Predicting Electrochemical Potential of Quinones Using Machine Learning Methods

Master's Thesis

Project Report
by
Elif Çetin

Aalborg University
Department of Chemistry and Bioscience

This report was written and typeset in LaTeX. Machine learning models were developed in Python, using the Chemprop and RDKit libraries, and analyzed in RStudio. Grammarly was used for proofreading.

AALBORG UNIVERSITY
STUDENT REPORT

Chemistry and Bioscience
Aalborg University
http://www.aau.dk

Title:
Predicting Electrochemical Potential of Quinones Using Machine Learning Methods

Theme:
Master's Thesis

Project Period:
Spring Semester 2025

Project Group:
CE4-4-F25

Participant:
Elif Çetin

Supervisor(s):
Sergey Kucheryavskiy
Jens Muff

Copies: 1

Page Numbers: 66

Date of Completion:
June 1, 2025

Abstract:

With growing energy demands and the transition to renewable sources, the need for grid-scale energy storage systems is increasing. Quinones, a group of redox-active organic compounds that can be derived from fungi and bacteria, are promising biomolecules for use in redox flow batteries due to their tunability. This study investigates the prediction of quinone standard reduction potentials using machine learning, comparing transformer-based large language models (LLMs) and graphical neural networks (GNNs). The best-performing configurations of LLM and GNN models achieved average test set R^2 values of 0.734 and 0.721, respectively. However, LLMs have exhibited poorer performance on validation sets compared to test sets, indicating issues with model fitting. Within the optimal configurations, the top individual LLM and GNN models achieved an R^2 of 0.777 and 0.774 on the test set, respectively. While LLMs demonstrated slightly better accuracy, they require significantly higher training times and computational costs.

Table of Contents

Preface

This report is written by Elif Çetin as part of the long master's thesis in Chemical Engineering. The project period is from September 2024 to June 2025, and will be presented in June 2025. This project is supervised by Associate Professor Sergey Kucheryavskiy and co-supervised by Associate Professor Jens Muff from the Department of Chemistry and Bioscience.

Acknowledgments.
I would like to express my deepest gratitude to my supervisor, Sergey Kucheryavskiy, for his invaluable guidance, insightful feedback, and continuous support throughout the course of this research. His expertise and mentorship have been instrumental in shaping this thesis. I am also sincerely grateful to my co-supervisor, Jens, for providing the essential data that made this study possible. Finally, I would like to thank my partner, Dean, for his unwavering support, patience, and encouragement throughout my studies. His understanding and care made all the difference during this challenging period.

[illegible]

Elif Çetin

Chapter 1

Introduction

As technology advances and the world population rises, the energy demand also increases. Although fossil fuels are still an important part of the energy sector, awareness of their impact on the environment is growing. With these changes, climate laws and agreements are focusing on transitioning away from fossil fuels, towards renewable energy sources [1]. However, renewable energy brings its own set of obstacles. Due to the nature of the renewable energy sources, they require grid-scale electrical energy storage systems [2]. Redox-flow batteries are an ideal candidate for the solution of this energy storage problem [3].

Quinones are organic molecules that have a wide array of uses in energy storage, pharmaceuticals, and biological systems. In recent years, the use of quinones in batteries as active materials has gained more attention as they can be an alternative to the transition metals used in lithium-ion batteries. Organic quinones are renewable, biodegradable, and more sustainable compared to metal-based compounds since they can be obtained from biomass [4]. Quinones produced by bacteria or fungi display promising results, showing that they can replace commonly used quinones derived from petrochemicals [5]. Additionally, quinones are renewable and more affordable, making them a better and economically viable solution for large-scale energy storage compared to metals [6].

Quinone-based systems, unlike lithium-ion batteries, are not reliant on finite resources. Furthermore, quinones have tunable molecular properties, and their performance can be enhanced or customized [3]. In battery applications, quinones undergo reversible redox reactions where they accept and donate electrons. These charge and discharge cycles make quinones suitable compounds for use in redox-flow batteries.

Despite these advantages, choosing efficient quinone candidates is a challenging process. Discovering their standard reduction potential, which is a critical property that determines their suitability for energy storage applications, is traditionally done with experimental work. This is a costly and resource-intensive process that involves laboratory tests and chemical synthesis since many bioquinones only has been identified as secondary metabolites in too low concentrations for experimental purification and testing [7]. ML methods can boost this process by incorporating knowledge acquired by the traditional methods into complex models connecting the structure of the molecules with their properties [8, 9].

ML applications are not limited to molecular property predictions of quinones. For example, deep learning models are used in various areas such as predicting the bioactivity of drugs, the structure of glycans from mass spectrometry data, the properties of catalysts in chemical reactions, and many more [10, 11]. These advancements highlight the future potential of ML in chemistry, enabling faster and more cost-effective discovery processes.

This research focuses on comparing different ML methods to predict the standard reduction potential of quinones using their molecular structures as inputs. The dataset used to train these ML models was generated through density functional theory (DFT) calculations performed in the Gaussian 09 program, rather than derived from experimental data [5]. By using pre-trained models and further refining them with this quinone-specific data, the study aims to demonstrate how ML techniques can optimize the process of efficient quinone discovery. This approach both aligns with the current focus of computational chemistry advancements and addresses important challenges in sustainability and energy storage areas.

The structure of this report is as follows: Chapter 2 explains the theoretical background of quinones, redox-flow batteries, principles of machine learning methods, and different molecular structure notations that are used throughout this work. It additionally provides a review of the existing literature on relevant research on quinones, computational chemistry, and ML methods. Chapter 3 shows the experimental approach, including technical details, chosen architectures, and training procedures. The results and discussion given in Chapter 4 analyze the performances of the different models and their comparisons. Finally, Chapter 5 summarizes the findings, discusses limitations, and gives conclusions.

Chapter 2

Theoretical Background

This chapter provides a detailed framework for the concepts utilized in this study and discusses advancements in their respective fields. It also explains the properties of quinones, how they are obtained, and their significance in energy storage. Different chemical notations that represent the molecular structure of compounds are shown and compared. Lastly, the principles of machine learning (ML) and their importance in computational chemistry are highlighted.

2.1 Quinones

Quinones are a class of organic compounds that are characterized by their fully conjugated cyclic dione structure. They come in various sizes, structures, and with a wide array of functional groups. Quinones are generally named after the main aromatic system from which they were derived. Hence, benzene-derived quinones are classed as benzoquinones, naphthalene-derived quinones are classed as naphthoquinones, anthracene-derived quinones are classed as anthraquinones, etc. [12]. Different structures of these quinone classes can be seen in Figure 2.1. Because of their redox-active structure, they play a fundamental role in chemical and biological processes. This redox-active property makes quinones an integral electron carrier in energy storage systems, pharmaceuticals, and biological systems [13, 14, 15]. They are studied in a wide range of fields, from their potential in high-performance flow batteries to their neuroprotective effects that can prevent Alzheimer's disease [16].

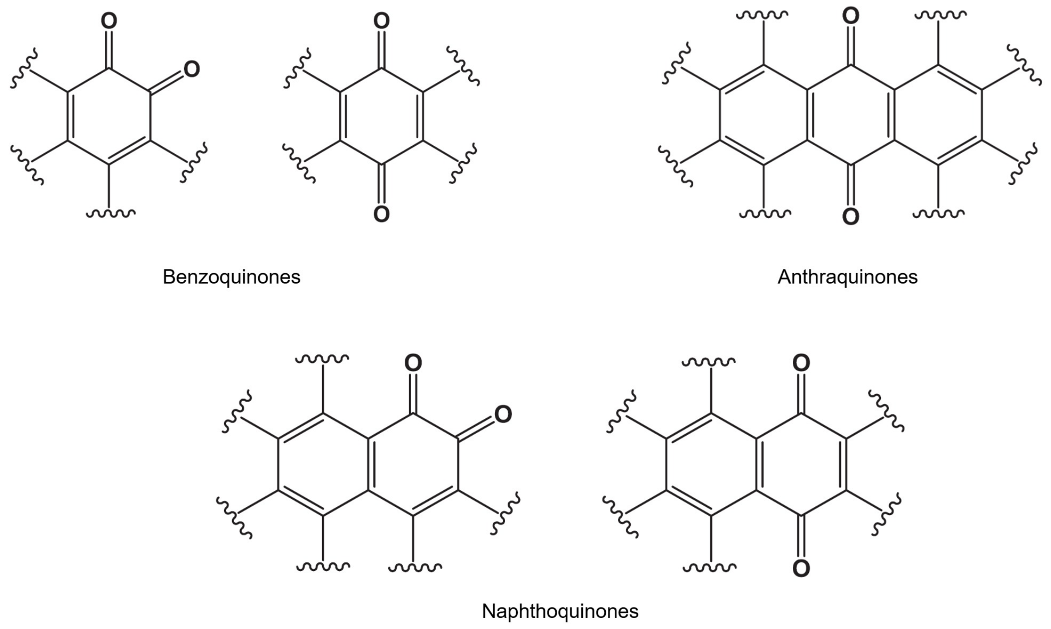


Figure 2.1 Most common quinone classes [17].

Quinones can be found commonly in nature and exist within various organisms, including algae, fungi, plants, animals, and bacteria. Ubiquinones (coenzyme Q) are a key component of the electron transport chain in mitochondria and facilitate Adenosine triphosphate (ATP) production [14]. In photosynthetic organisms, plastoquinones play a critical role in electron transport within photosystem II [15]. Bacteria utilize menaquinones as electron carriers in metabolic pathways for anaerobic respiration [18]. The structural diversity of naturally occurring quinones comes from enzymatic modifications of their aromatic precursors and provides a wide array of functions and adaptability to different environmental conditions [19].

Quinones can be synthesized in various ways, including the oxidation of aromatic hydrocarbons or phenols. Chemical oxidation using agents like chromates or enzymatic synthesis under controlled conditions are also common methods [19]. The tunability of quinones enables their properties to be optimized for specific applications. Solubility, redox potential, and stability can be adjusted through modifications. For example, adding electronegative groups can increase their reduction potential, improving their energy density for battery applications [13]. Possible substitutions and configurations create many quinone derivatives, providing a wide selection to be researched [20]. Historically, quinone discovery and synthesis have been done manually in laboratories. These processes include isolation from natural resources or chemical synthesis and analytical characterization steps. However, these traditional methods are time and resource-intensive. Studies documenting the extraction and identification of quinones from various biological sources underline the meticulous efforts required [7].

2.2 Redox-flow batteries

Redox-flow batteries (RFBs) are a type of rechargeable battery where the energy is stored in liquid electrolytes that flow through an electrochemical cell [21]. Two separate tanks hold these electrolytes. Anolyte is the negative electrolyte tank that keeps the anodic redox-active material solution on the oxidation side. Catholyte is the positive electrolyte tank that keeps the cathodic redox-active material solution on the reduction side. These solutions flow from tanks to separate compartments of a cell stack. In the cell stack, RFBs contain bipolar plates that ensure electrolytes do not contact current collectors and a membrane that allows ions but not whole molecules to pass. Redox reactions occur at the electrode, which enables the battery to charge and discharge. These redox reactions are reversible. Hence, the battery can be recharged multiple times.

The separation between the reaction and the storage area makes power and capacity independent from each other. For RFBs, the area of the electrode in the cell stack determines the power, the difference between the reduction potentials of the redox couples determines the cell voltage, while the volume of storage tanks and the concentration of the contained electrolytes determine the capacity [22]. Since the concentration of the solution is an important factor, highly soluble redox-active compounds are preferred in RFBs. Performance of RFBs is noted based on their Coulombic efficiency, voltage efficiency, and energy efficiency. The ratio of charge and discharge capacities is the Coulombic efficiency, the ratio of charge and discharge voltages is the voltage efficiency, and finally, the product of the Coulombic efficiency and voltage efficiency gives the energy efficiency[21].

In commercial applications, vanadium species are the most commonly used electrolytes in RFBs. Vanadium RFBs (VRFBs) can achieve 80% energy efficiency and have theoretically infinite lifetime [23]. However, VRFBs face challenges: the limited vanadium resources causing VRFB costs to skyrocket, high corrosion rate, and toxicity [24]. Other metal-based RFBs also suffer from similar challenges, such as limited resources, complex preparation processes, solubility issues, harmful extraction and mining practices, and lower efficiency [25]. Recent studies have been investigating organic compounds and their redox activities to find an alternative to metal-based RFBs [22].

2.2.1 Quinones in redox-flow batteries

Quinones gather attention for their applications in RFBs [26]. Their redox properties, tunability, and abundance make them suitable for sustainable energy storage solutions [13, 27]. Quinones provide efficient energy storage and release by

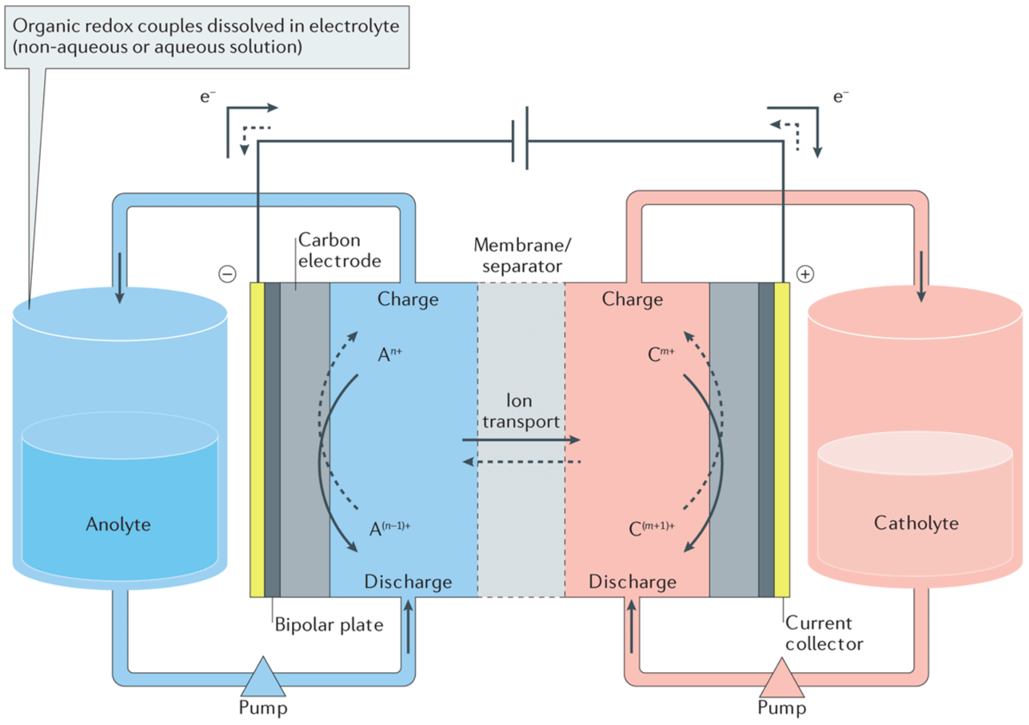


Figure 2.2 Illustration of a redox-flow battery [21].

undergoing reversible redox reactions. An example redox equilibria between benzoquinone and hydroquinone can be seen in Figure 2.3. This reaction provides high energy density and fast charge-discharge cycles in RFBs [28]. Additionally, quinone-based systems have a lower environmental impact compared to traditional metal-based batteries as they are biodegradable and can be obtained from renewable sources such as bacteria and fungi [29, 30].

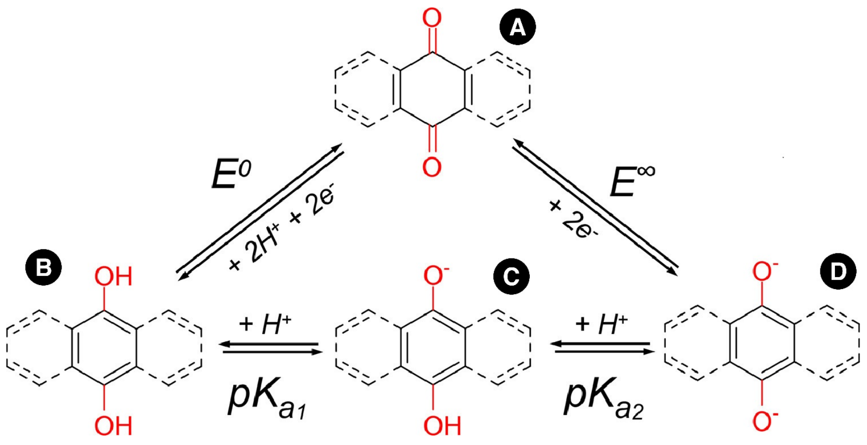


Figure 2.3 Redox and acid-base equilibria between benzoquinone (A) and hydroquinone in acid (B), monodeprotonated (C) or dideprotonated (D) forms [31].

Anthraquinones, a group of quinones present in plants and lichens, were studied for their redox active properties in the early 1900s [32]. In 1983, Roy and Aditya experimented with Anthraquinone-2-sulfonate for its potential use in battery systems [33]. It took some time for anthraquinones to be studied in RFBs in particular. In 2012 Wang et al. experimented with anthraquinone derivatives in non-aqueous metal-organic RFBs [34]. In 2014, Huskinson et al. developed a metal-free, organic–inorganic aqueous RFB with 9,10-anthraquinone-2,7-disulfonic acid (AQDS) and bromine as redox couples in sulphuric acid [27]. This metal-free battery demonstrated a power density of 600 mW cm^{-2} at a current density of 1,300 mA cm^{-2}. Additionally, after ten cycles, it had a 99.2% capacity retention at 500 mA cm^{-2} current density. Later on, to replace the toxic bromine in the RFB, Lin et al. deployed nontoxic, nonvolatile, and noncorrosive food additive ferricyanide as the positive and hydroxylated anthraquinones as the negative redox couple in potassium hydroxide solution [4]. At room temperature, this battery exhibited 450 mW cm^{-2} power density, and at 45 °C, 700 mW cm^{-2}. Through 100 cycles, this RFB had 0.1% capacity loss per cycle with a current efficiency exceeding 99%. Currently, anthraquinone derivatives are the main electrolyte material used in aqueous organic RFBs. They can resolve several issues that aqueous metal ions exhibit with their low cost, corrosivity, and toxicity. In addition, anthraquinones have high chemical stability and a large molecular size, which decreases crossover issues.

Apart from anthraquinones, other quinone types have been worked on as potential electrolytes in aqueous organic RFBs. For example, phoenicin with ferricyanide redox couple exhibited an initial capacity of 235.1 mAh (11.75 AhL^{-1}) and had 0.35% capacity loss per cycle [35]. In another study, Yan et al. developed a polymer-based aqueous organic RFB with polyhydroquinone as cathode, polyimide as anode, and polymer particulate slurry electrolyte [36]. This battery showed a capacity of 10 AhL^{-1}) with 0.1% capacity loss per cycle over 300 cycles operated at 20 mA cm^{-2} current density. While these values are worse than anthraquinone-based RFB, it is still a comparable performance and shows that other varieties of quinones also have the potential to be a good candidate as an RFB electrolyte.

