Perceptron	5.318
Kernel Ridge Regression	4.726
Graph Convolutional Network	54.246
Graph Attention Network	39.932

Table 2: The final result

6 Conclusion

In this study, we explored various machine learning and graph learning methods to predict atomization energy based on molecular graph representations. Our experiments demonstrated the importance of feature engineering, as the incorporation of additional features, such as the Sorted Coulomb matrix and Atom coordinates, led to significant improvements in the model's performance.

Among the machine learning models, Kernel Ridge Regression exhibited the best performance, achieving the lowest mean absolute error. In contrast, the graph learning methods, Graph Attention Network and Graph Convolutional Network, showed inferior performance compared to the machine learning models. This outcome could be attributed to the limited number of features for each node and the small training dataset, which hindered the robustness of the graph learning models.

Future research should focus on expanding the training dataset and exploring additional node features to enhance the performance of graph learning methods. Moreover, investigating other graph learning techniques and potential modifications to existing methods might reveal promising approaches for predicting atomization energy from molecular graph representations.

References

[1] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. 1

[2] M. Rupp, A. Tkatchenko, K.-R. Müller, and O. A. von Lilienfeld. Fast and accurate modeling of molecular atomization energies with machine learning. *Physical Review Letters*, 108:058301, 2012. 1, 3

[3] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. *6th International Conference on Learning Representations*, 2017. 1