Facultade de Química

Grao en Química

Traballo de Fin de Grao

Mellora de métodos computacionais semiempíricos para a predición de reactividade química mediante técnicas de "machine learning"

Autor:

Xabier García Andrade

Titor:

Emilio Martínez Nuñez

Departamento de Química Física

Cotitor:

Pablo García Tahoces

Departamento de Electrónica e Computación

Xullo de 2021. Curso 2020/2021

Traballo de Fin de Grao

- **Rama do coñecemento:** Química Física
- **Departamento, centro, institución ou empresa:** Facultade de Química (USC)
- **Titor:** Dr. Emilio Martínez Nuñez
- **Cotitor:** Dr. Pablo García Tahoces

Autorización dos titores:

Dr. Emilio Martínez Nuñez, Profesor Titular do Departamento de Química Física da Universidade de Santiago de Compostela. Dr. Pablo García Tahoces, Profesor Titular do Departamento de Electrónica e Computación da Universidade de Santiago de Compostela.

Certifican: que a presente memoria adxunta, titulada "*Mellora de métodos computacionais semiempíricos para a predición de reactividade química mediante técnicas de 'machine learning'*", que presenta Xabier García Andrade, foi realizada baixo a súa dirección na Facultade de Química da Universidade de Santiago de Compostela.

Considerando que a nomeada memoria constitúe o seu traballo de fin de grao, autorizan a súa presentación na Universidade de Santiago de Compostela.

Para que así conste, asinan o presente informe en Santiago de Compostela a día 3 de Xullo de 2021.

MARTINEZ NUÑEZ EMILIO - 33306236M
Firmado digitalmente por MARTINEZ NUÑEZ EMILIO - 33306236M
Nombre de reconocimiento (DN): c=ES, serialNumber=IDCES-33306236M, givenName=EMILIO, sn=MARTINEZ NUÑEZ, cn=MARTINEZ NUÑEZ EMILIO - 33306236M
Fecha: 2021.07.02 16:47:00 +02'00'

GARCIA TAHOCES PABLO - 10066395P
Firmado digitalmente por GARCIA TAHOCES PABLO - 10066395P
Nombre de reconocimiento (DN): c=ES, serialNumber=IDCES-10066395P, givenName=PABLO, sn=GARCIA TAHOCES, cn=GARCIA TAHOCES PABLO - 10066395P
Fecha: 2021.07.05 10:14:33 +02'00'

Abstract

O machine learning e a intelixencia artificial estanse convertindo en técnicas cada vez máis presentes na investigación en química computacional. A medida que temos acceso a unha maior cantidade datos de química cuántica, aumentan as posibilidades de uso de algoritmos intelixentes para a exploración do espazo químico.

Por outra parte, aínda non dispoñemos de métodos eficientes para a predición de propiedades de reacción cuantitativas. Entre estas propiedades atópase a enerxía de activación, cuxa predición de forma precisa proporcionaría un método para o descubrimento de novos mecanismos de reacción e forneceríanos control sobre a cinética das reaccións.

O presente traballo intenta buscar un algoritmo baseado en machine learning para a predición de enerxías de activación. O noso modelo depende dun cálculo a nivel semiempírico (PM7) e proporciona unha predición a nivel DFT mediante machine learning. Con este procedemento, conseguimos precisión química cun custo computacional reducido. Ademais de obter un rendemento equiparable ao estado da arte, esta alternativa contribúe con descritores personalizados, que poden ser incorporados en novos procedementos de minaría de datos na química. Por último, tamén proporciona unha interpretación do modelo dende a perspectiva da intuición química.

Resumen

El machine learning y la inteligencia artificial se están convirtiendo en técnicas cada vez más presentes en la investigación en química computacional. A medida que tenemos acceso a una mayor cantidad de datos de química cuántica, aumentan las posiblidades de uso de algoritmos inteligentes para la exploración del espacio químico.

Por otra parte, todavía no disponemos de métodos eficientes para la predicción de propiedades de reacción cuantitativas. Entre estas propiedades se

encuentra la energía de activación, cuya predicción de forma precisa proporcionaría un método para el descubrimiento de nuevos mecanismos de reacción y nos facilitaría el control sobre la cinética de las reacciones.

El presente trabajo intenta buscar un algoritmo basado en machine learning para la predicción de energías de activación. Nuestro modelo depende de un cálculo a nivel semiempírico (PM7) y proporciona una predicción a nivel DFT mediante machine learning. Con este procedimiento, conseguimos precisión química con un coste computacional reducido. Además de obtener un rendimiento equiparable al estado del arte, esta alternativa contribuye con descriptores personalizados, que pueden ser incorporados en procedimientos de minería de datos en la química. Por último, también proporciona una interpretación del modelo desde una perspectiva de la intuición química.

Abstract

Machine learning and artificial intelligence are becoming ubiquitous techniques in computational chemistry research. With quantum-chemical data becoming increasingly available, intelligent algorithms are taking the upper hand in the exploration of the chemical space.

On the other hand, we still lack efficient algorithms when it comes to predicting quantitative reaction properties. Within this properties, accurately predicting activation energies would enable rapid discovery of new reaction mechanisms and would grant control over chemical kinetics.