Other organic compounds have been researched for battery applications apart from quinones. For instance, nitroxide radicals have been studied due to their redox activity and their rather stable radical forms. However, they have exhibited cycling stability issues and lower energy densities [37]. Conducting polymers like polythiophenes and polypyrroles were also considered but faced challenges related to solubility and processability [20]. Quinones became the better candidates due to their tunability with chemical modifications, scalability, lower cost, and redox properties [13]. Since electrolyte concentration and the potential difference between redox couples are critical to RFB performance, the solubility and reduction potential of redox-active compounds are key factors.

2.3 Chemical notations

Molecular properties depend on many things, including the combination of atoms and bonds as well as the way they are combined. Such as single or double bonds, cyclic or aromatic structures, chirality, hybridization, and conjugated bonds. Therefore, in order to be able to find the link between the structure and the properties, one has to find a way to represent the structure in a precise, informative, and, if possible, unambiguous way, which can then be used for machine learning modelling.

Chemical notations are standardized methods for representing such molecular structures. They are used as a tool for clear communication and data sharing among chemists and computational models. Different types of notations have many uses, such as simplifying the representation of complex structures and differentiating similar molecules [38, 39].

SMILES (Simplified Molecular Input Line Entry System) encodes molecular structures as linear strings using ASCII characters. Atoms are represented by their atomic symbols (e.g., C for carbon, O for oxygen), and bonds are denoted by symbols like - (single), = (double), and (triple). For instance, the SMILES string CCC represents propane, indicating a chain of three carbon atoms connected by single bonds. Ring structures are denoted by numbers; for example, c1ccccc1 represents benzene, where c denotes aromatic carbon atoms, and the matching numbers 1 indicate the start and end of the ring. While SMILES is compact and widely used, it can produce multiple valid representations for the same molecule, leading to redundancy [38].

Canonical SMILES addresses the redundancy in standard SMILES by providing a unique representation for each molecule. This is achieved through a canonicalization algorithm that consistently orders atoms and bonds, ensuring that each distinct molecule corresponds to a single, unique SMILES string. This standardization is crucial for database indexing and comparison tasks [40].

DeepSMILES is a variant of SMILES designed to be more compatible with machine learning models. It modifies the syntax to reduce the complexity associated with ring closures and branching. For example, instead of using paired numbers to denote ring openings and closures, DeepSMILES uses a single symbol to indicate ring closures, simplifying the parsing process. This streamlined syntax helps in reducing errors during model training and generation tasks [41].

InChI (International Chemical Identifier) provides a hierarchical, layered representation of chemical structures, capturing detailed information about atoms, bonds, stereochemistry, and isotopes. Each InChI string begins with a version identifier and is followed by layers separated by slashes, each conveying specific structural information. This notation ensures a unique and unambiguous representation of chemical substances, making it suitable for database searches and interoperability between software systems [42]. However, InChI strings are often longer and more complex than SMILES, which can be a drawback for certain applications.

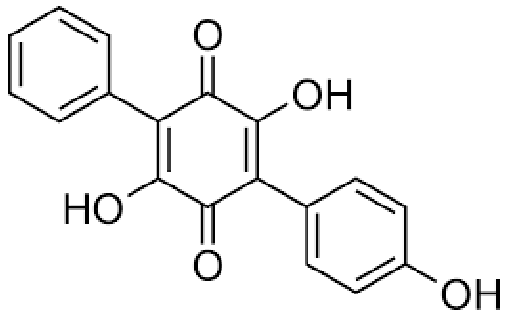


Figure 2.4 Ascocorynin structure

Table 2.1 Different notation representations of Ascocorynin

Formula	C18 H12 O5
SMILES	Oc1ccc(cc1)C1=C(O)C(=O)C(=C(C1=O)O)c1ccccc1
Deep SMILES	Occcccc6))C=CO)C=O)C=CC6=O))O))cccccc6
Canonical SMILES	C1(=C(C(=O)C(=C(C1=O)O)c1ccccc1)O)c1ccc(cc1)O
InChI	InChI=1S/C18H12O5/c19C18H12O5/c19-12-8-6-11(7-9-12)14-17(22)15(20)13(16(21)18(14)23)10-4-2-1-3-5-10/h1-9,19-20,23H

To illustrate these notations, consider the molecule Ascocorynin. Ascocorynin is a member of the benzoquinone family that is obtained from the ascocoryne sarcoides fungus. Figure 2.4 shows the chemical structure of ascocorynin, which consists of a quinone core with one phenyl- and one hydroxyphenyl-side chain attached. Notation representations of ascocorynin, and its formula are given in Table 2.1. The difference between notations can be seen in the table. The choice of notation depends on the specific requirements of the ML model and application, such as compatibility, simplicity, or precision [39].

2.4 Chemical graph representations

Beyond textual notations, molecules can be represented as graphs, which is a natural and intuitive approach in cheminformatics. In a molecular nodes (vertices) represent atoms in the molecule, while edges represent chemical bonds between atoms.

Each node is labeled with the atomic symbol, and each edge is labeled with the bond type (single, double, triple, etc.). An example graph representation of ascocorynin can be seen in Figure 2.5.

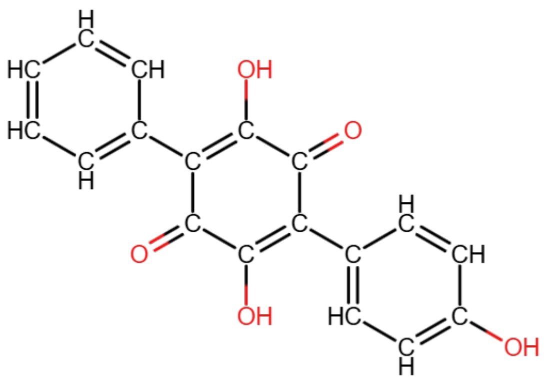


Figure 2.5 Ascocorynin graph representation

Graph-based representations are particularly advantageous for machine learning applications, especially with models like GNNs, which can directly operate on graph-structured data. They allow for the incorporation of rich structural information, such as atom types, bond types, and molecular topology, facilitating more accurate predictions of molecular properties.

In summary, while textual notations like SMILES and InChI offer compact and standardized ways to represent molecules, graph-based representations provide a more detailed and structurally informative approach, especially beneficial for advanced computational modeling and machine learning tasks.

2.5 Machine learning

In order to connect the molecular structures, represented as a graph, text, or any other way, and their properties, one has to create a mathematical model that will describe this connection. This can be done in two ways. Hard models, widely utilized in computational chemistry, are based on current understanding of molecular dynamics, such as density functional theory and electronic structure theory. Such models let researchers predict the properties of a single molecule simply by its structure, without experimental data. The downside of these methods is that they are computationally expensive and, therefore, not suited for, e.g., screening, when thousands of candidates must be evaluated. Soft models, also known as black-box models, in contrast, are data-driven, meaning they are based on empirical data, like molecules whose properties are already known, from experiments, or by using traditional methods. These methods are part of Machine Learning, as they learn from data.

Machine learning (ML) is a scientific discipline that enables systems to learn from data, recognize patterns, and make predictions and decisions with minimal human interference. For scientific studies in areas like chemistry and material science, ML efficiently analyzes complex datasets and predicts outcomes to speed up the discovery processes [43, 44, 45].

ML approaches can be categorized into parametric models, non-parametric models, and neural networks. In parametric models, a fixed set of parameters defines the model [46]. For example, linear regression models have a fixed weight vector and bias, and the number of parameters does not grow with more data. Other examples of parametric methods are: logistic regression and polynomial regression. On the other hand, the complexity of non-parametric models can grow with the number of data points. For instance, the depth of a full-depth decision tree scales as a function of the training set. Similarly, for k-nearest neighbours, the training data is stored as learned parameters, so the number of parameters grows linearly with the training set size [47]. Non-parametric methods include: k-nearest neighbours, decision trees, random forest, and radial basis function kernel support vector machines.

Artificial neural networks (ANNs) are another group of ML methods, which are employed in this study. Typical ANNs are considered parametric models since their structure is predetermined and has a set number of parameters, such as the number of layers and nodes in each layer. Hence, when ANNs are trained, their complexity remains unchanged, independent of the size of the training data. However, if the parameters are not fixed, ANNs can be considered non-parametric. An example non-parametric ANN has been studied by Philipp and Carbonell, aiming to eliminate the difficulty of determining the optimal model structure [48]. They have employed a novel optimization algorithm, "Adaptive Radial-Angular Gradient Descent", where they continuously added new units while eliminating redundant ones during training.

2.5.1 Neural network architectures

Multilayer perceptrons (MLPs) are the earliest and simplest architecture of NNs. They are developed to mimic the structure of the human brain with neurons in separate layers and dense connections. In feedforward networks, each neuron in one layer is fully connected to all the neurons of the neighbouring layers [49]. Their connected structure can be observed in the Figure 2.6.

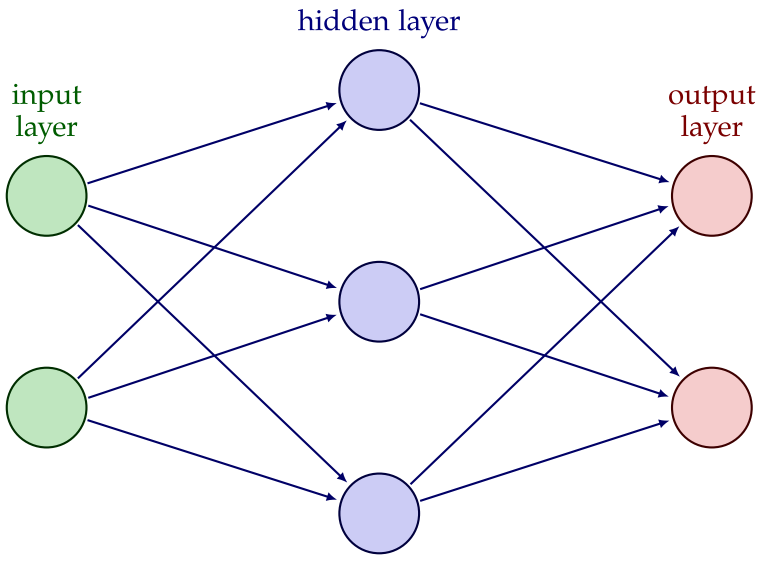


Figure 2.6 Example MLP structure

MLPs learn complex functions via nonlinear activations. Each artificial neuron in an MLP performs a weighted sum of its inputs, akin to multiple linear regression [50]. Mathematically, for a neuron receiving an input vector $\mathbf{x} = [x_1, x_2, \ldots, x_n]$, the output z before activation is computed as:

$$z = \sum_{i=1}^{n} w_i x_i + b = w^\top x + b, \tag{2.1}$$

where w_i represents the weight associated with input x_i, b is a learnable bias term, and w is the weight vector.

To introduce non-linearity and allow the model to learn complex functions, an activation function $f(z)$ is applied to the output of each neuron. Without activation functions, MLPs would be limited to representing only linear transformations, regardless of their depth [51]. Commonly used activation functions include:

- *Sigmoid*: $f(z) = \frac{1}{1+e^{-z}}$, maps input to the range $(0,1)$, historically used in early networks but prone to vanishing gradients.
- *Hyperbolic tangent (tanh)*: $f(z) = \tanh(z)$, maps input to $(-1,1)$, centered at zero, and provides better gradient flow than sigmoid.
- *Rectified Linear Unit (ReLU)*: $f(z) = \max(0,z)$, widely used due to computational efficiency and reduced likelihood of vanishing gradients [52].

Despite their simplicity, MLPs form the building blocks for more advanced neural network architectures and remain useful in various applications, including regression, classification, and time-series prediction [53].

MLPs, due to their structure, ignore spatial or sequential structures. Convolutional neural networks (CNNs) were developed to process spatial data for image classification and recognition. CNNs are feedforward networks with convolutional layers, subsampling layers (pooling layer), and fully connected (dense) layers [54]. A typical CNN architecture example is given in Figure 2.7.

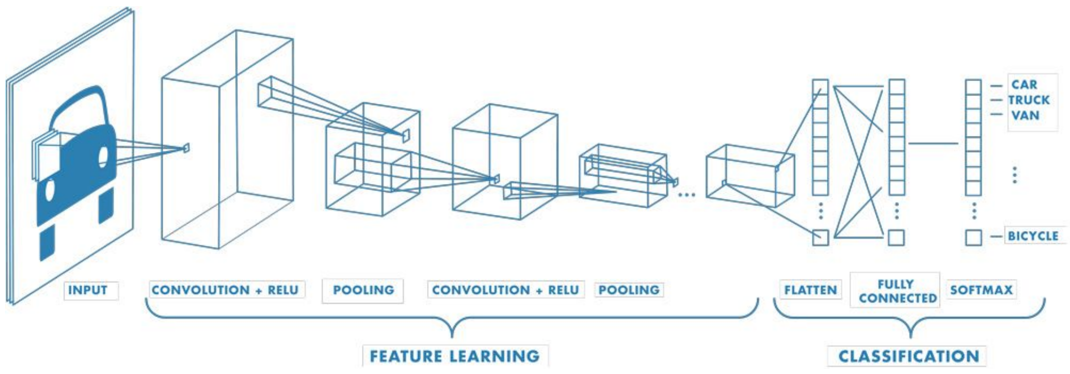


Figure 2.7 Architecture of a CNN with multiple convolutional layers that classifies images [55].

In the convolutional layers, a small learnable matrix known as a kernel or filter slides over the input data to compute local features. The kernel size (e.g., 3×3, 5×5) defines the spatial extent of the filter, and the stride determines how far the filter moves at each step. Each kernel extracts different kinds of patterns, such as edges, textures, or shapes [56]. The result of this convolution operation is a feature map that encodes the presence and location of learned features across the input space. This mechanism enables weight sharing, meaning the same set of weights is reused across different parts of the input, dramatically reducing the number of trainable parameters compared to MLPs. Following each convolutional operation, a non-linear activation function such as ReLU (Rectified Linear Unit) is applied to introduce non-linearity and allow the network to learn complex patterns [57].

In the pooling layers, often max-pooling or average-pooling, the feature maps get downsampled by summarizing the presence of features in subregions. This operation reduces spatial dimensions, speeds up training, and adds spatial invariance by making the network less sensitive to the exact position of features [58]. The final layers are typically fully connected (dense) layers, which use learned features to make predictions. CNNs are powerful for image and structured data since they can utilize shared weights and translation invariance. With the local filtering, CNNs can capture details in images since pixels close to each other are more likely to be related. VGG, ResNet, and EfficientNet are examples of CNNs with different design strategies.

Recurrent neural networks (RNNs) are sequential learning models. They process data one step at a time and have cyclic (recurrent) connections that can hold a hidden state. This hidden state acts as a memory that carries information forward to understand data like text or time series [59]. When training RNNs via backpropagation through sequences, the gradients must travel backward through many time steps. In this process, gradients can become very small (vanish) or very large (explode). If they vanish, RNNs can't retain long-term dependencies, like remembering the beginning of a paragraph. As a solution to this vanishing/exploding gradient problem, Long Short-Term Memory (LSTM) models were introduced. LSTMs contain gates that can manage long-term memory [60]. In an LSTM forget gate decides what to discard, the input gate decides what new info to store, and the output gate controls what to send to the next step. With this architecture, LSTMs resolve the gradient vanishing/exploding issue of the classical RNNs. Since RNNs can process sequential data, they have been the traditional method for language processing. However, due to their inability to parallelize, they are slow to train, and even with an LSTM solution, they struggle with long-range dependency retention.

In 2017, Vaswani et al. reported a new architecture, the Transformer, in their paper All You Need Is Attention. Transformer models are based on multi-head self-attention mechanisms and remove the use of convolution and recurrence [61]. Unlike recurrent neural networks (RNNs), which process inputs sequentially, Transformers convert input data into tokens and allow for parallel processing of input sequences, improving computational efficiency and enabling the modeling of long-range dependencies. The main improvement is the multi-head self-attention mechanism, which enables each token to compute attention scores with every other token in the input. With the help of positional encodings added to the input embeddings, they can retain order and process sequences without needing recurrence. These can be fixed (e.g., sinusoidal) or learned embeddings that allow the model to distinguish between different positions in the sequence.

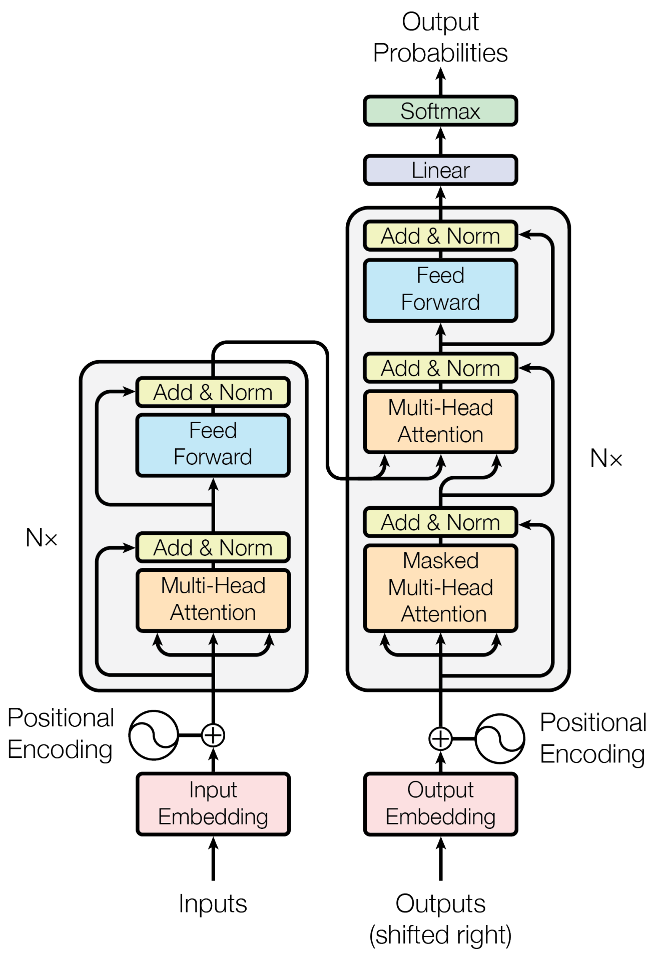


Figure 2.8 Architecture of the Transformer model [61].

The standard architecture includes encoder blocks with multi-head attention and feed-forward layers, layer normalization, residual connections, and dropout for regularization, and decoder blocks for sequence generation tasks (e.g., translation). Transformer architecture is demonstrated in Figure 2.8. Transformers have become the foundation of state-of-the-art models like GPT, BERT, Llama, DALL-E, Stable Diffusion 3, and Sora, which extend the architecture for tasks ranging from language modeling to image generation and chemical property prediction [62]. In cheminformatics, Transformer models can process chemical notations by tokenizing them, enabling them to predict molecular properties, reactivity, and even generate novel compounds. For example, the Psiformer applies self-attention to approximate quantum wavefunctions and has outperformed traditional variational methods in predicting molecular energies [63]. The transformer-based language model is the first type of ML that will be investigated in this paper.

