Machine Learning for HOMO–LUMO Gap Prediction and Inverse Molecular Design

Authors:
Thibo Van Eeckhoorn, Yarno De Jaeger and Mattice Criel

Ghent University

Academic Year: 2025–2026

1 Introduction

Still need to add the references An important property of chemical systems, including organic molecules, is their HOMO-LUMO gap which is the energy difference between the highest occupied molecular orbital (HOMO) and lowest unoccupied molecular orbital (LUMO). This is a fundamental descriptor of the electronic structure of a molecule. It governs key properties such as reactivity, optical absorption and charge transport characteristics. As a consequence, accurate determination of the HOMO-LUMO gap is essential for molecular design in chemistry, materials science, and molecular engineering.

Experimentally, the HOMO-LUMO gap can be approximated through optical spectroscopy or voltammetry.[1] In UV-Vis absorption spectroscopy, electron excitation from the ground state (associated with the HOMO) to excited states (related to the LUMO) produces an absorption onset, or optical band edge, whose energy corresponds to the optical gap.[2] Similarly, photoluminescence spectroscopy captures the energy released during radiative relaxation. However, experimental measurements require synthesized, purified materials and are not suitable for large-scale exploration of hypothetical chemical structures. An alternative way to get the HOMO-LUMO gap is with computational chemistry, such as Hartree-Fock and Density functional theory (DFT). Despite their accuracy, these quantum chemical methods are computationally expensive and scale poorly with molecular size and dataset volume. These factors are the motivation to search for faster predictive approaches to get insight in the electronic properties of molecules.

Machine learning (ML) provides a strategy to overcome these computational limitations.[3] After training, ML models can predict electronic properties with negligible computational cost compared to traditional quantum chemistry. Recent work has demonstrated that methods such as gradient-boosted decision trees, kernel regressors, and graph neural networks (GNNs) achieve mean absolute errors well below 0.2 eV in HOMO-LUMO gap prediction tasks ADD REFS. Crucially, large-scale quantum-chemical datasets like QM9, which contains orbital energies for over 130,000 small organic molecules, enable the development and benchmarking of ML models with the data volume required for robust learning.

Beyond property prediction, ML also enables generative molecular design. Traditional experimental workflows require iterative synthesis and characterization to identify molecules with desired electronic properties. ML-based generative models, however, make it possible to explore hypothetical chemical structures, optimize properties in silico, and drastically reduce the number of costly experiments. This accelerates molecular discovery pipelines, supports safer design exploration, and opens chemical regions that would be impractical to investigate experimentally.

In this study, we pursue a two-fold objective. First, we build predictive models for HOMO-LUMO gap estimation using both Light Gradient Boosting Model (LightGBM) as a high-performance gradient boosting framework, and a Graph Convolutional Network (GCN) as a graph neural network (GNN) that learns molecular representations directly from graph topology. Second, we integrate the predictive GCN into a conditional variational autoencoder (CVAE) to generate novel molecular structures with user-specified target HOMO-LUMO gaps. This combined framework enables both accurate property prediction and inverse molecular design, supporting accelerated discovery workflows in molecular design.

2 Methodology

2.1 Dataset

For all models in this study, we used the QM9 dataset,[4] a widely adopted quantum chemistry dataset comprising 130,831 small organic molecules with associated properties, including the HOMO–LUMO gap. Only molecules with valid SMILES representations were retained, leaving 129,012 molecules, can be seen in Figure 2.1. SMILES (Simplified Molecular Input Line Entry System) is a text-based notation that encodes chemical structures as ASCII strings, enabling efficient computational processing; for example, benzene is represented as `C1=CC=CC=C1`. While SMILES strings can be interpreted by computers, they are not directly suitable as input for most machine learning models, which require numerical representations. Consequently, preprocessing is necessary to transform molecular structures into informative, fixed-length numerical features suitable for model training.

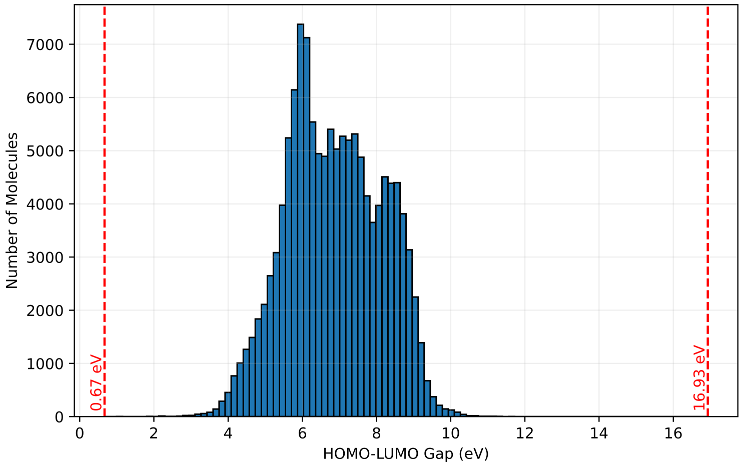


Figure 2.1: Illustration of the bandgap distribution in the QM9 dataset where only the valid molecules are kept. The vertical dashed lines indicates the minimum and maximum bandgap values in the dataset.