Graph Neural Networks (GNNs) are a class of neural networks designed to operate on graph-structured data, where information is represented as nodes (entities) and edges (relationships). Unlike CNNs or RNNs, GNNs can model non-Euclidean data structures such as social networks, molecules, or citation graphs [64]. The main idea behind GNNs is that each node aggregates information from its neighbours. In each layer, nodes update their representations based on the features of adjacent nodes and the edges connecting them. This allows GNNs to learn both local and global structural patterns in graphs. Message Passing Neural Networks (MPNNs) are a widely used GNN framework where nodes exchange and update messages over multiple iterations. The aggregation between nodes occurs through this message passing. MPNNs are a commonly used architecture for molecular discovery studies, and Chemprop is a useful Python package that provides MPNN models and tools [65]. The architecture of a Chemprop MPNN is given in Figure 2.9.

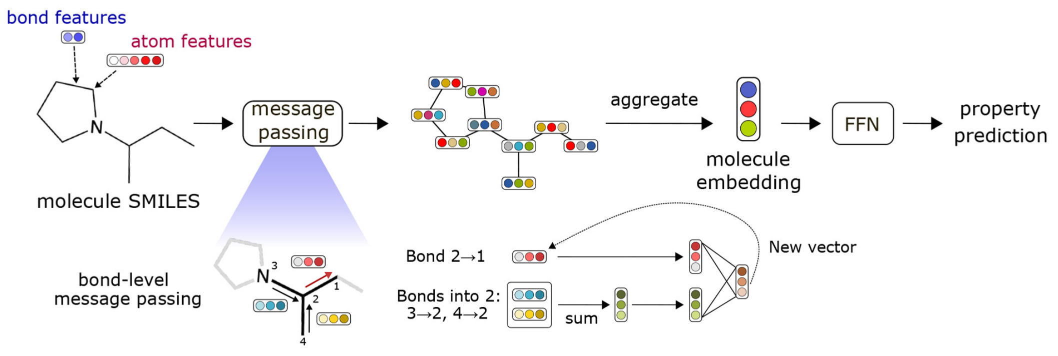


Figure 2.9 Architecture of a Chemprop MPNN [65].

Other variants of GNNs include Graph Convolutional Networks (GCNs), Spectral GNNs (based on graph Fourier transforms), Graph Attention Networks (GATs), and Graph Isomorphism Networks (GINs), each proposing different ways of aggregating neighbour information. GNNs have shown strong performance in tasks like node classification, link prediction, and molecular property prediction [66]. For molecular chemistry, atoms are represented as nodes, and the bonds as edges in GNNs. This structure enables GNNs to capture the spatial relations, distance, and interactions between atoms. They can effectively predict molecular properties and aid discovery processes [66, 67]. Hence, GNNs will be the second investigated ML type in this paper to predict standard reduction potentials of quinones.

2.5.2 Machine learning training

ML models have some fundamental concepts that define the main steps and their properties that are used for training and optimization of the models. It starts with forward propagation, where the data is passed as input, goes through the model, and a prediction is obtained as output. Then, the loss is calculated between the true value and the predicted value. As the final step, backpropagation occurs, where each neuron's weights and biases are updated according to the gradient, starting from the output layer.

To train a model, one typically partitions data into three sets: training, validation, and test. The training set is used to fit the model parameters (weights) via optimization. A separate validation set is used to tune hyperparameters and check for overfitting (e.g., by early stopping). Finally, the test set that was not used during training or validation provides an unbiased estimate of final performance. The training set is fed into the model in batches and repeated in epochs. A batch refers to a subset of the training data processed per model update, while an epoch represents one complete pass of the entire dataset [50].

During both training and evaluation, the model produces predicted outputs $\hat{y}$ for each input, which are compared to the true (measured) values y. In regression, y is a real target and $\hat{y}$ is a prediction. The discrepancy between $\hat{y}$ and y is quantified by a loss function [68]. For regression tasks, the mean squared error (MSE) is a common loss function that was also used in this study and is calculated as follows.

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{2.2}$$

Training a neural network involves minimizing the loss function over the training set by iteratively adjusting the model's weights and biases. The gradients of the loss concerning each parameter are computed using the backpropagation algorithm. These gradients are then used by an optimization algorithm to update the parameters. In this study, the Adam optimizer was employed due to its adaptive learning rate capabilities and computational efficiency.

Adam (Adaptive Moment Estimation) combines the advantages of two popular optimization methods: AdaGrad and RMSProp. It maintains exponentially decaying averages of both the gradients (first moment) and the squared gradients (second moment), which are used to compute individual adaptive learning rates for each parameter [69].

The update rules for Adam are as follows:

$$m_t = \beta_1 m_{t-1} + (1-\beta_1)\nabla L(\theta_t) \tag{2.3}$$

$$v_t = \beta_2 v_{t-1} + (1-\beta_2)\left(\nabla L(\theta_t)\right)^2 \tag{2.4}$$

$$\hat{m}_t = \frac{m_t}{1-\beta_1^t} \tag{2.5}$$

$$\hat{v}_t = \frac{v_t}{1-\beta_2^t} \tag{2.6}$$

$$\theta_{t+1} = \theta_t - \eta \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon} \tag{2.7}$$

Here, θ_t denotes the parameters (weights and biases) at time step t, $\nabla L(\theta_t)$ is the gradient of the loss function concerning those parameters, and η is the learning rate, a hyperparameter that determines the size of the step taken in the direction of the negative gradient, which means how quickly or slowly a model learns and adapts [69]. β_1 and β_2 are decay rates for the moment estimates. ϵ is a small constant (e.g., 10^{-8}) added for numerical stability. $\hat{m}_t$ and $\hat{v}_t$ are bias-corrected estimates of the first and second moments.

This optimizer adaptively adjusts the learning rates of each parameter, which allows it to converge faster and more robustly than traditional stochastic gradient descent, particularly on noisy or sparse gradients. This iterative process gradually reduces the loss on the training data and improves model performance.

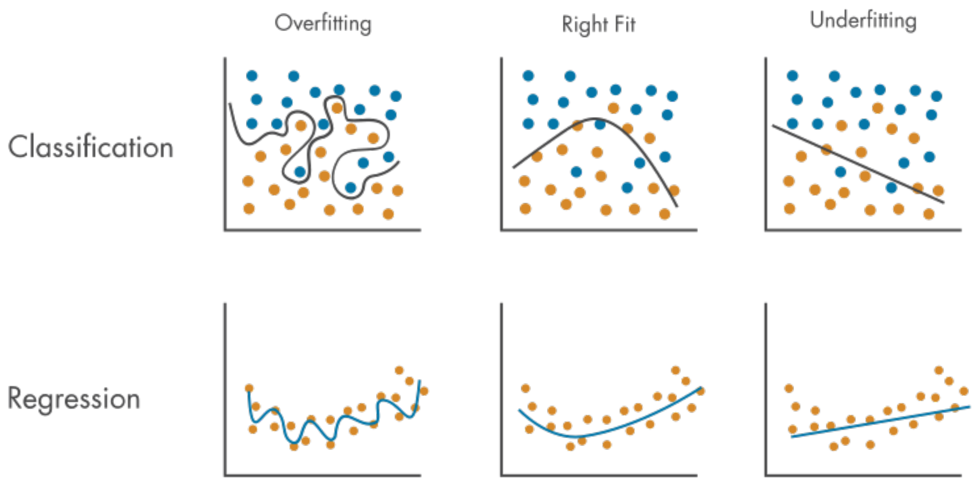


Figure 2.10 Overfitting and underfitting model examples [70].

Tuning and choosing the optimal values for hyperparameters such as learning rate, number of epochs, batch size, and optimizers are essential to obtain a proper fit for a model. Poorly tuned models may suffer from underfitting or overfitting. Examples of different model fittings for classification and regression models are given in Figure 2.10. Overfitting happens when a model learns the training data, including noise and outliers, too well, due to too much training or high complexity of the model, resulting in poor generalization and adaptation to new data. Underfitting happens when a model is too simple or the training is insufficient to capture the complex patterns in the data, resulting in poor performance on both training and new data [71].

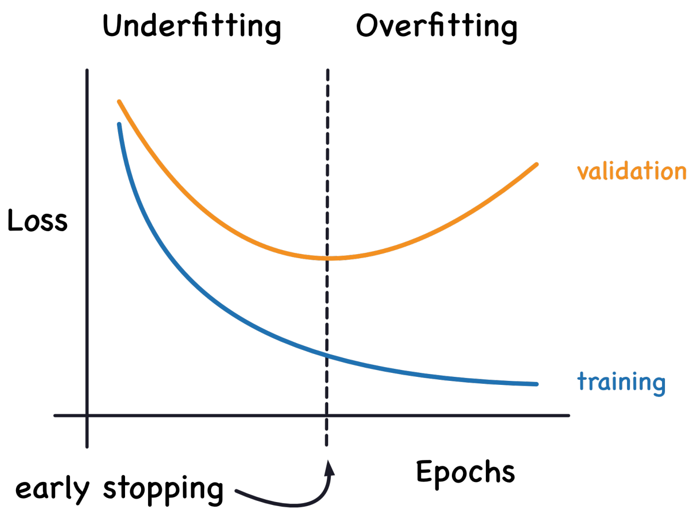


Figure 2.11 Visualization of early stopping on a learning curve [72].

To mitigate overfitting, one commonly used technique is early stopping. During training, the validation loss is monitored after each epoch. A patience parameter defines the number of epochs to wait for improvement in validation loss. If no improvement is observed within this window, training is stopped. Figure 2.11 illustrates this concept: while the training loss continues to decrease, the validation loss begins to increase after a certain point. The optimal stopping point corresponds to the minimum validation loss, which prevents the model from further overfitting. If training continues past this point, the model begins to overfit the training data. Contrarily, if training is stopped too early, before reaching the validation loss minimum, the model may be underfit and fail to learn meaningful patterns.

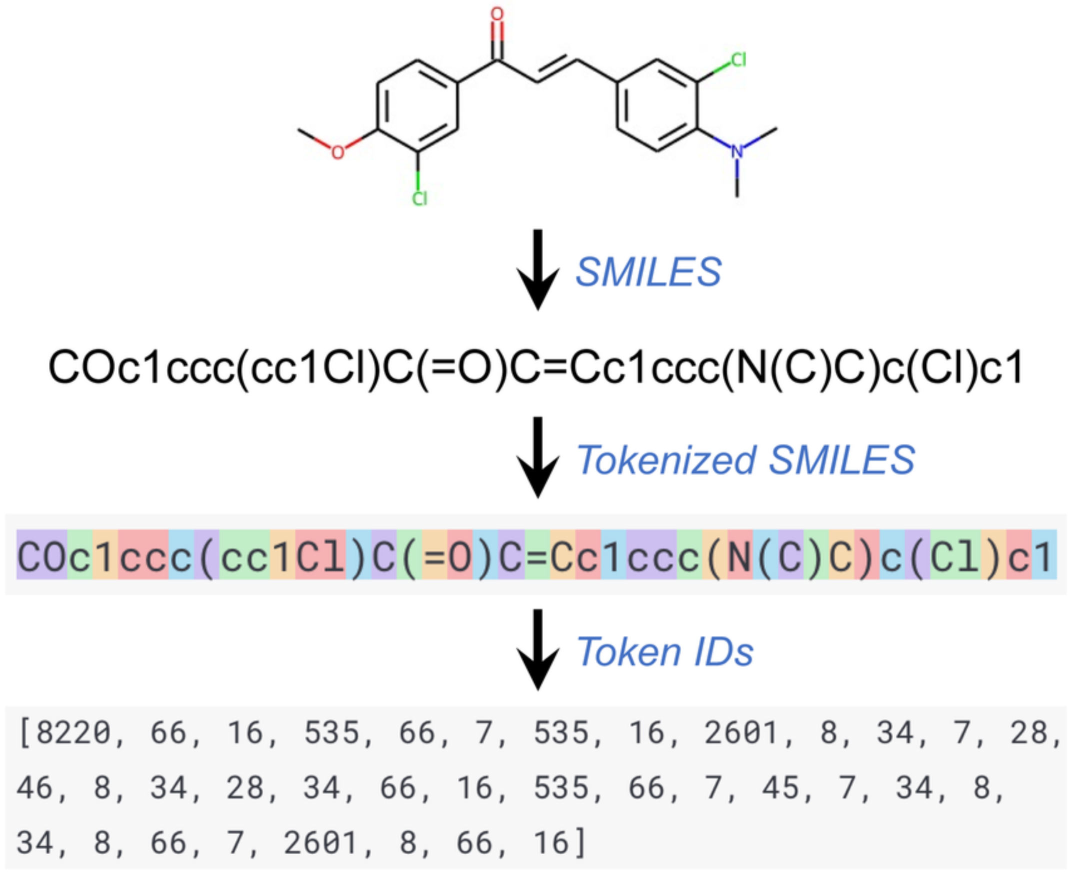
2.5.3 Transfer learning

Deep learning methods require large datasets to train accurate models. However, in many cases like chemical discovery, datasets at hand are limited [9]. Similarly, in our case, the present dataset consists of only 988 quinones with their structures and standard reduction potentials. In these types of cases, a model trained on one task, like generic molecular data, is repurposed for a related, more specific task, like drug discovery. Since more generic datasets can contain large amounts of data, pre-trained models can learn general features [73]. To be applied to the target task, they will then be fine-tuned. Fine-tuning enables the pre-trained model to adapt to the specific task at hand. While fine-tuning, early layers can be frozen so their weights do not update to retain the learned general features. On later steps, they can be unfrozen, allowing them to update. Freezing and unfreezing layers and weights help balance stability vs. adaptability during fine-tuning.

An example of transfer learning can be taking a GPT model that is pre-trained on a large language database, then fine-tuning it to specifically classify scientific papers according to their field. Similarly, the ChemBERTa model that was pre-trained on the SMILES notations of 77M PubChem molecules can be fine-tuned with our rather small dataset of quinones to predict their standard reduction potentials [74].

2.5.4 Feature representation

To utilize molecular data in neural networks, the input must be numerically encoded to be interpretable by ML models. ML algorithms do not inherently understand chemical or textual representations such as SMILES or InChI. Instead, these chemical notations are first tokenized, segmented into discrete, meaningful units, before being transformed into vector embeddings via methods like one-hot encoding, WordPiece, or domain-specific learned embeddings [75]. In transformer-based models tailored for chemical applications, tokenization plays a central role. Chemical strings are treated similar to natural language sentences, where each character or substring can correspond to atoms (e.g., 'C', 'N'), bonds ('=', '#'), or structural markers like ring closures '1', '2' [76]. These tokens capture both local features, like adjacent bonded atoms, and global features as ring systems or functional group patterns, providing a rich and structured input for ML algorithms. Steps of converting an example SMILES string to tokens can be observed in Figure 2.12. Recent advances have highlighted that tokenization schemes significantly affect model performance in chemical property prediction. For example, atom-in-SMILES tokenization improves token specificity and model accuracy by preserving chemically meaningful units [76]. Similarly, subword methods like Byte Pair Encoding and the novel Atom Pair Encoding (APE) have shown enhanced classification performance across biological datasets by retaining semantic cohesion in chemical sequences [77].


O
Cl
O
Cl
N
SMILES
COc1ccc(cc1Cl)C(=O)C=Cc1ccc(N(C)C)c(Cl)c1
Tokenized SMILES
COc1ccc(cc1Cl)C(=O)C=Cc1ccc(N(C)C)c(Cl)c1
Token IDs

```
[8220, 66, 16, 535, 66, 7, 535, 16, 2601, 8, 34, 7, 28,
46, 8, 34, 28, 34, 66, 16, 535, 66, 7, 45, 7, 34, 8,
34, 8, 66, 7, 2601, 8, 66, 16]
```

Figure 2.12 Tokenizing a SMILES string [78].

For GNNs, feature representation can get more detailed and customized. In the molecular prediction field, atoms are represented as nodes, and bonds are represented as edges in graphical data. Graph featurizers use an adjacency matrix for connectivity plus feature vectors for atoms and bonds [66]. Typical atom features can include atomic number, degree, formal charge, chiral tag, number of bonded hydrogens, hybridization type, aromaticity, and mass. Typical bond features can include may include bond type (single/double/triple/aromatic), conjugation (yes/no), ring (yes/no), and stereochemistry. In 2D graphs, molecule geometry is often lost. Advanced specialized models can incorporate spatial features, such as bond angles and inter-atomic distances, if needed. The choice of featurizers crucially affects model performance and should be determined carefully.

2.5.5 Machine learning applications in chemistry

Machine learning (ML) has revolutionized chemical research by providing new methods for molecular chemistry. ML can help speed up processes like predicting molecular properties, designing new compounds, and developing pharmaceuticals [79, 80]. ML methods can quickly identify promising molecule candidates by analyzing big datasets with their pattern recognition and outcome prediction features [81, 82, 83, 84, 45].

ML model applications have even been recognized by receiving the Nobel Prize in Chemistry 2024. One half of it has been awarded to David Baker for his work on computational protein design, and the other half has been awarded to Hassabis and Jumper for developing the AlphaFold2 model. Baker and his coworkers have developed a program that can produce proteins with imaginative generation. This program can design proteins that do not exist in the Protein Data Bank, are unlike any naturally occurring protein, and whose predicted structure matches the experimental one. Winner of the second half of the prize, AlphaFold2, is a convolutional neural network model developed by DeepMind that can predict the 3D structures of proteins from amino acid sequences. It has been an over-50-year effort for scientists to understand the complex structures of proteins. With AlphaFold2, it is now possible to predict structures, which enables researchers to understand antibiotic resistance and advance many other studies [85].

Many other works have used ML methods for energy applications. Shafian et al. combined computational quantum chemistry with reinforcement learning to optimize organic semiconductor materials that are used for solar cell applications. Their research represents the use of machine learning for quantum chemistry and how it can address energy challenges [86]. Amin et al. introduced an ML-based energy Hessian method for creating fast and specialized molecular force fields, which significantly reduced computational costs and time for energy calculations in molecular dynamics [87]. Chen et al. explored atomic mechanisms in lithium-ion diffusion using ML, contributing to advancements in battery technology [88]. Gomez et al. studied how ML enhances the analysis of data generated from microfluidic systems, emphasizing applications in monitoring molecular interactions [89].

Similarly, there are plenty of studies that have utilized ML for drug discovery or similar molecular property prediction purposes. Noviandy et al. used voting-based ML frameworks to predict hepatitis C virus inhibitors. This work demonstrates the potential of interpretable ML in drug discovery [90]. Tian et al. utilized transfer learning techniques to improve the predictive accuracy of bioactive small molecules [91]. Ghosh et al. developed an automated ML-based workflow for virtual screening of COX-2 inhibitors, providing a framework for fast and scalable drug screening [92]. Wang et al. utilized ML models for ADMET evaluation for drug permeability prediction, facilitating the development of safer and more effective pharmaceuticals. Their results have also shown that deep learning methods still do not outperform simple ML methods in drug discovery applications [93].

In another study, deep learning was utilized by interpreting chemical language representations to predict the bioactivity of compounds by Özçelik and Grisoni. By applying natural language processing techniques, the model learned chemical notations and predicted molecular functions with high precision [94]. Their work is very similar to the methods that are used in this paper, using language models to predict molecular properties with chemical notations, and they obtained promising results. Urban et al. applied deep learning models to predict glycan structures from tandem mass spectrometry. This method significantly sped up the glycan structure analysis and showed the potential of ML in simplifying complex chemical procedures [8]. Jain et al. utilized ML models and quantum chemical calculations to predict molecular toxicology. Their work shows the role of ML in understanding the molecular basis of toxicity [95].

GNNs are emerging as a powerful tool for predicting molecular properties [96]. They can represent molecules as graphs with atoms as nodes and bonds as edges. This special representation of molecules enables GNNs to capture intricate structural details, such as bond lengths, angles between bonds, chiral tags, number of neighbouring atoms for each atom, conformers, etc., and accurate prediction of properties like solubility, toxicity, and reactivity. They are also highly adaptable for various molecular representations, which increases their potential to advance computational chemistry [97].

Tran et al. used a combination of GCNs with iterative refinement of LSTMs that can learn distance metrics of small molecules [98]. While their method struggled to generalize to unseen molecular scaffolds, it performed significantly better than prior methods for low-data learning for drug discovery. Gilmer et al. utilized MPNNs' ability to featurize local and global features of the entire input graph to predict molecular properties [99]. Their MPNN method demonstrated promising results and can predict quantum properties of organic molecules significantly faster than the alternative, expensive quantum mechanical simulation method (DFT). Jo and his co-workers combined message passing with an attention mechanism for chemical classification [**Jo2017**]. Their work was the first application of using SMILES strings as direct input to MP algorithms for molecular property prediction, and they obtained comparable results to several benchmarks. Okabe et al. developed a Virtual Node Graphical Neural Network (VNGNN) to predict phonon properties in materials. VNGNNs use extra virtual nodes to augment the flexibility of GNNs to account for the variable and arbitrary dimensions of their outputs [11]. The addition of the virtual nodes was deemed necessary due to the unstable nature of phonons in Okabe et al.'s work.

2.5.6 Challenges and future directions

Despite its advantages, ML in molecular chemistry faces challenges. The quality and diversity of training datasets are critical, as poor or biased data can lead to inaccurate predictions. Building a high-quality training dataset requires extensive research, careful pre-processing, and rigorous validation to ensure accuracy and robustness, while also aiming to collect as many data points as possible. Additionally, meticulously adjusting model hyperparameters is an important but time-consuming step. Without finetuning, ML models tend to overfit or underfit to the training data, making future predictions inaccurate. Especially for work with large datasets that require high computational power to train models, this finetuning step can be very time and resource-intensive. However, after finding ideal parameters and obtaining a model, that same model can be used as many times as needed, quickly.

In conclusion, the integration of ML into molecular chemistry is a significant alternative tool for researchers that can accelerate the discovery processes if the correct tools and previous research, and data exist. While specific references for quinones are sparse, different ML types like the transformer and GNNs are promising tools for identifying new quinone derivatives.

2.6 Problem formulation

The development of efficient and sustainable energy storage systems, such as RFBs, relies heavily on the identification of suitable redox-active materials. Quinones have emerged as promising candidates due to their favorable electrochemical properties and structural tunability. A key parameter in evaluating quinones for RFB applications is their standard reduction potential. Traditionally, accurate estimation of redox potentials requires quantum chemical calculations, such as density functional theory (DFT), which are computationally expensive and time-consuming, or resource-intensive laboratory experiments.

Efficient and scalable prediction of standard redox potentials is critical for accelerating the discovery of redox-active materials for RFBs. While machine learning (ML) can accelerate the screening process, it remains unclear which ML architectures and molecular representations yield the best predictive performance for this task. If reliable predictions can be made using ML models, costly computational or experimental methods can be reserved for the most promising candidates, reducing the time and resources spent on less viable options.

The objective of this thesis is to investigate and compare two types of ML methods: transformer-based pre-trained large language models (LLMs) and GNNs, both of which will use chemical notation representations as input. The goal is to evaluate and compare these methods in terms of predictive accuracy and computational efficiency, in the context of accelerating the discovery of quinone-based materials for energy storage applications. For LLMs, two pre-trained models will be explored, and the study will investigate which chemical notation yields the best performance. For GNNs, SMILES strings will first be converted into graphs to determine whether representing molecular structure as graphs, rather than tokens, leads to improved predictive performance, despite both formats taking an equivalent amount of information. As a step further, a 3D GNN will be employed to assess whether incorporating 3D structural features further enhances model accuracy compared to using 2D representations alone. The methods, models, and datasets used in this study will be described in detail in the following chapter.

Based on this objective, the thesis aims to answer the following research questions:

- Which pre-trained transformer-based LLM provides the best performance for predicting the redox potential of quinones?
- Which chemical notation (SMILES, SMILES Canonical, Deep SMILES, InChI) yields the highest predictive accuracy when used as input to LLMs?
- Does converting chemical notations into graph-based representations and using GNNs improve predictive accuracy compared to LLMs, despite both using equivalent input information?
- Does incorporating 3D features, such as bond distances and angles, enhance the performance of GNNs over 2D representations?
- Are the performance differences between the ML models statistically significant, and which model is optimal for the task of predicting the electrochemical potential of quinones?

Chapter 3

Methods

This chapter of the report explains the methods used for this study. It outlines the steps taken during experimentation, describes how the data was gathered, identifies the models utilized, and explains how they were tuned.

3.1 Transformer based models

Two pretrained transformer-based models were used for comparison. These models use chemical notation texts as inputs and transform them into tokens, allowing them to process and predict an output. ChemBERTa-77M-MTR is the first pre-trained model used for this work and will be referred to as the ChemBERTa model. It is a transformer-based model pre-trained on 77 million SMILES strings from the PubChem database by Seyone Chithrananda [74]. It employs a multitask regression (MTR) objective, where the model learns the similarities in the outputs simultaneously [100]. Even though it was pre-trained with SMILES notation inputs, its performance with other notations will be evaluated and compared as well.

The PubChem10M_SMILES_BPE_450k model is the other transformer-based model that was used for this work and will be referred to as the PubChem model. It is pretrained on 10 million SMILES strings from PubChem by Seyone Chithrananda. It utilizes Byte Pair Encoder (BPE) for tokenization, capturing substructures within molecules. BPE is a combination of character and word-level representation that can decompose unfamiliar strings down to subwords it might recognize. This use of BPE can help a model better identify key characteristics of molecules such as atomic composition, bond types, and molecular configurations [74].

Using pre-trained models speeds up the training process. Since these models already come with a general knowledge of chemical compounds, by just adding extra hidden layers and training them with our dataset, they can predict quinone properties effectively. To use these pre-trained models, training and evaluation were performed using Huggingface Transformers and PyTorch packages on Python.

3.1.1 Model Training and Hyperparameter Tuning

Each model was trained using the respective input representations. The data set was split in a stratified order to ensure the same sets for all models. The data was split as 80% training, 10% validation, and 10% test. Mean squared error was used as the loss function, and Adam optimizer with weight decay was used. An exponential learning rate scheduler was also used with different gamma values. To prevent overfitting, early dropout was added. If the loss of the validation set during training did not improve for 20 epochs in a row, the training loop stopped early, and the model with the best validation loss value proceeded to the evaluation step.

A comprehensive hyperparameter tuning process was conducted, comparing various configurations to identify the optimal settings for each model. The hyperparameters considered included: learning rate, weight decay, gamma, batch size, and number of nodes in hidden layers. These hyperparameter values were chosen after numerous trials and errors, and best best-performing candidates were chosen. In Table 3.1, all the different parameter configuration sets are given.

The models were structured to have three hidden layers and a one-node output layer to predict the reduction potential of the quinones. The number of nodes in hidden layers was one of the parameters that was tuned and compared as well. The alternative number of nodes was chosen as 32-16-8 and 64-32-16. All the configurations were tried one by one for all model types with all the different notations and number of node alternatives.

Table 3.1 Configuration numbers and hyperparameter options

Configuration no.	Batch size	Learning rate	Weight decay	Gamma
1	5	1.00E-04	0.05	0.9
2	10	1.00E-04	0.05	0.9
3	5	5.00E-05	0.05	0.9
4	10	5.00E-05	0.05	0.9
5	5	1.00E-05	0.05	0.9
6	10	1.00E-05	0.05	0.9
7	5	1.00E-04	0.1	0.9
8	10	1.00E-04	0.1	0.9
9	5	5.00E-05	0.1	0.9
10	10	5.00E-05	0.1	0.9
11	5	1.00E-05	0.1	0.9
12	10	1.00E-05	0.1	0.9
13	5	1.00E-04	0.05	0.95
14	10	1.00E-04	0.05	0.95
15	5	5.00E-05	0.05	0.95
16	10	5.00E-05	0.05	0.95
17	5	1.00E-05	0.05	0.95
18	10	1.00E-05	0.05	0.95
19	5	1.00E-04	0.1	0.95
20	10	1.00E-04	0.1	0.95
21	5	5.00E-05	0.1	0.95
22	10	5.00E-05	0.1	0.95
23	5	1.00E-05	0.1	0.95
24	10	1.00E-05	0.1	0.95

All 24 model configurations given in Table 3.1 were used as training configurations, and the optimal setup for each model was chosen later on. Due to the nature of machine learning and how weights are randomized within a limit during training, each configuration was used three times to ensure that no outliers under- or overperformed. As a result, for all the different models with different input types, 24 different parameter configurations, and 2 different node layer structure models were trained 3 separate times. Meaning a total of 144 models were trained for each input type, for each model was trained and used to choose the optimal one. R Studio was utilized to graph and assess the performance of these models.

3.2 Graphical neural networks

In this study, GNNs are utilized in two different ways. The first method is using GNNs with SMILES notations. This model derives 2D graph representations from SMILES chemical notations and will be referred to as 2D-GNN. The second method is using GNNs with 3D Coordinates. This model utilizes 3D spatial information as input to construct graphs, where nodes represent atoms positioned in three-dimensional space, and edges represent bonds, allowing the model to learn from the spatial configuration of molecules. The coordinate information will be taken from the Gaussian files of the molecules. This second model will be referred to as 3D-GNN.

Unlike transformer-based models, a suitable GNN model pre-trained on a large chemical dataset was not found. Hence, building the models and training them from scratch was necessary, requiring more epochs and precise fine-tuning steps. After careful consideration and investigating related papers, it was determined to proceed with MPNNs. The Chemprop package, as it was easy, user-friendly, and open source, was chosen to build our MPNNs [36, 65].

3.2.1 Featurizers

As mentioned in the previous chapter, the selection of the featurizers is a crucial step and can determine the performance of a GNN. The default atom and bond featurizer types from Chemprop were used while customizing their feature options, such as which atomic numbers or hybridization types to include. To not include redundant features, all the atoms and bonds in the quinone dataset were analyzed using the RDKit package, and all possible features were recorded. Some quinones in the dataset contain stars (*) in their notations. In chemical notations, a star can represent a chiral center or an asymmetric carbon atom. To be able to retain the stars in the featurizers, some additional features were included, such as 0 atomic number, unspecified chiral tag, and unspecified hybridization type. All the utilized custom featurizers are given in Table 3.2.

Table 3.2 Featurizers used for the GNN models

Featurizer	Options	Correspondence
Atom Featurizers		
Atomic number	0, 6, 7, 8, 16, 17, 35	*, C, N, O, S, Cl, Br
Degree	0, 1, 2, 3, 4	number of neighbors
formal charge	0	Charge of the atom
Chiral tag	0, 1, 2	Unspecified, CW, CCW
Number of H	0, 1, 2, 3, 4	Hydrogens bonded
Hybridization	Unspecified, SP, SP2, SP3	Hybridization type
Aromaticity	1, 0	Aromatic or not
Mass	float32 value (0 - 1)	Atomic Mass/100
Bond Featurizers		
Bond type	Single, Double, Triple, Aromatic	Type of the bond
Conjugated	1, 0	Aromatic or not
In ring	1, 0	In a ring or not
Stereochemistry	0, 1, 2, 3	None, Any, Z, E

2D-GNN model uses solely the given featurizers and options. 3D-GNN also utilizes all of the featurizers given in the table. However, to capture spatial data, they need more feature extractors added to the default bond featurizers. Hence, on top of the ones given in the Table 3.2, an additional block that calculates the bond angle and the bond distance was added to the bond featurizers for the 3D-GNN. To keep all the features between 0 and 1, similar to the atomic mass operation, distance and angle values were divided by 10 before returning them to the model.

3.2.2 Model Training and Hyperparameter Tuning

The data set was split with Chemprop's built-in tool using a structure-based splitting method, Kennard Stone, that uses Euclidean distances to group the data. The data was split as 70% training, 15% validation, and 15% test.

For training, Adam optimizer with Noam learning rate scheduler is used. Noam scheduler increases the initial learning rate linearly to the given maximum learning rate during the first warmup steps. Afterwards, it decreases the learning rate exponentially to the target final learning rate over the remaining epochs. The lightning package is utilized in Chemprop for training and evaluating steps. The lightning package provides useful tools like the learning rate finder and monitor to streamline the process of choosing an optimal learning rate. Mean squared error (MSE) was used as the loss function, and root mean squared error (RMSE) and Coefficient of Determination (R^2) were monitored as evaluation metrics. Models were trained over a maximum of 300 epochs, and the first 10 epochs were used as the warmup epochs. To prevent overfitting, early dropout was added with a 50-epoch patience.

The overall MPNN consists of three components: message passing layer, aggregation, and the predictor. Apart from the learning rate that was determined with the help of the lightning tool kit. The number of hidden layers of the different sections of the GNNs and their dropouts must be tested to determine the optimal GNN structure. After numerous trials and errors, the best two options for each parameter were determined, and 16 possible configurations given in Table 3.3 were experimented with similarly to the transformer-based models' configurations.

Table 3.3 Configuration numbers and hyperparameters for GNNs

Configuration no.	Depth of MP (*d_mp*)	Dropout of MP	Layers of FFN (*n_ffn*)	Dropout of FFN
1	4	0.05	1	0.15
2	3	0.05	1	0.15
3	4	0.1	1	0.15
4	3	0.1	1	0.15
5	4	0.05	2	0.15
6	3	0.05	2	0.15
7	4	0.1	2	0.15
8	3	0.1	2	0.15
9	4	0.05	1	0.2
10	3	0.05	1	0.2
11	4	0.1	1	0.2
12	3	0.1	1	0.2
13	4	0.05	2	0.2
14	3	0.05	2	0.2
15	4	0.1	2	0.2
16	3	0.1	2	0.2

The message passing layer used is BondMessagePassing, which passes messages along directed bonds rather than atoms. This approach allows the model to capture edge-level information more effectively. The message passing component has *d_mp* number of fully connected layers: an input layer, (*d_mp* - 2) hidden layers of size 300, and an output layer. A ReLU activation function and a chosen dropout rate are applied to introduce non-linearity and reduce overfitting, respectively. Dropout works by randomly deactivating a fraction of the neurons during training, which helps prevent the model from relying too heavily on specific features and improves generalization. NormAggregation is used in the aggregation step to combine atom-level representations into a single graph-level embedding. After aggregation, a batch normalization layer is applied to stabilize training by normalizing feature distributions. Finally, the predictor is a regression-type feed-forward network (RegressionFFN) consisting of *n_ffn* hidden layers followed by an output layer, for a total of (*n_ffn + 1*) layers. Each layer has 300 hidden units, with a chosen dropout rate applied between them. This predictor maps the graph embedding to the target molecular property.

3.3 Data Collection

The dataset used in this study comprises 988 quinone molecules and was originally generated through density functional theory (DFT) calculations performed using the Gaussian 09 program, as described by Kristensen et al. [5]. Each molecule entry includes its molecular formula, computed standard reduction potential, solvation free energy (G_{solv}), molecular weight, biological source, and molecular structure. The dataset is accompanied by a directory of output files in the `.gau` format.

Gaussian 09 is a widely used quantum chemistry software package that allows for detailed electronic structure calculations [101]. It offers a comprehensive suite for electronic structure calculations. Gaussian predicts molecular properties, reaction pathways, and spectroscopic characteristics using quantum mechanical principles. It can be used for applications like modeling complex chemical systems, interpreting experimental data, and designing molecules with specific attributes. It is a versatile and reliable program. This makes it indispensable in both academic and industrial research environments.

Following the completion of this study's modeling and training phases, it came to light that certain molecules in the dataset, specifically some fusarubins and phenicin, exhibited inconsistencies between their DFT-calculated redox potentials and experimental electrochemical behavior. Internal correspondence with the co-supervisor indicated that fusarubin compounds did not display meaningful reversible redox peaks experimentally, and that early calculations for phenicin significantly overestimated its redox potential. It was found to be -0.216 V through experimentation instead of 1.666V, which is given in the dataset.

Due to the timing of this information becoming available, these specific molecules were retained in the modeling process. However, this insight highlights the importance of exercising caution when relying solely on theoretical data for machine learning tasks. While DFT methods are generally robust and widely accepted for redox potential prediction [102, 103], they are not without limitations, particularly when applied to structurally complex or electronically unusual molecules. These limitations should be considered when interpreting model performance and assessing generalization to experimental systems.

The statistics of the quinone dataset are given in Table 3.4. The distribution of the reduction potentials can also be seen in the Figure 3.1.

Table 3.4 Statistics of the reduction potential dataset

Statistic	Value
Units	Volt [V]
Number of data	988
Minimmum	-1.382
Maximum	1.666
Mean	0.201
Median	0.251
Standard deviation	0.363

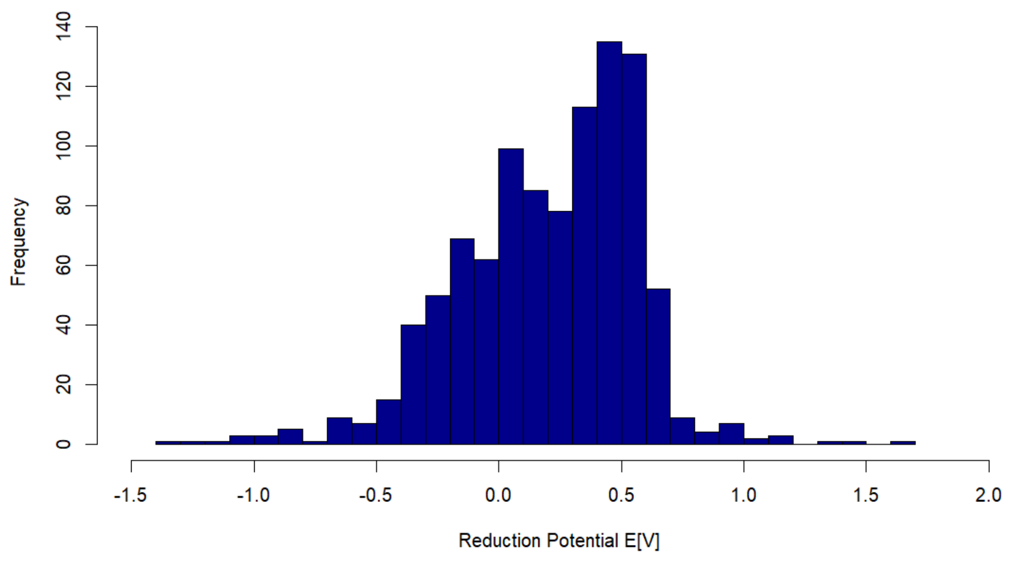


Figure 3.1 Histogram of the reduction potentials of quinones

To be used and compared as the different input types for the transformer-based models, the four chemical notations for each quinone were generated. By using the formula and ID of each molecule and the OpenBabel tool, SMILES, DeepSMILES, Canonical SMILES, and InchI notations of each quinone were obtained. These notations, along with the reduction potentials of the quinones, were used as the training data for the transformer-based models. For 2D-GNN, only SMILES notations were utilized along with reduction potentials of the quinones.