2.2 Molecular Representations and Preprocessing

In this study, we focus on two different models for HOMO-LUMO gap prediction: LightGBM, a classical machine learning model and GCN, a deep learning model, which require different types of molecular representations. Therefore, two types of molecular representations were generated from the SMILES strings.

2.2.1 Descriptor-based features for LightGBM

Using RDKit, a set of 217 molecular descriptors was generated for each molecule from the SMILES representation.[5] These descriptors capture various aspects of molecular structure and properties, including molecular size, shape, electronic characteristics, surface area, and the presence of functional groups. Together, they provide a fixed-length numerical representation of each molecule suitable for machine learning models such as LightGBM.

2.2.2 Graph-based representation for GCN:

For graph neural networks, molecules are represented as graphs where atoms are nodes and bonds are edges. As the QM9 dataset represents molecules as SMILES, all molecule SMILES were translated to a node and edge matrix using the *Chem* package of *RDKit*[6]. Node features include atomic number, formal charge, hybridization, aromaticity, and whether the atom is in a ring structure. The edge matrix was defined as an adjacency matrix where the diagonal elements were set to 1, indicating a self-connection, which makes the matrix amenable to convolutions[7]. The distance of each bond was included in the edge matrix as 1 over the bond distance. The representation of a molecule as a node and edge matrix allows the GCN to learn structural features directly from molecular graphs without relying on precomputed descriptors.

2.3 Data Splitting

To ensure unbiased evaluation, the dataset was split into training, validation, and test sets. For all models, nested cross-validation was used:

- **Outer loop:** 5-fold cross-validation to estimate generalization performance
- **Inner loop:** 10-fold cross-validation for hyperparameter tuning

The splits were stratified with 10 bins to preserve the distribution of HOMO-LUMO gaps across the different folds for better results.

2.4 LightGBM

LightGBM is a gradient-boosted decision tree algorithm optimized for speed and high-dimensional data. It grows trees in a leaf-wise rather than level-wise manner, allowing it to capture complex, non-linear relationships more efficiently than many traditional boosting methods. This makes it well-suited for tabular molecular descriptor data, as it is robust to feature scaling, multicollinearity, and missing values.

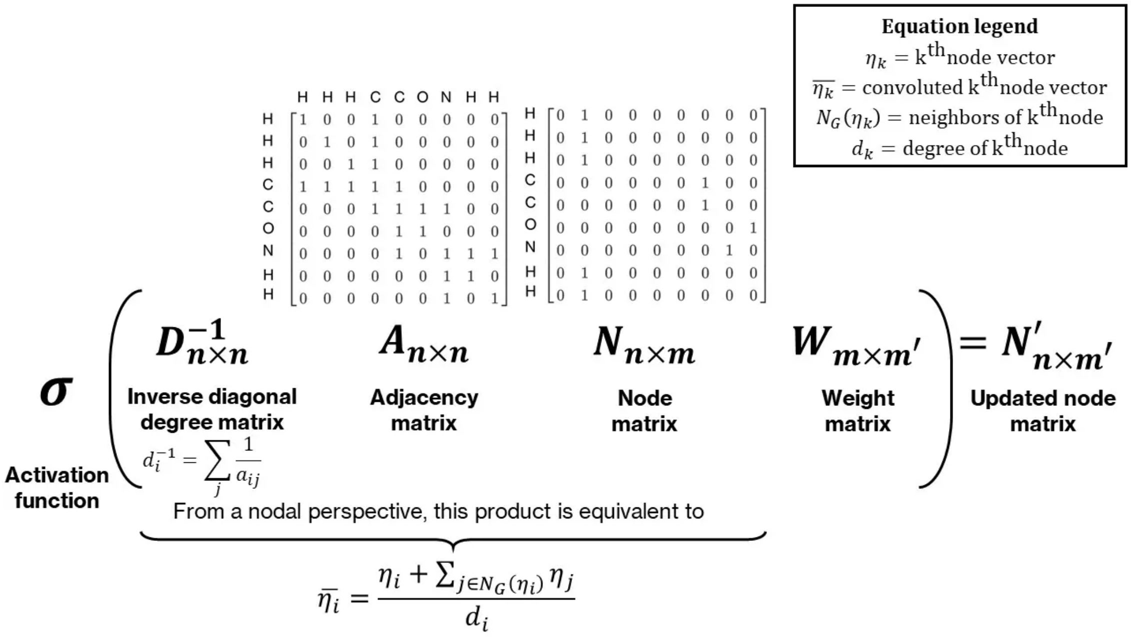
Hyperparameter optimization for the LightGBM model was carried out using Optuna. The search space included parameters controlling model complexity and regularization: the number of leaves (num_leaves, 16–256), the learning rate (learning_rate, 1×10^{-3}–0.1, log-scaled), feature subsampling (feature_fraction, 0.7–1.0), sample bagging (bagging_fraction, 0.7–1.0; bagging_freq, 1–5), and the minimum number of samples per leaf (min_data_in_leaf, 10–100). Other LightGBM settings, such as the boosting type (gbdt) and objective function (regression_l1), were kept fixed throughout. Optuna selected the optimal hyperparameter configuration by minimizing the mean absolute error (MAE) obtained from the inner loop of the nested cross-validation, using a total of 50 trials.