For the 3D-GNN model, Gaussian files were utilized. The Gaussian file of each quinone molecule contains a list where every atom's 3D coordinates are given. However, these files fail to provide the bonds between the atoms. Since quinone molecules are large and have complex structures, it is a challenge to extract the exact structure from the coordinates alone. A custom script was written in Python to draw the structures of molecules from their Gaussian files by drawing bonds between atoms that are closer than a threshold (Figure 3.2a). These molecules, while having inaccurate bond types between atoms, can provide accurate 3D atomic coordinates and an estimated structure. Afterwards, SMILES notations are utilized to draw 2D structures of the quinones (Figure 3.2b). These SMILES molecules can act as a template with accurate bonds between atoms, even though the angles and bond distances are wrong. For heavy atoms, a signature (element, degree, neighbor elements) is computed from both molecules. The code then pairs atoms from the SMILES and Gaussian molecule that share the same signature and creates an atom mapping. Utilizing the atom mapping, the Gaussian molecule is reordered to match the atom indices with the template molecule. To obtain the final 3D structure with accurate atomic positions, bonds, and structure, the Gaussian molecule's bond orders are rewritten using the template molecule (Figure 3.2c).

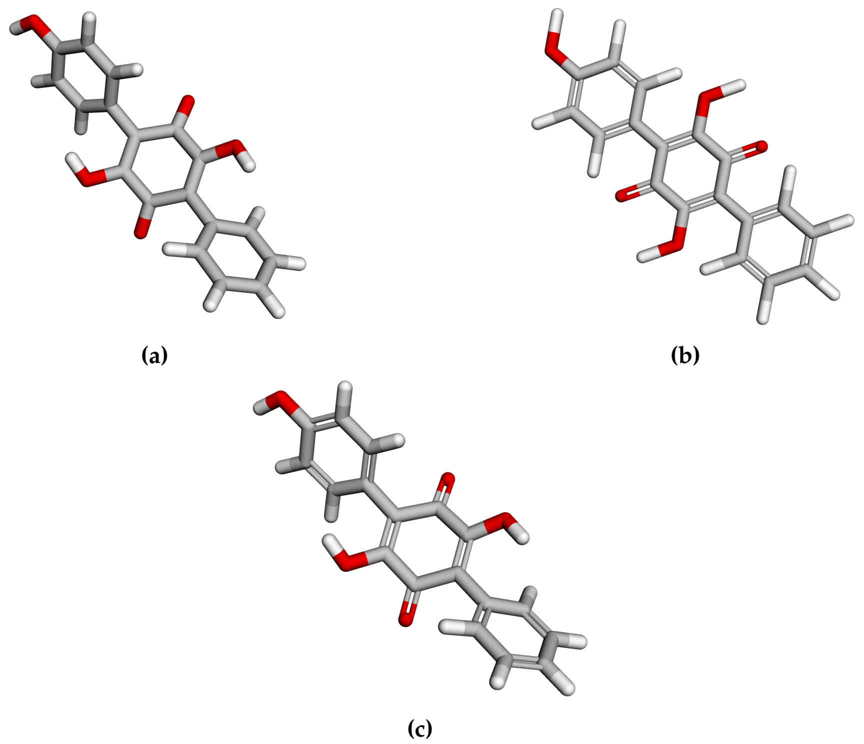

(a) (b) (c)

Figure 3.2 Initial Gaussian (a), 2D SMILES template (b), and Combined 3D (c) structure of the Ascocorynin molecule.

This custom script was used to combine all the Gaussian files and SMILES notations of all the quinones. The obtained RDKit molecule objects were saved as *.mol* files. Unlike Gaussian files, RDKit molecules contain information on a molecule's atoms, 3D structure, bonds, bond types, and many more elaborate atom, bond, and molecule features. Moreover, RDKit molecule files are compatible with common chemical computational tools like Chemprop and OpenBabel, and the RDKit package provides many useful tools to analyze molecules in Python.

As a summary, transformer-based models use four different notations as inputs to determine which notation performs better. 2D-GNN uses SMILES notations to extract atom and bond features as inputs. Lastly, 3D-GNN uses the SMILES-Gaussian combination *.mol* files to extract atom and bond features, including spatial features (bond angle and bond distance) as inputs. All models aim to predict the reduction potential of the quinones and compare which method and input type can provide higher accuracy. All model training and calculations were done using a computer with *Nvidia GeForce RTX 3080* graphics processing unit (GPU). For comparison and simulating conditions where a strong GPU is not present, sample model calculations were done using the CPU as the device instead of utilizing the graphics card. The time it takes for each model to train with GPU vs CPU was noted, as it indicates the computational cost of a model.

To compare the performances of the models, Root Mean Squared Error (RMSE) and Coefficient of Determination (R^2) were used as figures of merit. To compare the performance of the machine learning models, two commonly used regression metrics were employed: Root Mean Squared Error (RMSE) and the Coefficient of Determination (R^2). These metrics provide complementary insights into model accuracy and goodness of fit.

The Root Mean Squared Error (RMSE) quantifies the average magnitude of the prediction errors, penalizing larger errors more heavily due to the squaring operation. It is derived from the Mean Squared Error (MSE), which is also derived from Residual Sum of Squares (SS_R), which is calculated as:

$$SS_R = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{3.1}$$

$$\text{MSE} = \frac{SS_R}{n} \tag{3.2}$$

$$\text{RMSE} = \sqrt{MSE} \tag{3.3}$$

where y_i is the actual and $\hat{y}_i$ is the predicted value by the regression model. Where n is the number of observations.

Here, y_i represents the true value, $\hat{y}_i$ the predicted value, and n the number of data points. RMSE retains the same unit as the target variable, making it directly interpretable in the context of the problem. A lower RMSE indicates better predictive accuracy.

The Coefficient of Determination (R^2), on the other hand, measures how well the model explains the variance in the target variable. It is defined based on the total variability in the data (Total Sum of Squares, SS_T) and the unexplained variability (Residual Sum of Squares,SS_R):

$$Mean = \bar{y} = \frac{\sum_{i=1}^{n} y_i}{n} \tag{3.4}$$

$$SS_T = \sum_{i=1}^{n}(y_i - \bar{y})^2 \tag{3.5}$$

$$R^2 = 1 - \frac{SS_R}{SS_T} \tag{3.6}$$

An R^2 value of 1 indicates perfect prediction, whereas a value of 0 suggests that the model performs no better than simply predicting the mean of the target variable. In this study, RMSE and R^2 were used together to asses prediction errors and model fits.

Chapter 4

Results and discussion

In this chapter, a thorough comparison of all the different models and their results is given. The optimal configuration for each input type of transformer-based models is chosen. Results of the GNN models are also explained. All the best-performing models are compared with each other. As a result, which input with which ML model is the best option for predicting the reduction potential of quinones is determined.

4.1 Results

The results of each model were logged into CSV files during training for assessment. Both validation and test sets' Root Mean Squared Error (RMSE) and Coefficient of Determination (R^2) values were used as metrics. The progression of loss at each epoch and predicted values vs. the real values of the output were utilized to aid in choosing the best configurations.

4.1.1 Transformer based models

First, transformer-based model results will be compared. Different hyperparameters were compared using R Studio, and optimal configurations were chosen. For example, for the ChemBERTa model by plotting learning rate vs test R^2 of the model with Deep SMILES notation inputs and coloring the points by their gamma values in figure 4.1a, it can be determined that for all learning rate values, 0.95 gamma values performs the best and the learning rate 1e-04 and 5e-05 performs similarly.

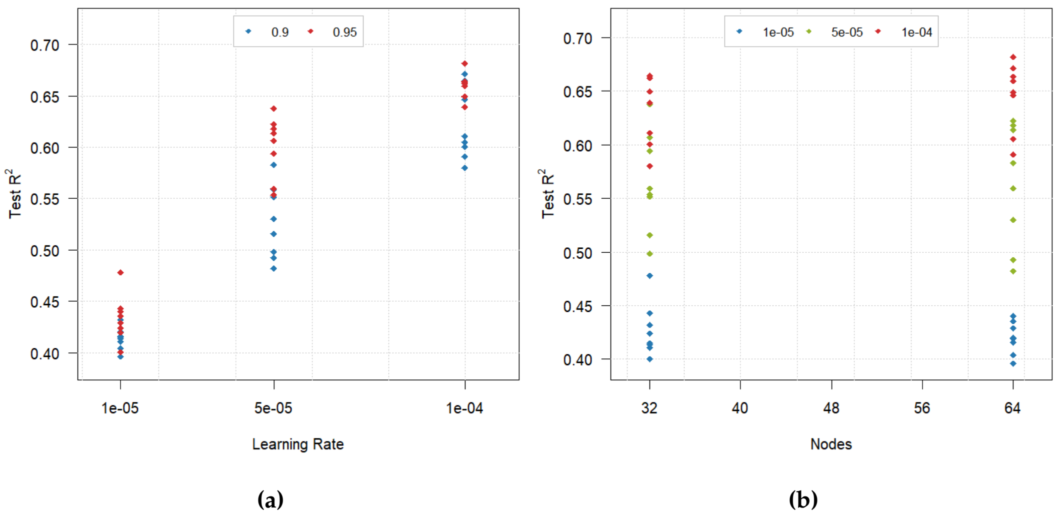


Figure 4.1 Learning Rate vs test R^2 plot, colored by gamma values (a), and Number of nodes vs test R^2 plot, colored by learning rates(b) plots of ChemBERTa model with Deep SMILES notation inputs

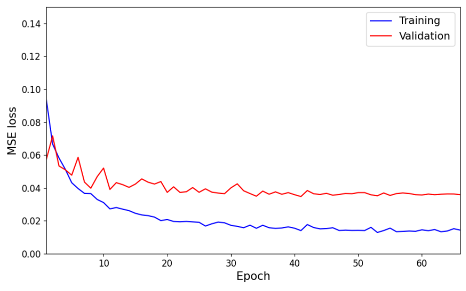
By subsetting the dataset to choose only gamma = 0.95 models, and comparing other parameters, like test R^2 vs. the number of nodes in the first hidden layer in Figure 4.1b, and coloring the graph by learning rate, it can be seen that the optimal learning rate for this model is 1e-04.

Similarly, if batch size is plotted against a loss metric, optimal weight decay can be determined as 0.1. The best configurations were systematically determined by repeating these steps across all input notation types and models. In the provided example involving Deep SMILES notation input, the results had a clear distinction, making selecting an optimal configuration easier than other notation types. However, for other models and notations, there was no clear distinction every time. In scenarios where the results were closer and choosing a configuration proved more challenging, epoch vs loss graphs, validation and test R^2 plots, and predicted vs true output graphs were all taken into account. As a result, the final model setups were chosen and presented in Table 4.1.

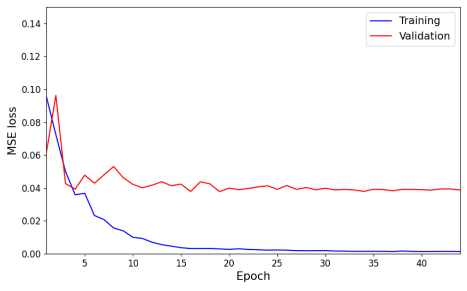
Table 4.1 Chosen optimal configurations for transformer-based models

Input notation	Configuration no.	Hidden layer nodes	Mean validation R^2	Mean test R^2
ChemBERTa				
SMILES	19	64-32-1	0.592	0.666
SMILES Canonical	19	64-32-1	0.547	0.628
Deep SMILES	19	64-32-1	0.532	0.560
InChI	20	64-32-1	0.157	0.343
PubChem				
SMILES	16	64-32-1	0.623	0.662
SMILES Canonical	16	64-32-1	0.580	0.740
Deep SMILES	20	32-16-1	0.595	0.537
InChI	21	32-16-1	0.234	0.494

The ChemBERTa model performed best when the hidden layer structure was with 64 and 32 nodes for all input types, while the PubChem model had mixed results, with half of the notations performing better with 32 and 16-node hidden layers. Optimal batch size and weight decay have varied results for different input types. However, for some notations, batch size and weight decay changes did not have very significant differences. In those cases, epoch vs loss graphs and validation performances were taken into account to determine the best configuration. Another thing to note is how the test set performs better than the validation set. It is not an expected behaviour since the model is evaluated over the validation set after each epoch, and the validation results guide the hyperparameter adjustments. For ChemBERTa models, SMILES notation inputs, and for PubChem models, SMILES Canonical inputs were chosen as the best. Computation times were also recorded, revealing that training a ChemBERTa model takes approximately 3 minutes, whereas training a PubChem model requires about 10 minutes when the calculations were done with the GPU. However, these numbers were significantly higher when only the CPU was available for calculations, with ChemBERTa taking 50 minutes, and the PubChem taking 3 hours to train a single model.



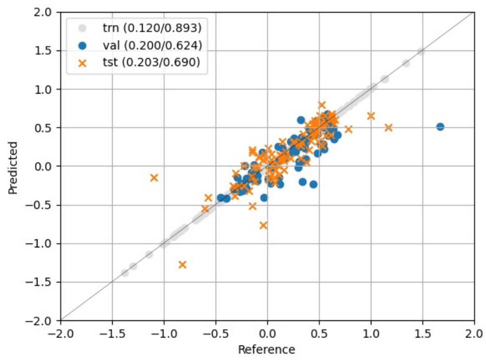
(a)



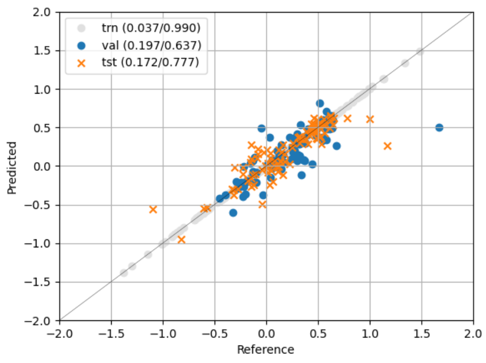
(b)

Figure 4.2 Epoch vs Loss graphs of ChemBERTa model with SMILES (a), and PubChem model with SMILES Canonical (b) inputs.

By examining epoch vs loss graphs in Figure 4.2, a trend can be seen. The training loss decreases at a rapid pace, while the validation loss does not improve significantly after a few initial epochs. The number of epochs does not always reach 100 either, due to the early stopping system that was added to avoid overfitting. This countermeasure seems to be functioning well since no significant overfitting can be seen in graphs where the validation loss shows an increasing trend. Overall, the epoch vs loss plots show an expected trend where the training loss is lower than validation, and they both become stable after the model has been through sufficient epochs. The main things to note in these plots would be that the ChemBERTa model has closer validation and training losses, while the PubChem model shows a bigger discrepancy.



(a)



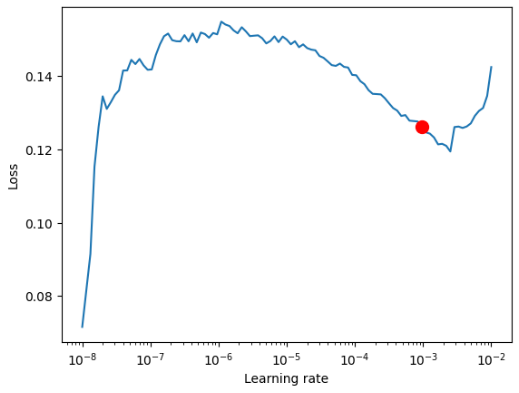
(b)

Figure 4.3 True vs Predicted reduction potential E[V] graphs of ChemBERTa model with SMILES (a), and PubChem model with SMILES Canonical (b) inputs.

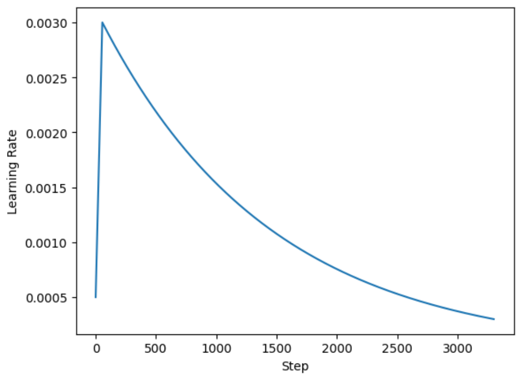
The true and predicted values of the reduction potentials of the quinones in the test and validation sets were also plotted for comparison for all the chosen model configurations in figures 4.3a and 4.3b.

4.1.2 GNN models

To choose the learning rate intervals, the learning rate finder tool from the lightning package was used. With the help of the plot that the learning rate finder computed, the optimal initial learning rate was determined as 0.0005, the maximum learning rate as 0.003, and the final learning rate as 0.0003. In Figure 4.4, the results of the learning finder, and how the Noam learning rate scheduler changes the learning rate through epochs of the GNNs can be seen.



(a)



(b)

Figure 4.4 Results of the learning rate finder (a) and the progression of the learning rate of GNNs over batch steps (b).

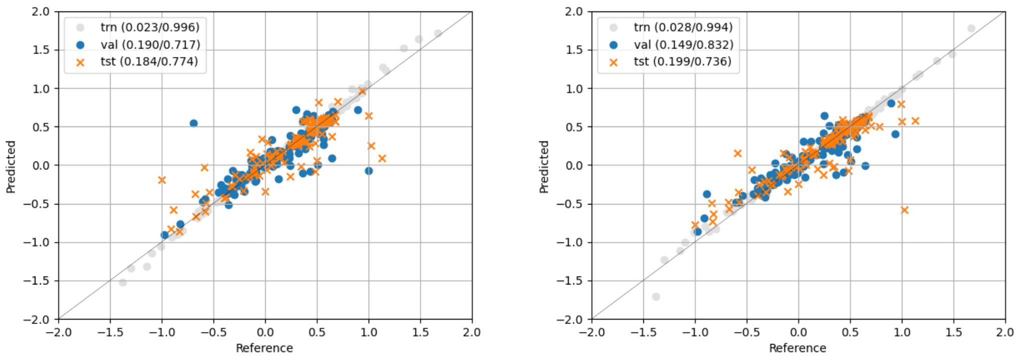
Following a similar approach to that used for selecting the optimal configuration of transformer-based models, each GNN model was trained three times per configuration, with the average performance used to identify the best option. Training a 2D-GNN model took only 1 minute, and a 3D-GNN model required around 7 minutes to complete on a GPU. When using CPU only, their computation times did not show a significant increase; a 2D-GNN model took around 3 minutes, and a 3D-GNN model took 10 minutes to complete training. Meaning, training GNNs without a GPU takes less time than transformer-based pre-trained models take with a strong GPU. This shows that even without a high-spec GPU available, GNN models are a viable choice.

In Table 4.2, chosen configurations for GNN models and their mean R^2 values for the test and validation sets can be seen.

Table 4.2 Results of the GNN models

Model	Configuration no.	Mean Validation R^2	Mean Test R^2
2D-GNN	2	0.840	0.742
3D-GNN	15	0.809	0.727

The issue with the test set performing better than the validation set does not exist for the GNN models. As per the expected behaviour, validation sets fit the GNN models better than the test sets. The predicted vs true reduction potential values of the best performing models with the chosen configurations are also plotted and given in Figure 4.5.



(a) **(b)**

Figure 4.5 True vs Predicted reduction potential E[V] plots of 2D-GNN(a) and 3D-GNN(b)

4.1.3 Statistical comparison of model performances

To enable a robust comparison between the machine learning methods explored in this study, a final evaluation was conducted on the best-performing configuration of each model type. The selected models were: ChemBERTa (configuration 19 with SMILES input), PubChem (configuration 16 with SMILES Canonical input), the 2D-GNN (configuration 2), and the 3D-GNN (configuration 15). Each model was trained independently 30 times to capture the variability in training performance.

The root mean square error (RMSE) on the validation set was chosen as the primary metric for statistical evaluation. Figure 4.6 displays the distribution of RMSE values across the 30 runs for each model, and the corresponding mean RMSE and R^2 scores on both validation and test sets are summarized in Table 4.3. These results serve as the basis for the subsequent statistical analyses.

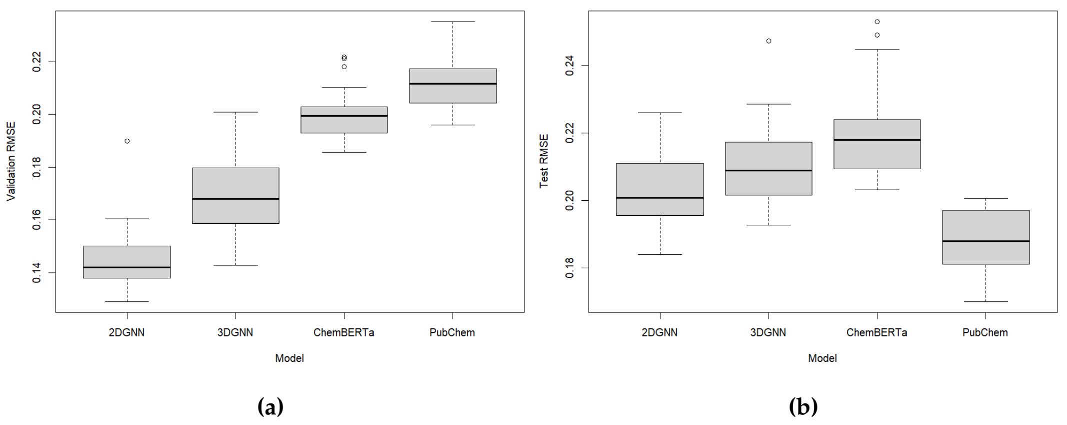


Figure 4.6 Box plots for the validation (a) and test (b) RMSE values of each model

Table 4.3 Results of models

Model	Mean Validation RMSE	Mean Validation R^2	Mean Test RMSE	Mean Test R^2
Chemberta	0.200	0.626	0.219	0.637
PubChem	0.211	0.581	0.188	0.734
2D-GNN	0.145	0.834	0.204	0.721
3D-GNN	0.168	0.782	0.210	0.704

To assess whether the observed differences in model performance are statistically significant, an analysis of variance (ANOVA) was conducted. This test evaluates the null hypothesis that the mean RMSE values of all models are equal, formally expressed as:

$$H_0 : \mu_1 = \mu_2 = \mu_3 = \mu_4 \tag{4.1}$$

where μ_i denotes the mean validation RMSE of model i. The ANOVA test computes an F-statistic, defined as the ratio of between-group variance to within-group variance:

$$F = \frac{\text{MS}_{\text{between}}}{\text{MS}_{\text{within}}} \tag{4.2}$$

Here, $\text{MS}_{\text{between}}$ represents the mean square error between the model groups, and $\text{MS}_{\text{within}}$ corresponds to the average variance within each model group. A resulting p-value less than the significance threshold ($\alpha = 0.05$) indicates that at least one model exhibits statistically different performance from the others [104].

The ANOVA test yielded a statistically significant result ($p < 0.05$), rejecting the null hypothesis and justifying further pairwise comparisons through Tukey's

Honestly Significant Difference (HSD) test for all pairs except 2D and 3D-GNN, as well as 3D-GNN and ChemBERTa. This post-hoc analysis determines which specific model pairs differ in performance by computing confidence intervals for the differences in their group means. The Tukey HSD criterion uses the following formula:

$$\text{HSD} = q_{\alpha,k,N-k} \cdot \sqrt{\frac{\text{MS}_{\text{within}}}{n}} \tag{4.3}$$

In this equation, $q_{\alpha,k,N-k}$ is the critical value from the studentized range distribution for significance level α, k is the number of groups, N is the total number of observations, and n is the number of observations per group. If the confidence interval for a mean difference between two models does not include zero, the difference is deemed statistically significant. The resulting pairwise comparisons are visualized in Figure 4.7 for all model combinations.

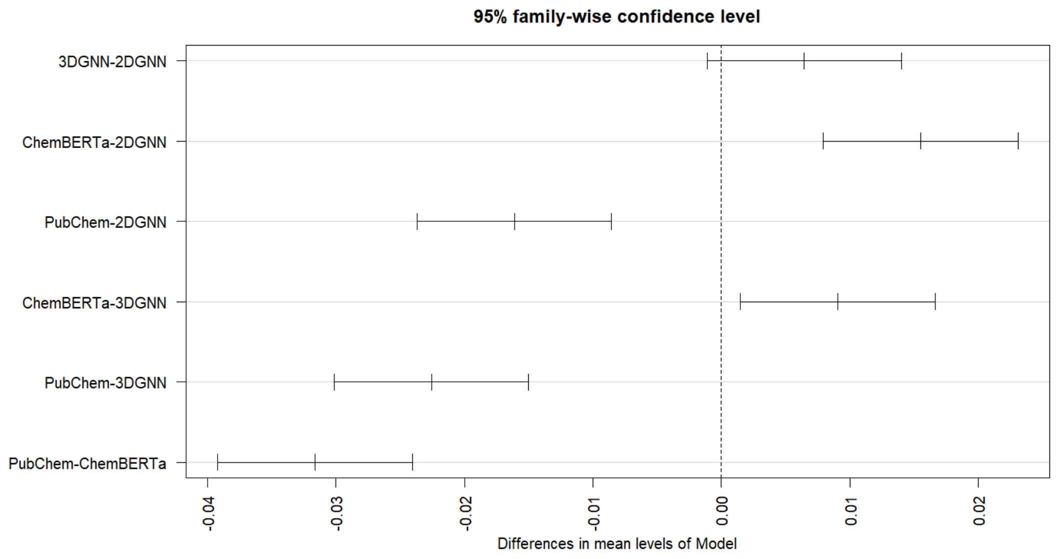


Figure 4.7 Tukey HSD results of the models

Based on the test outcomes, ChemBERTa exhibited the weakest overall performance, with the highest mean RMSE on the test dataset. The PubChem model achieved the best test set performance, showing the lowest RMSE values among all models. The Tukey HSD test confirmed that PubChem's test RMSE was significantly lower than those of the other three models. However, a counterintuitive observation, which was already noted previously, was made in that PubChem-BERT also had the lowest validation set accuracy among all. This discrepancy may suggest model fitting issues that warrant further investigation.

The 2D-GNN model demonstrated the highest validation set accuracy and also achieved the second-best performance on the test set, indicating strong generalizability. Besides, the addition of 3D structural features in the 3D-GNN did not lead to a statistically significant improvement. According to the Tukey test, the difference in performance between the 2D and 3D GNNs was not significant at the 95% confidence level, suggesting that incorporating 3D molecular geometry does not provide added value in this specific prediction task.

Moreover, the statistical comparisons validated a performance distinction between transformer-based language models (LLMs) and graph neural networks (GNNs), with the ANOVA and Tukey tests indicating significant differences in their predictive capabilities.

4.2 Discussion

For both types of models, certain molecules consistently appear as outliers, as illustrated in Figures 4.3 and 4.5. These recurring outliers are listed in Table 4.4. As briefly mentioned in the previous chapter, further investigation revealed that furasubins behave markedly differently in experimental settings compared to predictions made by DFT calculations [105]. Phenicin, which has the highest reduction potential value in the dataset, was observed to exhibit a reduction potential of -0.216V in experimental studies, contradicting DFT predictions. These findings show that ML models might perform more accurately if the provided training set has more reliable data and the DFT dataset is revised. In contrast, the Altertoxin I and julichrome Q3,3 molecules stand out as outliers in the ML models without any clear experimental or structural explanation. One possible reason for this may be Julichrome's unusually large size, which contains 38 carbon atoms, making it significantly more complex than most other molecules in the dataset. This increased complexity may challenge the models' ability to accurately capture their behavior.

Table 4.4 Outlier quinones

Quinone	Formula	Standard reduction potential (V)	Errored model
5-Hydroxy-3-methoxy dihydrofusarubin D	$C_{16}H_{20}O_7$	1.028	GNNs
5-Hydroxy dihydrofusarubin B	$C_{15}H_{18}O_7$	1.126	ChemBERTa, PubChem, 2D-GNN
Phenicin	$C_{14}H_{10}O_6$	1.666	ChemBERTa, PubChem
Julichrome Q3, 3	$C_{38}H_{38}O_{16}$	-0.585	3D-GNN
Altertoxin I	$C_{20}H_{16}O_6$	-1.094	ChemBERTa, PubChem

Among transformer-based models, the PubChem model performed better on the test sets, with the SMILES Canonical notation input being the best with a mean 0.734 test R^2 value. However, the validation set had significantly worse results than the test set, with 0.581 R^2. This issue exists across all the input notation types for all transformer-based models. While this problem was less prevalent on the ChemBERTa model, the best R^2 value of the validation sets was 0.626, while it was 0.637 for the test set with SMILES input. Stratified split, in theory, should distribute quinone molecules to the sets equally according to their reduction potentials and minimize the chance of lucky splits. To test if this was due to a lucky split, different splitting methods were tested. The same issue was observed with the random splitting method and different split ratios. This is not an expected behaviour since ML models are evaluated by the validation set during training and aim to minimize the validation loss. The validation set is expected to have a better fit than the test set. These results can point to potential model fitting issues. Apart from this issue, the PubChem model with SMILES Canonical notation input was determined as the best-performing transformer model.

While GNNs showed slightly poorer accuracy on test sets, they exhibited the expected behaviour of validation set fitting better than the test set. This might be due to both the different architecture of the MPNN models and the structure-based splitting method, Kennard Stone split, that the Chemprop tool provides. Between 2D and 3D GNNs, 2D-GNN has a better performance. This was a surprising result since the 3D-GNN has the additional featurizers with bond distances and bond angles, and the atoms are positioned on the accurate 3D coordinates provided by their Gaussian files. These extra features were expected to help the model understand the complex structure of the quinones and predict their reduction potentials better. However, it seems that the 2D features were enough to capture the fundamental patterns of the quinone molecules. While the mean test R^2 value of 0.721 of the 2D-GNNs with optimal configuration is slightly lower than the PubChem/SMILES Canonical models, the validation R^2 value of 0.834 is significantly higher and makes the model fitting more reliable.

In addition to the performance differences, training a 2D-GNN model without an available GPU takes only 3 minutes, whereas PubChem requires 3 hours. With a strong GPU, the difference is smaller, as 2D-GNN takes 1 minute while PubChem requires 10 minutes. Even though it might seem as only 9 minutes of difference, it is still 9 times more computing time. This computation cost can add up if fine-tuning is necessary, as it was in this case, and many different configurations should be trialed multiple times. There is a significant computational cost difference, even for a high-spec computer that was used in this work. With a lower-spec computer, this gap can prove to be even larger, making the PubChem model a non-viable option for cases with limited budget and time. In such cases, 2D-GNNs provide significantly faster training times with a very slight drop in test set accuracy.

Furthermore, the best-performing model out of all the trials for the PubChem model shows a 0.777 test R^2 value, while the best 2D-GNN has a test R^2 value of 0.774. These two have very close values, and obtaining an as accurate model with 2D-GNNs might be possible with more trial runs. With that said, if time and budget are not a concern and obtaining the highest possible test set performance is the goal, the PubChem model would be the ideal choice.

As a future direction, it can be aimed to collect more data on different quinone molecules to have a larger dataset than 988. While the current amount was sufficient to do initial testing, configuration, and decision making, a larger dataset will provide more material to both train and test models. The solubility of the redox active compounds is just as important as their reduction potential for their performance in RFBs. Hence, another opportunity to utilize the ML models would be to deploy them to predict the solvation energies of quinones. This can be done in a process similar to the one provided in this study. That said, these methods and models can be used for many applications in molecular discovery and drug design fields, as the investigated literature suggests.

Chapter 5

Conclusions

Quinones are redox-active organic compounds that are renewable, configurable, and obtainable from fungi and bacteria. The abundance and the wide array of structures and properties they provide make them good candidates for electrolytes in RFBs. ML methods can cut down the resources and time required for determining good quinone candidates by predicting their reduction potentials. Transformer-based models have options that are pre-trained on relevant large molecular data. The tested ChemBERTa and PubChem models, which performed well on the test sets with R^2 values of 0.637 and 0.734, exhibited unexpectedly poor performance on the validation sets, with R^2 values of 0.626 and 0.582, respectively. Apart from this issue, the PubChem model with SMILES Canonical inputs has the overall best performance on the test set. GNN models proved that the addition of spatial features does not improve the model accuracy. On the contrary, 2D-GNNs performed, albeit statistically insignificantly, better than the 3D-GNNs. Among all the models, 2D-GNNs showed the second-best performance behind the PubChem model. Although the PubChem model proved to be the superior model, the training time and computational cost of the model were also significantly higher compared to the GNN models. With the high computational cost and the validation set fitting issues in mind, the choice of the best model for the task of predicting the standard reduction potentials of quinones depends on specific conditions such as time and budget restrictions and precision requirements.

References

[1] Omar Ellabban, Haitham Abu-Rub, and Frede Blaabjerg. "Renewable energy resources: Current status, future prospects and their enabling technology". In: *Renewable and Sustainable Energy Reviews* 39 (Nov. 2014), pp. 748–764. ISSN: 1364-0321. DOI: 10.1016/J.RSER.2014.07.113. URL: https://www.sciencedirect.com/science/article/pii/S1364032114005656.

[2] A. R. Dehghani-Sanij et al. "Study of energy storage systems and environmental challenges of batteries". In: *Renewable and Sustainable Energy Reviews* 104 (Apr. 2019), pp. 192–208. ISSN: 1364-0321. DOI: 10.1016/J.RSER.2019.01.023. URL: https://www.sciencedirect.com/science/article/pii/S1364032119300334.

[3] Fuead Hasan, Vivekananda Mahanta, and Ahmed Adel Abdelazeez. *Quinones for Aqueous Organic Redox Flow Battery: A Prospective on Redox Potential, Solubility, and Stability*. 2023. DOI: 10.1002/admi.202300268.

[4] Kaixiang Lin et al. "Alkaline quinone flow battery". In: *Science* 349 (6255 Sept. 2015), pp. 1529–1532. ISSN: 10959203. DOI: 10.1126/SCIENCE.AAB3033,. URL: https://pubmed.ncbi.nlm.nih.gov/26404834/.

[5] Sebastian Birkedal Kristensen et al. "Simulation of electrochemical properties of naturally occurring quinones". In: *Scientific Reports* 10 (1 Dec. 2020), pp. 1–10. ISSN: 20452322. DOI: 10.1038/S41598-020-70522-Z;SUBJMETA=114,45,631;KWRD=BIOCHEMISTRY,COMPUTATIONAL+BIOLOGY+AND+BIOINFORMATICS. URL: https://www.nature.com/articles/s41598-020-70522-z.

[6] Cuiping Han et al. *Organic quinones towards advanced electrochemical energy storage: Recent advances and challenges*. 2019. DOI: 10.1039/c9ta05252f.

[7] Ki Chul Kim et al. "Unveiled correlations between electron affinity and solvation in redox potential of quinone-based sodium-ion batteries". In: *Energy Storage Materials* 19 (May 2019), pp. 242–250. ISSN: 2405-8297. DOI: 10.1016/J.ENSM.2019.01.017.

[8] James Urban et al. "Predicting glycan structure from tandem mass spectrometry via deep learning". In: *Nature Methods* 21 (7 2024). ISSN: 15487105. DOI: 10.1038/s41592-024-02314-6.

[9] Francesca Grisoni. *Chemical language models for de novo drug design: Challenges and opportunities*. 2023. DOI: 10.1016/j.sbi.2023.102527.

[10] Michael W. Mullowney et al. *Artificial intelligence for natural product drug discovery*. 2023. DOI: 10.1038/s41573-023-00774-7.

[11] Ryotaro Okabe et al. “Virtual Node Graph Neural Network for Full Phonon Prediction”. In: *Nature Computational Science* 4 (7 July 2024). DOI: 10.1038/s43588-024-00661-0. URL: https://doi.org/10.1038/s43588-024-00661-0.

[12] Kenneth O. Eyong, Victor Kuete, and Thomas Efferth. “Quinones and Benzophenones from the Medicinal Plants of Africa”. In: *Medicinal Plant Research in Africa: Pharmacology and Chemistry* (Jan. 2013), pp. 351–391. DOI: 10.1016/B978-0-12-405927-6.00010-2. URL: https://www.sciencedirect.com/science/article/abs/pii/B9780124059276000102.

[13] Bo Yang et al. “High-Performance Aqueous Organic Flow Battery with Quinone-Based Redox Couples at Both Electrodes”. In: *Journal of The Electrochemical Society* 163 (7 2016). ISSN: 0013-4651. DOI: 10.1149/2.1371607jes.

[14] Giorgio Lenaz et al. “Mitochondrial Quinone Reductases: Complex I”. In: *Methods in Enzymology* 382 (Jan. 2004), pp. 3–20. ISSN: 0076-6879. DOI: 10.1016/S0076-6879(04)82001-9.

[15] Antony R. Crofts et al. “Mechanism of ubiquinol oxidation by the bc1 complex: Role of the iron sulfur protein and its mobility”. In: *Biochemistry* 38 (48 1999). ISSN: 00062960. DOI: 10.1021/bi990961u.

[16] Da bao Chen et al. “Quinones as preventive agents in Alzheimer’s diseases: focus on NLRP3 inflammasomes”. In: *The Journal of pharmacy and pharmacology* 72 (11 Nov. 2020), pp. 1481–1490. ISSN: 2042-7158. DOI: 10.1111/JPHP.13332. URL: https://pubmed.ncbi.nlm.nih.gov/32667050/.