After nested cross-validation, the best-performing hyperparameters were selected (either by averaging over outer folds or by majority vote). The final LightGBM model was then trained on the entire dataset to produce predictions for comparison with the GCN model.

2.5 GCN

The implementation of the GCN model was based on a tutorial about creating a simple Pytorch-based GCN[7]. Some edits to the tutorial code were made, for example the addition of node features, the addition of an R^2 metric, and correcting a mistake with the standardization of the loss.

Starting from the node and edge matrix representations of a graph, the GCN model first applied a graph convolution using multiple convolution layers. Each convolution layer gives a node information about its neighbors using the matrix multiplication shown in Figure 2.2. After the convolution layers *pooling* is applied, which turns the 2-dimensional matrix into a 1-dimensional vector that contains the means of every column of the node matrix. This vector can then be passed to the neural network, which is a simple multilayer perceptron (MLP), a fully connected feedforward neural network. For more details on the implementation, see the GitHub repository add ref to repo.


Equation legend
η_k = k$^{\text{th}}$node vector
$\overline{\eta_k}$ = convoluted k$^{\text{th}}$node vector
$N_G(\eta_k)$ = neighbors of k$^{\text{th}}$node
d_k = degree of k$^{\text{th}}$node
σ
Activation function
$D^{-1}_{n\times n}$
Inverse diagonal degree matrix
$d_i^{-1} = \sum_j \frac{1}{a_{ij}}$
$A_{n\times n}$
Adjacency matrix
$N_{n\times m}$
Node matrix
$W_{m\times m'}$
Weight matrix
$= N'_{n\times m'}$
Updated node matrix
From a nodal perspective, this product is equivalent to
$\overline{\eta_i} = \frac{\eta_i + \sum_{j\in N_G(\eta_i)} \eta_j}{d_i}$

$$\begin{array}{c|ccccccccc}
 & H & H & H & C & C & O & N & H & H \\
\hline
H & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
H & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
H & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\
C & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\
C & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\
O & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\
N & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 \\
H & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\
H & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1
\end{array}
\quad
\begin{array}{c|ccccccccc}
H & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
H & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
H & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
C & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
C & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
O & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
N & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
H & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
H & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{array}$$

Figure 2.2: Graph convolution for an acetamide molecule[7].

Hyperparameter optimization for the GCN model was performed using a simple grid search, as this allowed easy parallelization of hyperparameter tuning. The parameters and their possible values included in the grid search are: batch_size (64, 128, 256, 384), hidden_nodes (64, 96, 128), n_conv_layers (1-5), n_hidden_layers (1-3), learning_rate (0.001, 0.003, 0.005, 0.007, 0.01). Other parameters such as the maximum dimensions of the node and edge matrix, and the number of epochs were kept constant. After nested cross-validation, the same best-performing hyperparameters were obtained for each outer fold: batch_size = 256, hidden_nodes = 128, n_conv_layers = 4, n_conv_layers = 2, learning_rate = 0.003, the number of epochs was kept constant at 50.

References

[1] Sworakowski, J. How Accurate Are Energies of HOMO and LUMO Levels in Small-Molecule Organic Semiconductors Determined from Cyclic Voltammetry or Optical Spectroscopy? *Synth. Met.* **2018**, *235*, 125–130.

[2] Costa, J. C.; Taveira, R. J.; Lima, C. F.; Mendes, A.; Santos, L. M. Optical band gaps of organic semiconductor materials. *Optical Materials* **2016**, *58*, 51–60.

[3] Hasan, M. M.; Tarkhaneh, O.; Bungay, S. D.; Poirier, R. A.; Islam, S. M. Predicting HOMO-LUMO Gaps Using Hartree-Fock Calculated Data and Machine Learning Models. *J. Chem. Inf. Model.* **2025**, *65*, 9497–9515, Epub 2025 Sep 10, PMID: 40929702.

[4] Team, P. torch_geometric.datasets.QM9 — PyTorch Geometric Documentation. https://pytorch-geometric.readthedocs.io/en/2.6.1/generated/torch_geometric.datasets.QM9.html, 2024; Accessed: 2025-12-09.

[5] RDKit: Open-Source Cheminformatics, RDKit Documentation and Python Tutorials. https://www.rdkit.org/docs/GettingStartedInPython.html, 2025; Accessed December 6, 2025.

[6] RDKit Developers, rdkit.Chem Module Documentation. 2025; https://www.rdkit.org/docs/source/rdkit.Chem.html, Accessed: 2025-12-15.

[7] Deshmukh, G. Building a Graph Convolutional Network for Molecular Property Prediction. 2023; https://medium.com/data-science/building-a-graph-convolutional-network-for-molecular-property-prediction-978b0ae10ec4, Accessed: 2025-12-15.