[17] Eliane Teixeira Sousa, Wilson A Lopes, and Jailson B de Andrade. “FONTES, FORMAÇÃO, REATIVIDADE E DETERMINAÇÃO DE QUINONAS NA ATMOSFERA”. In: *Quim. Nova* 39 (4 2016), pp. 486–495. DOI: 10.5935/0100-4042.20160034. URL: http://dx.doi.org/10.5935/0100-4042.20160034.

[18] Michio Kurosu and Eeshwaraiah Begari. *Vitamin K2 in electron transport system: Are enzymes involved in vitamin K2 biosynthesis promising drug targets?* 2010. DOI: 10.3390/molecules15031531.

[19] Roger C. Prince, P. Leslie Dutton, and M. R. Gunner. “The aprotic electrochemistry of quinones”. In: *Biochimica et Biophysica Acta (BBA) - Bioenergetics* 1863 (6 Aug. 2022), p. 148558. ISSN: 0005-2728. DOI: 10.1016/J.BBABIO.2022.148558.

[20] K. Namsheer and Chandra Sekhar Rout. "Conducting polymers: a comprehensive review on recent advances in synthesis, properties and applications". In: *RSC Advances* 11 (10 Jan. 2021), pp. 5659–5697. ISSN: 2046-2069. DOI: 10.1039/D0RA07800J. URL: https://pubs.rsc.org/en/content/articlehtml/2021/ra/d0ra07800jhttps://pubs.rsc.org/en/content/articlelanding/2021/ra/d0ra07800j.

[21] Minjoon Park et al. "Material design and engineering of next-generation flow-battery technologies". In: *Nature Reviews Materials* 2 (1 Nov. 2016), pp. 1–18. ISSN: 20588437. DOI: 10.1038/NATREVMATS.2016.80. URL: https://www.nature.com/articles/natrevmats201680.

[22] Zening Li et al. "Recent Progress in Organic Species for Redox Flow Batteries". In: *Energy Storage Materials* 50 (Sept. 2022), pp. 105–138. ISSN: 2405-8297. DOI: 10.1016/J.ENSM.2022.04.038. URL: https://www.sciencedirect.com/science/article/pii/S2405829722002331.

[23] M. Rychcik and M. Skyllas-Kazacos. "Characteristics of a new all-vanadium redox flow battery". In: *Journal of Power Sources* 22 (1 Jan. 1988), pp. 59–67. ISSN: 0378-7753. DOI: 10.1016/0378-7753(88)80005-3. URL: https://www.sciencedirect.com/science/article/pii/0378775388800053.

[24] Kyle Lourenssen et al. "Vanadium redox flow batteries: A comprehensive review". In: *Journal of Energy Storage* 25 (Oct. 2019), p. 100844. ISSN: 2352-152X. DOI: 10.1016/J.EST.2019.100844. URL: https://www.sciencedirect.com/science/article/pii/S2352152X19302798.

[25] Sung Hee Shin, Sung Hyun Yun, and Seung Hyeon Moon. "A review of current developments in non-aqueous redox flow batteries: characterization of their membranes for design perspective". In: *RSC Advances* 3 (24 May 2013), pp. 9095–9116. ISSN: 2046-2069. DOI: 10.1039/C3RA00115F. URL: https://pubs.rsc.org/en/content/articlelanding/2013/ra/c3ra00115f.

[26] Bo Yang et al. "An Inexpensive Aqueous Flow Battery for Large-Scale Electrical Energy Storage Based on Water-Soluble Organic Redox Couples". In: *Journal of The Electrochemical Society* 161 (9 2014), A1371–A1380. ISSN: 0013-4651. DOI: 10.1149/2.1001409JES. URL: https://www.researchgate.net/publication/303749939_An_Inexpensive_Aqueous_Flow_Battery_for_Large-Scale_Electrical_Energy_Storage_Based_on_Water-Soluble_Organic_Redox_Couples.

[27] Brian Huskinson et al. "A metal-free organic-inorganic aqueous flow battery". In: *Nature* 505 (7482 2014). ISSN: 00280836. DOI: 10.1038/nature12909.

[28] Kaixiang Lin et al. "A redox-flow battery with an alloxazine-based organic electrolyte". In: *Nature Energy* 1 (9 2016). ISSN: 20587546. DOI: 10.1038/nenergy.2016.102.

[29] Sai Prasad Ega and Palaniappan Srinivasan. “Quinone materials for supercapacitor: Current status, approaches, and future directions”. In: *Journal of Energy Storage* 47 (Mar. 2022), p. 103700. ISSN: 2352-152X. DOI: 10.1016/J.EST.2021.103700.

[30] Thiago Bertaglia et al. “Eco-friendly, sustainable, and safe energy storage: a nature-inspired materials paradigm shift”. In: *Cite this: Mater. Adv* 5 (2024), p. 7534. DOI: 10.1039/d4ma00363b.

[31] Théophile Gaudin and Jean Marie Aubry. “Prediction of Pourbaix diagrams of quinones for redox flow battery by COSMO-RS”. In: *Journal of Energy Storage* 49 (May 2022), p. 104152. ISSN: 2352-152X. DOI: 10.1016/J.EST.2022.104152. URL: https://www.sciencedirect.com/science/article/pii/S2352152X22001864.

[32] C. E. Senseman and O. A. Nelson. “Catalytic Oxidation of Anthracene to Anthraquinone”. In: *Industrial and Engineering Chemistry* 15 (5 May 1923), pp. 521–524. ISSN: 00197866. DOI: 10.1021/IE50161A040. URL: /doi/pdf/10.1021/ie50161a040.

[33] A. Roy and S. Aditya. “A novel photogalvanic cell using anthraquinone-2-sulfonate”. In: *International Journal of Hydrogen Energy* 8 (2 Jan. 1983), pp. 91–96. ISSN: 0360-3199. DOI: 10.1016/0360-3199(83)90091-5. URL: https://www.sciencedirect.com/science/article/pii/0360319983900915.

[34] Wei Wang et al. “Anthraquinone with tailored structure for a nonaqueous metal–organic redox flow battery”. In: *Chemical Communications* 48 (53 June 2012), pp. 6669–6671. ISSN: 1364-548X. DOI: 10.1039/C2CC32466K. URL: https://pubs.rsc.org/en/content/articlelanding/2012/cc/c2cc32466k.

[35] Charlotte Overgaard Wilhelmsen et al. “Demonstrating the Use of a Fungal Synthesized Quinone in a Redox Flow Battery”. In: *Batteries Supercaps* 6 (1 Jan. 2023), e202200365. ISSN: 2566-6223. DOI: 10.1002/batt.202200365. URL: https://chemistry-europe.onlinelibrary.wiley.com/doi/10.1002/batt.202200365.

[36] Wen Yan et al. “All-polymer particulate slurry batteries”. In: *Nature Communications* 10 (1 Dec. 2019), pp. 1–11. ISSN: 20411723. DOI: 10.1038/S41467-019-10607-0. URL: https://www.nature.com/articles/s41467-019-10607-0.

[37] J. C. Barbosa et al. “Molecular design of functional polymers for organic radical batteries”. In: *Energy Storage Materials* 60 (2023). ISSN: 24058297. DOI: 10.1016/j.ensm.2023.102841.

[38] David Weininger. "SMILES, a Chemical Language and Information System: 1: Introduction to Methodology and Encoding Rules". In: *Journal of Chemical Information and Computer Sciences* 28 (1 Feb. 1988), pp. 31–36. ISSN: 00952338. DOI: 10.1021/CI00057A005. URL: https://pubs.acs.org/doi/abs/10.1021/ci00057a005.

[39] Jonathan M. Goodman et al. "InChI version 1.06: now more than 99.99% reliable". In: *Journal of Cheminformatics* 13 (1 Dec. 2021), pp. 1–8. ISSN: 17582946. DOI: 10.1186/S13321-021-00517-Z. URL: https://jcheminf.biomedcentral.com/articles/10.1186/s13321-021-00517-z.

[40] David Weininger, Arthur Weininger, and Joseph L. Weininger. "SMILES. 2. Algorithm for Generation of Unique SMILES Notation". In: *Journal of Chemical Information and Computer Sciences* 29 (2 1989), pp. 97–101. ISSN: 00952338. DOI: 10.1021/CI00062A008. URL: https://pubs.acs.org/doi/abs/10.1021/ci00062a008.

[41] Noel M O'boyle and Andrew Dalke. "DeepSMILES: An Adaptation of SMILES for Use in Machine-Learning of Chemical Structures". In: (Sept. 2018). DOI: 10.26434/CHEMRXIV.7097960.V1. URL: https://chemrxiv.org/engage/chemrxiv/article-details/60c73ed6567dfe7e5fec388d.

[42] Stephen R. Heller et al. "InChI, the IUPAC International Chemical Identifier". In: *Journal of Cheminformatics* 7 (1 May 2015), p. 23. ISSN: 17582946. DOI: 10.1186/S13321-015-0068-4. URL: https://pmc.ncbi.nlm.nih.gov/articles/PMC4486400/.

[43] Yann Lecun, Yoshua Bengio, and Geoffrey Hinton. "Deep learning". In: *Nature 2015 521:7553* 521 (7553 May 2015), pp. 436–444. ISSN: 1476-4687. DOI: 10.1038/nature14539. URL: https://www.nature.com/articles/nature14539.

[44] M. I. Jordan and T. M. Mitchell. "Machine learning: Trends, perspectives, and prospects". In: *Science* 349 (6245 July 2015), pp. 255–260. ISSN: 10959203. DOI: 10.1126/SCIENCE.AAA8415. URL: https://www.science.org/doi/10.1126/science.aaa8415.

[45] Shilpa Shilpa, Gargee Kashyap, and Raghavan B. Sunoj. "Recent Applications of Machine Learning in Molecular Property and Chemical Reaction Outcome Predictions". In: *The journal of physical chemistry. A* 127 (40 Oct. 2023), pp. 8253–8271. ISSN: 1520-5215. DOI: 10.1021/ACS.JPCA.3C04779. URL: https://pubmed.ncbi.nlm.nih.gov/37769193/.

[46] Fariha Imam, Petr Musilek, and Marek Z. Reformat. "Parametric and Nonparametric Machine Learning Techniques for Increasing Power System Reliability: A Review". In: *Information 2024, Vol. 15, Page 37* 15 (1 Jan. 2024), p. 37.

ISSN: 2078-2489. DOI: 10.3390/INFO15010037. URL: https://www.mdpi.com/2078-2489/15/1/37/htmhttps://www.mdpi.com/2078-2489/15/1/37.

[47] Kevin P. Murphy. *Machine learning: a probabilistic perspective (adaptive computation and machine learning series)*. MIT Press, 2012. ISBN: 978-0-262-01802-9.

[48] George Philipp and Jaime G. Carbonell. "Nonparametric Neural Networks". In: *5th International Conference on Learning Representations, ICLR 2017 - Conference Track Proceedings* (Dec. 2017). DOI: 10.48550/arXiv.1712.05440. URL: https://arxiv.org/pdf/1712.05440.

[49] Kit Yan Chan et al. "Deep neural networks in the cloud: Review, applications, challenges and research directions". In: *Neurocomputing* 545 (Aug. 2023), p. 126327. ISSN: 0925-2312. DOI: 10.1016/J.NEUCOM.2023.126327. URL: https://www.sciencedirect.com/science/article/pii/S0925231223004502.

[50] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. URL: http://www.deeplearningbook.org.

[51] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. "Multilayer feedforward networks are universal approximators". In: *Neural Networks* 2.5 (1989), pp. 359–366. ISSN: 0893-6080. DOI: https://doi.org/10.1016/0893-6080(89)90020-8. URL: https://www.sciencedirect.com/science/article/pii/0893608089900208.

[52] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. "Deep Sparse Rectifier Neural Networks". In: *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*. Ed. by Geoffrey Gordon, David Dunson, and Miroslav Dudík. Vol. 15. Proceedings of Machine Learning Research. Fort Lauderdale, FL, USA: PMLR, 2011, pp. 315–323. URL: https://proceedings.mlr.press/v15/glorot11a.html.

[53] Daniel Svozil, Vladimír Kvasnicka, and Jirí Pospichal. "Introduction to multilayer feed-forward neural networks". In: *Chemometrics and Intelligent Laboratory Systems* 39.1 (1997), pp. 43–62. ISSN: 0169-7439. DOI: https://doi.org/10.1016/S0169-7439(97)00061-0. URL: https://www.sciencedirect.com/science/article/pii/S0169743997000610.

[54] Rui Zhao et al. "Deep learning and its applications to machine health monitoring". In: *Mechanical Systems and Signal Processing* 115 (Jan. 2019), pp. 213–237. ISSN: 0888-3270. DOI: 10.1016/J.YMSSP.2018.05.050. URL: https://www.sciencedirect.com/science/article/pii/S0888327018303108.

[55] MATLAB Simulink. *Convolutional Neural Network - MATLAB & Simulink*. URL: https://www.mathworks.com/discovery/convolutional-neural-network.html. (Accessed May 2025).

[56] Muhammad Imron Rosadi, Lukman Hakim, and M. Faishol A. "Classification of Coffee Leaf Diseases using the Convolutional Neural Network (CNN) EfficientNet Model". In: *IAIC International Conference Series* 4 (1 Dec. 2023), pp. 58–69. ISSN: 2774-5899. DOI: 10.34306/CONFERENCESERIES.V4I1.627. URL: https://aptikom-journal.id/conferenceseries/article/view/627.

[57] Artur Sobolewski and Kamil Szyc. "Architecture optimization techniques for Convolutional Neural Networks: further experiments and insights". In: *International Journal of Electronics and Telecommunications* vol. 70.No 1 (2024), 87–94. DOI: 10.24425/ijet.2024.149518. URL: http://journals.pan.pl/Content/130697/11_4458_Sobolewski_L_sk.pdf.

[58] Nurul Atikah Mazlan et al. "Convolution Neural Network (CNN) Architectures Analysis for Photovoltaic (PV) Module Defect Images Classification". In: *2022 International Conference on Computer Engineering, Network, and Intelligent Multimedia (CENIM)*. 2022, pp. 390–395. DOI: 10.1109/CENIM56801.2022.10037564.

[59] Qingchen Zhang et al. "A survey on deep learning for big data". In: *Information Fusion* 42 (July 2018), pp. 146–157. ISSN: 1566-2535. DOI: 10.1016/J.INFFUS.2017.10.006. URL: https://www.sciencedirect.com/science/article/pii/S1566253517305328#bib0051.

[60] Sepp Hochreiter and Jürgen Schmidhuber. "Long Short-Term Memory". In: *Neural Computation* 9 (8 Nov. 1997), pp. 1735–1780. ISSN: 08997667. DOI: 10.1162/NECO.1997.9.8.1735.

[61] Ashish Vaswani et al. "Attention is All you Need". In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon et al. Vol. 30. Curran Associates, Inc., 2017. URL: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

[62] Jacob Devlin et al. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference* 1 (Oct. 2018), pp. 4171–4186. URL: https://arxiv.org/abs/1810.04805v2.

[63] Ingrid von Glehn, James S. Spencer, and David Pfau. *A Self-Attention Ansatz for Ab-initio Quantum Chemistry*. 2023. arXiv: 2211.13672 [physics.chem-ph]. URL: https://arxiv.org/abs/2211.13672.

[64] Zonghan Wu et al. "A Comprehensive Survey on Graph Neural Networks". In: *IEEE Transactions on Neural Networks and Learning Systems* 32.1 (Jan. 2021), 4–24. ISSN: 2162-2388. DOI: 10.1109/tnnls.2020.2978386. URL: http://dx.doi.org/10.1109/TNNLS.2020.2978386.

[65] Esther Heid et al. "Chemprop: A Machine Learning Package for Chemical Property Prediction". In: *Journal of Chemical Information and Modeling* 64.1 (2024), pp. 9–17. DOI: 10.1021/acs.jcim.3c01250.

[66] Agnieszka Wojtuch et al. "Extended study on atomic featurization in graph neural networks for molecular property prediction". In: *Journal of Cheminformatics* 15 (1 Dec. 2023), pp. 1–27. ISSN: 17582946. DOI: 10.1186/S13321-023-00751-7. URL: https://jcheminf.biomedcentral.com/articles/10.1186/s13321-023-00751-7.

[67] Franco Scarselli et al. "The graph neural network model". In: *IEEE Transactions on Neural Networks* 20 (1 Jan. 2009), pp. 61–80. ISSN: 10459227. DOI: 10.1109/TNN.2008.2005605.

[68] Sebastian Ruder. "An overview of gradient descent optimization algorithms". In: (Sept. 2016). URL: https://arxiv.org/abs/1609.04747v2.

[69] Diederik P. Kingma and Jimmy Lei Ba. "Adam: A Method for Stochastic Optimization". In: *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings* (Dec. 2014). URL: https://arxiv.org/abs/1412.6980v9.

[70] MATLAB Simulink. *Overfitting - MATLAB & Simulink*. URL: https://www.mathworks.com/discovery/overfitting.html. (Accessed May 2025).

[71] Andrew Y. Ng. "Feature selection, L1 vs. L2 regularization, and rotational invariance". In: *Proceedings, Twenty-First International Conference on Machine Learning, ICML 2004* (2004), pp. 615–622. DOI: 10.1145/1015330.1015435. URL: https://dl.acm.org/doi/10.1145/1015330.1015435.

[72] Ryan Holbrook and Alexis Cook. *Overfitting and Underfitting*. URL: https://www.kaggle.com/code/ryanholbrook/overfitting-and-underfitting. (Accessed May 2025).

[73] Faras Jamil et al. "A deep boosted transfer learning method for wind turbine gearbox fault detection". In: *Renewable Energy* 197 (2022), pp. 331–341. ISSN: 0960-1481. DOI: https://doi.org/10.1016/j.renene.2022.07.117. URL: https://www.sciencedirect.com/science/article/pii/S096014812201134X.

[74] Seyone Chithrananda, Gabriel Grand, and Bharath Ramsundar Deepchem. "ChemBERTa: Large-Scale Self-Supervised Pretraining for Molecular Property Prediction". In: (Oct. 2020). URL: https://arxiv.org/abs/2010.09885v2.

[75] Nived Rajaraman, Jiantao Jiao, and Kannan Ramchandran. *Toward a Theory of Tokenization in LLMs*. 2025. arXiv: 2404.08335. URL: https://arxiv.org/abs/2404.08335.

[76] Umit V. Ucak, Islambek Ashyrmamatov, and Juyong Lee. "Improving the quality of chemical language model outcomes with atom-in-SMILES tokenization". In: *Journal of Cheminformatics* 15 (1 Dec. 2023), pp. 1–13. ISSN: 17582946. DOI: 10.1186/S13321-023-00725-9/TABLES/2. URL: https://jcheminf.biomedcentral.com/articles/10.1186/s13321-023-00725-9http://creativecommons.org/publicdomain/zero/1.0/.

[77] Miguelangel Leon et al. "Comparing SMILES and SELFIES tokenization for enhanced chemical language modeling". In: *Scientific Reports* 14 (1 Dec. 2024), pp. 1–13. ISSN: 20452322. DOI: 10.1038/S41598-024-76440-8;SUBJMETA=638,639,705;KWRD=CHEMISTRY,MATHEMATICS+AND+COMPUTING. URL: https://www.nature.com/articles/s41598-024-76440-8.

[78] Zikai Xie et al. "Fine-tuning GPT-3 for machine learning electronic and functional properties of organic molecules". In: *Chemical Science* 15 (2 Jan. 2024), pp. 500–510. ISSN: 2041-6539. DOI: 10.1039/D3SC04610A. URL: https://pubs.rsc.org/en/content/articlehtml/2024/sc/d3sc04610a.

[79] John A. Keith et al. "Combining Machine Learning and Computational Chemistry for Predictive Insights into Chemical Systems". In: *Chemical Reviews* 121 (16 Aug. 2021), pp. 9816–9872. ISSN: 15206890. DOI: 10.1021/acs.chemrev.1c00107. URL: https://pubs.acs.org/doi/full/10.1021/acs.chemrev.1c00107.

[80] Nicholas E. Jackson et al. "Introduction to Machine Learning for Molecular Simulation". In: *Journal of Chemical Theory and Computation* 19 (14 July 2023), pp. 4335–4337. ISSN: 15499626. DOI: 10.1021/ACS.JCTC.3C00735. URL: https://pubs.acs.org/doi/full/10.1021/acs.jctc.3c00735.

[81] Jan G. Rittig et al. "Graph neural networks for the prediction of molecular structure-property relationships". In: *Machine Learning and Hybrid Modelling for Reaction Engineering* (July 2022), pp. 159–181. DOI: 10.1039/BK9781837670178-00159. URL: http://arxiv.org/abs/2208.04852http://dx.doi.org/10.1039/BK9781837670178-00159.

[82] Frank Noé et al. "Machine learning for molecular simulation". In: *Annual Review of Physical Chemistry* 71 (Volume 71, 2020 Apr. 2020), pp. 361–390. ISSN: 15451593. DOI: 10.1146/annurev-physchem-042018-052331. URL: https://www.annualreviews.org/content/journals/10.1146/annurev-physchem-042018-052331.

[83] Masashi Tsubaki and Teruyasu Mizoguchi. "On the equivalence of molecular graph convolution and molecular wave function with poor basis set". In: *Advances in Neural Information Processing Systems* 2020-December (Nov. 2020). ISSN: 10495258. URL: https://arxiv.org/abs/2011.07929v1.

[84] Sarfaraz K. Niazi and Zamara Mariam. "Recent Advances in Machine-Learning-Based Chemoinformatics: A Comprehensive Review". In: *International journal of molecular sciences* 24 (14 July 2023). ISSN: 1422-0067. DOI: 10.3390/IJMS241411488. URL: https://pubmed.ncbi.nlm.nih.gov/37511247/.

[85] The Royal Swedish Academy of Sciences. *Press release: The Nobel Prize in Chemistry 2024 - NobelPrize.org*. Sept. 2024. URL: https://www.nobelprize.org/prizes/chemistry/2024/press-release/.

[86] Shafidah Shafian et al. "Development of Organic Semiconductor Materials for Organic Solar Cells via the Integration of Computational Quantum Chemistry and AI-Powered Machine Learning". In: *ACS Applied Energy Materials* (2025). ISSN: 25740962. DOI: 10.1021/ACSAEM.4C02937. URL: https://pubs.acs.org/doi/abs/10.1021/acsaem.4c02937.

[87] Ishan Amin, Sanjeev Raja, and Aditi S Krishnapriyan. "Towards Fast, Specialized Machine Learning Force Fields: Distilling Foundation Models via Energy Hessians". In: (Jan. 2025). DOI: 10.48550/arXiv.2501.09009. URL: https://arxiv.org/abs/2501.09009v1.

[88] Jiafeng Chen et al. "Insights into the Atomic Mechanism of Lithium-Ion Diffusion in Li6PS5Cl via a Machine Learning Potential". In: *Chemistry of Materials* (2025). ISSN: 15205002. DOI: 10.1021/ACS.CHEMMATER.4C01152. URL: https://pubs.acs.org/doi/abs/10.1021/acs.chemmater.4c01152.

[89] Kin Gomez et al. "Microfluidic and Computational Tools for Neurodegeneration Studies". In: *Annual Review of Chemical and Biomolecular Engineering* (Jan. 2025). ISSN: 1947-5438. DOI: 10.1146/ANNUREV-CHEMBIOENG-082223-054547. URL: https://www.annualreviews.org/content/journals/10.1146/annurev-chembioeng-082223-054547.

[90] Teuku Rizky Noviandy et al. "Interpretable machine learning approach to predict Hepatitis C virus NS5B inhibitor activity using voting-based LightGBM and SHAP". In: *Intelligent Systems with Applications* 25 (Mar. 2025), p. 200481. ISSN: 2667-3053. DOI: 10.1016/J.ISWA.2025.200481.

[91] Li Tian et al. "Transfer learning applied in predicting small molecule bioactivity". In: *bioRxiv* (Jan. 2025), p. 2025.01.09.632140. DOI: 10.1101/2025.01.09.632140. URL: https://www.biorxiv.org/content/10.1101/2025.01.09.632140v1https://www.biorxiv.org/content/10.1101/2025.01.09.632140v1.abstract.

[92] Powsali Ghosh, Ashok Kumar, and Sushil Kumar Singh. "COX-2 Inhibitor Prediction With KNIME: A Codeless Automated Machine Learning-Based Virtual Screening Workflow". In: *Journal of Computational Chemistry* 46 (2 Jan. 2025), e70030. ISSN: 1096-987X. DOI: 10.1002/JCC.70030. URL: https://onlinelibrary.wiley.com/doi/full/10.1002/jcc.70030https:

//onlinelibrary.wiley.com/doi/abs/10.1002/jcc.70030https://onlinelibrary.wiley.com/doi/10.1002/jcc.70030.

[93] Dong Wang et al. "ADMET evaluation in drug discovery: 21. Application and industrial validation of machine learning algorithms for Caco-2 permeability prediction". In: *Journal of Cheminformatics 2025 17:1* 17 (1 Jan. 2025), pp. 1–14. ISSN: 1758-2946. DOI: 10.1186/S13321-025-00947-Z. URL: https://link.springer.com/articles/10.1186/s13321-025-00947-zhttps://link.springer.com/article/10.1186/s13321-025-00947-z.

[94] Rıza Özçelik and Francesca Grisoni. "A hitchhiker's guide to deep chemical language processing for bioactivity prediction". In: *Digital Discovery* (2025), pp. –. DOI: 10.1039/D4DD00311J. URL: http://dx.doi.org/10.1039/D4DD00311J.

[95] Sankalp Jain et al. "Introduction to the Research Topic: Advancements in Predictive Toxicology Methods". In: *Frontiers in Pharmacology* 16 (2025), p. 1556352. ISSN: 1663-9812. DOI: 10.3389/FPHAR.2025.1556352.

[96] Teng Jiek See et al. "Graph Neural Network-Based Molecular Property Prediction with Patch Aggregation". In: *Journal of Chemical Theory and Computation* (2024). ISSN: 15499626. DOI: 10.1021/acs.jctc.4c00798. URL: https://pubs.acs.org/doi/abs/10.1021/acs.jctc.4c00798.

[97] Zhenxing Wu et al. "Chemistry-intuitive explanation of graph neural networks for molecular property prediction with substructure masking". In: *Nature Communications 2023 14:1* 14 (1 May 2023), pp. 1–15. ISSN: 2041-1723. DOI: 10.1038/s41467-023-38192-3. URL: https://www.nature.com/articles/s41467-023-38192-3.

[98] Han Altae-Tran et al. "Low Data Drug Discovery with One-Shot Learning". In: *ACS Central Science* 3.4 (2017), pp. 283–293. DOI: 10.1021/acscentsci.6b00367. URL: https://doi.org/10.1021/acscentsci.6b00367.

[99] Justin Gilmer et al. "Neural Message Passing for Quantum Chemistry". In: *Proceedings of the 34th International Conference on Machine Learning*. Ed. by Doina Precup and Yee Whye Teh. Vol. 70. Proceedings of Machine Learning Research. PMLR, 2017, pp. 1263–1272. URL: https://proceedings.mlr.press/v70/gilmer17a.html.

[100] Walid Ahmad et al. "ChemBERTa-2: Towards Chemical Foundation Models". In: (Sept. 2022). URL: https://arxiv.org/abs/2209.01712v1.

[101] M.J. Frisch et al. "Gaussian 09 Revision A.1. Gaussian Inc". In: (Jan. 2009).

[102] Christopher J Cramer. *Essentials of Computational Chemistry Theories and Models*. Vol. 42. Wiley, 2004. DOI: 10.1021/ci010445m.

[103] Mu-Hyun Baik and Richard A. Friesner. "Computing Redox Potentials in Solution: Density Functional Theory as A Tool for Rational Design of Redox Agents". In: *The Journal of Physical Chemistry A* 106.32 (2002), pp. 7407–7412. DOI: 10.1021/jp025853n. eprint: https://doi.org/10.1021/jp025853n. URL: https://doi.org/10.1021/jp025853n.

[104] Andrew F. Siegel. "Chapter 15 - ANOVA: Testing for Differences among Many Samples, and Much More". In: *Practical Business Statistics (Sixth Edition)*. Ed. by Andrew F. Siegel. Sixth Edition. Boston: Academic Press, 2012, pp. 467–490. ISBN: 978-0-12-385208-3. DOI: https://doi.org/10.1016/B978-0-12-385208-3.00015-8. URL: https://www.sciencedirect.com/science/article/pii/B9780123852083000158.

[105] Sebastian Birkedal Kristensen. *Electrochemistry and Production Optimization of Naturally Occuring Quinones for Flow Battery Technology*. English. PhD supervisor: Associate Professor. Jens Muff, Aalborg University PhD co-supervisor: Associate Professor. Jens Laurids Sørensen, Aalborg University. 2021. DOI: 10.54337/aau456471612.

Appendix A

Results of model trials

Table A.1 Results of 2D-GNN Configurations

Configuration no.	Mean Validation R^2	Mean Test R^2
1	0.8373	0.7203
2	0.8400	0.7421
3	0.8177	0.6997
4	0.8062	0.7336
5	0.8248	0.7004
6	0.8650	0.6689
7	0.8361	0.6767
8	0.8612	0.6729
9	0.8082	0.7237
10	0.8355	0.7154
11	0.8198	0.7368
12	0.8335	0.7249
13	0.8249	0.7118
14	0.8661	0.6708
15	0.8378	0.6675
16	0.8559	0.7090

Table A.2 Results of 3D-GNN Configurations

Configuration no.	Mean Validation R^2	Mean Test R^2
1	0.7736	0.6796
2	0.7445	0.6462
3	0.7776	0.6785
4	0.7462	0.6641
5	0.7677	0.6921
6	0.8036	0.7001
7	0.7815	0.6861
8	0.7618	0.7089
9	0.7767	0.6605
10	0.7734	0.7081
11	0.7697	0.6966
12	0.7334	0.6585
13	0.7621	0.6844
14	0.7028	0.6150
15	0.8089	0.7271
16	0.7529	0.6831

Table A.3 Results of ChemBERTa with SMILES input and configuration 19

Model	Validation RMSE	Validation R^2	Test RMSE	Test R^2
ChemBERTa	0.2182	0.5546	0.2132	0.6583
ChemBERTa	0.2006	0.6219	0.2255	0.6185
ChemBERTa	0.201	0.6223	0.2056	0.6817
ChemBERTa	0.1913	0.6591	0.253	0.5204
ChemBERTa	0.2008	0.623	0.2055	0.6823
ChemBERTa	0.2076	0.5966	0.2215	0.6326
ChemBERTa	0.1989	0.628	0.2448	0.5486
ChemBERTa	0.193	0.6524	0.2181	0.6432
ChemBERTa	0.2042	0.6088	0.2388	0.5707
ChemBERTa	0.1996	0.627	0.2108	0.6652
ChemBERTa	0.2017	0.6211	0.2226	0.6275
ChemBERTa	0.187	0.674	0.2065	0.679
ChemBERTa	0.196	0.6395	0.2206	0.6336
ChemBERTa	0.1895	0.662	0.2277	0.6092
ChemBERTa	0.1959	0.6395	0.217	0.6458
ChemBERTa	0.1933	0.6495	0.2188	0.6404
ChemBERTa	0.1993	0.6282	0.2082	0.6751
ChemBERTa	0.2212	0.542	0.2149	0.653
ChemBERTa	0.2218	0.5384	0.2125	0.6615
ChemBERTa	0.1895	0.6643	0.2047	0.6849
ChemBERTa	0.1921	0.6558	0.2093	0.674
ChemBERTa	0.2035	0.6137	0.2269	0.6159
ChemBERTa	0.1987	0.6295	0.2491	0.5345
ChemBERTa	0.1884	0.667	0.2223	0.6281
ChemBERTa	0.201	0.6223	0.2055	0.6822
ChemBERTa	0.2102	0.5868	0.2113	0.6665
ChemBERTa	0.2001	0.6241	0.2032	0.69
ChemBERTa	0.2029	0.6157	0.2216	0.6312
ChemBERTa	0.1952	0.645	0.224	0.6235
ChemBERTa	0.1856	0.6755	0.2177	0.6443

Table A.4 Results of PubChem with SMILES Canonical input and configuration 16

Model	Validation RMSE	Validation R^2	Test RMSE	Test R^2
PubChem	0.2351	0.4834	0.197	0.7071
PubChem	0.2174	0.5557	0.1878	0.7345
PubChem	0.2167	0.5603	0.1857	0.7401
PubChem	0.2133	0.5755	0.1888	0.7319
PubChem	0.2046	0.6101	0.1906	0.7261
PubChem	0.2247	0.5304	0.19	0.7288
PubChem	0.2115	0.5807	0.188	0.734
PubChem	0.2052	0.6061	0.1783	0.7602
PubChem	0.212	0.5791	0.1803	0.7554
PubChem	0.2061	0.6031	0.1769	0.7645
PubChem	0.196	0.6405	0.1975	0.7059
PubChem	0.2111	0.5834	0.1972	0.7067
PubChem	0.2014	0.6186	0.181	0.7531
PubChem	0.2159	0.5643	0.1867	0.7374
PubChem	0.2266	0.5185	0.1975	0.7057
PubChem	0.2096	0.5895	0.1978	0.7046
PubChem	0.2167	0.5622	0.1983	0.7036
PubChem	0.2006	0.6256	0.196	0.7108
PubChem	0.2107	0.5849	0.1741	0.7716
PubChem	0.2081	0.5973	0.1878	0.7347
PubChem	0.2213	0.5419	0.1782	0.7607
PubChem	0.2043	0.6091	0.1911	0.7247
PubChem	0.1966	0.6384	0.1838	0.7455
PubChem	0.2116	0.5796	0.1821	0.7499
PubChem	0.2173	0.5587	0.1838	0.7456
PubChem	0.1967	0.6374	0.172	0.7774
PubChem	0.2178	0.5568	0.1788	0.7594
PubChem	0.1969	0.6364	0.2005	0.6972
PubChem	0.2118	0.5787	0.2006	0.6964
PubChem	0.2273	0.5161	0.1885	0.733

Table A.5 Results of 2D-GNN with configuration 2

Model	Validation RMSE	Validation R^2	Test RMSE	Test R^2
2D-GNN	0.1398	0.8464	0.1992	0.7346
2D-GNN	0.1408	0.8443	0.2092	0.7074
2D-GNN	0.1554	0.8103	0.1956	0.7441
2D-GNN	0.1382	0.85	0.209	0.708
2D-GNN	0.1501	0.823	0.2038	0.7222
2D-GNN	0.135	0.8569	0.199	0.7353
2D-GNN	0.1393	0.8475	0.226	0.6586
2D-GNN	0.1395	0.847	0.2093	0.7071
2D-GNN	0.1481	0.8277	0.1889	0.7613
2D-GNN	0.1379	0.8505	0.2208	0.6739
2D-GNN	0.144	0.837	0.1942	0.7477
2D-GNN	0.1336	0.8597	0.223	0.6674
2D-GNN	0.1365	0.8536	0.1919	0.7539
2D-GNN	0.1433	0.8386	0.2109	0.7027
2D-GNN	0.1552	0.8108	0.2123	0.6986
2D-GNN	0.1899	0.7168	0.1839	0.7739
2D-GNN	0.1418	0.842	0.1994	0.7341
2D-GNN	0.137	0.8526	0.1962	0.7427
2D-GNN	0.1598	0.7995	0.1955	0.7444
2D-GNN	0.1421	0.8413	0.2035	0.7231
2D-GNN	0.1356	0.8556	0.2204	0.6751
2D-GNN	0.1402	0.8457	0.2125	0.6981
2D-GNN	0.148	0.8278	0.1976	0.739
2D-GNN	0.1561	0.8085	0.2261	0.658
2D-GNN	0.1506	0.8217	0.202	0.7271
2D-GNN	0.1429	0.8396	0.199	0.7353
2D-GNN	0.1306	0.8661	0.1965	0.7418
2D-GNN	0.1607	0.7971	0.1922	0.7529
2D-GNN	0.1428	0.8398	0.1932	0.7505
2D-GNN	0.1291	0.8692	0.2051	0.7186

Table A.6 Results of 3D-GNN with configuration 15

Model	Validation RMSE	Validation R^2	Test RMSE	Test R^2
3D-GNN	0.1598	0.8055	0.2004	0.7321
3D-GNN	0.1924	0.7181	0.2173	0.685
3D-GNN	0.174	0.7693	0.2138	0.6949
3D-GNN	0.1766	0.7624	0.2286	0.6511
3D-GNN	0.1718	0.7753	0.2069	0.7143
3D-GNN	0.1431	0.8439	0.2012	0.7299
3D-GNN	0.1725	0.7734	0.2091	0.7082
3D-GNN	0.2009	0.6926	0.2473	0.592
3D-GNN	0.1515	0.8252	0.2243	0.6642
3D-GNN	0.188	0.7308	0.2024	0.7267
3D-GNN	0.1624	0.799	0.1989	0.7359
3D-GNN	0.1821	0.7474	0.2219	0.6714
3D-GNN	0.18	0.7533	0.2103	0.7049
3D-GNN	0.1797	0.7539	0.2197	0.678
3D-GNN	0.1709	0.7775	0.2192	0.6794
3D-GNN	0.1641	0.7948	0.2237	0.6661
3D-GNN	0.1558	0.8152	0.1957	0.7445
3D-GNN	0.1477	0.8339	0.1961	0.7433
3D-GNN	0.162	0.8001	0.215	0.6915
3D-GNN	0.1705	0.7786	0.2089	0.7087
3D-GNN	0.1533	0.8211	0.2156	0.6898
3D-GNN	0.1887	0.7287	0.2043	0.7213
3D-GNN	0.1533	0.821	0.2021	0.7274
3D-GNN	0.1605	0.8038	0.2089	0.7088
3D-GNN	0.1429	0.8445	0.2068	0.7145
3D-GNN	0.1607	0.8033	0.2038	0.7229
3D-GNN	0.1587	0.8083	0.2007	0.7311
3D-GNN	0.1739	0.7696	0.2016	0.7288
3D-GNN	0.1655	0.7912	0.1926	0.7524
3D-GNN	0.1872	0.733	0.2124	0.6989