The present work intends to seek for a machine learning-based algorithm to predict activation energies. Our model relies on a semiempirical calculation (PM7 level of theory) and resorts to machine learning to DFT accuracy. With this procedure, we can obtain chemical accuracy while limiting the computational expenditure. In addition to achieving state of the art performance, this approach contribute with innovative custom descriptors that can be harnessed in data mining techniques in the chemical domain and allow interpretability from the perspective of chemical intuition.

Contents

1 Introduction

Accurately predicting reactivity is certainly one of the most fundamental problems that can be faced in chemistry. Unravelling how any chemical system evolves and transforms with time opens many possibilities to develop new synthetic routes or enhancing current ones. A sufficiently trained chemist can definitely tell how reactants behave and what product they would yield. Nevertheless, sometimes the system is not tractable by human thinking. This limitation has lead to the development of new computational methodologies and algorithms that can emulate human-like performance, whose success is not only limited by computational power, but also by intrinsic algorithmic design flaws.

From a computational perspective, in order to predict reactivity, the most crucial parameter that must be determined in a chemical reaction is the activation energy. Calculating reaction parameters for simple enough systems has been possible since the development of Hartree Fock (HF) methods [1]. Despite HF scheme success, it fails to produce useful calculations as the system becomes larger. In this situation, semi-empirical methods emerged as a tool to handle large molecules by means of integrating empirical parameters and approximations together with the HF formalism. Semi-empirical methods [2] enabled large systems calculations, but their results differed from experimental data if the system did not resemble those used for the parametrization. In order to achieve higher accuracy, computational chemists had to resort to density functional theory (DFT), which is another *ab initio* method [3]. DFT saw unprecedented success, attaining quantum chemical accuracy in many different sets of molecules. Then again, the limitation of DFT calculations is the size of the system. There are other methods that provide even more accurate calculations, CCSD(T) (coupled cluster calculations), considered by many quantum chemists as the gold standard [4]. Coupled cluster methods are only restricted to very small systems becoming extremely expensive as the system becomes larger. For the purpose of the present work, the discussion will remain at the DFT level.

At this point, we can agree that our main target would be to find a way to surpass DFT limitations while still obtaining the same rigorous results. There have been many attempts to formulate a new theoretical framework that can utterly substitute DFT, but they stumble in the same pitfall which is not being scalable. This situation lead to stagnation within the theoretical chemistry field, until the rise of machine learning in the physical sciences. Machine learning has been shifting the scientific paradigm for the last decade, from purely theoretical modelling towards data-driven discoveries.

The applications in chemistry cover a broad range, from complex problems in retrosynthesis or reaction optimization to drug design [5] in the context of organic chemistry. Analytical chemistry is probably the subfield in chemistry where machine learning was first used, since the problems in analytical chemistry are typically of statistical nature, such as evaluating food quality [6]. In inorganic chemistry, recent developments of deep learning frameworks allow highly accurate prediction for inorganic materials [7].

Regarding predictions of reaction parameters, such as activation energy, machine learning based methodologies have seen significant growth as well. In particular, deep learning algorithms (a subfield within machine learning) dominate this area. Marquetand et al. [8] work was pioneering in the use of machine learning. They proposed a high-dimensional neural network to fit electronic structure data and construct potential energy surfaces for organic reactions. Allison et al. [9] described the application of an artificial neural network to the prediction of reaction rate constants for reactions involving OH radicals.

Closer related to the scope of our work, Green et al.(2020) [10] proposed a deep learning approach to calculate activation energies and other quantitative data from reaction data. They used a novel approach that relies on a directed message passing neural network that encodes molecules as graphs and predicts the activation energy. Nevertheless, this model does not utilize information available from the transition

states and only relies on information from the atomic connectivity from reactants and products.

After a thorough revision of the state of the art, it is clear that ML has taken over the computational chemistry space when it comes to quantitative prediction of reaction parameters. With the present work, we intend to expand on th type of methods, overcoming the shortcomings and flaws on common ML approaches by combining domain expertise as chemists and the most sophisticated machine learning algorithms.

2 Objectives and Work Plan

The present work seeks to expand current methods on reaction parameters prediction. In particular, our goal is to develop a deep learning-based model that can accurately predict activation energies while remaining computationally affordable. In the previous chapter we mentioned how most approaches sought to surpass electronic structure calculations or molecular dynamics and provide a machine learning model that can directly predict the desired activation energy.

Nevertheless, our approach is focused on leveraging a cheap calculation provided by semiempirical methods into higher accuracy (DFT level of theory). More precisely, the semiempirical level employed will be PM7 provided by MOPAC and the DFT functional with the desired precision is ωB97X-D3. This work can also be understood as a correction for the semiempirical level of theory or even as a reparametrization. In summary, we want to achieve higher accuracies without jeopardizing computational efficiency.

In addition to providing this correction, the interpretation of the model output from a chemical perspective will grant unique insights into how semiempirical and DFT calculation differ, which can provide clues on how to offset parametrizations for theoreticians working on these semiempirical methods.

With respect to the work plan, in order to train a model first it is necessary to obtain a curated dataset containing enough samples. For this work, the database produced by Grambow et al. (2020) [10] was used. They generated quantitative data for almost 12000 organic reactions at the ωB97X-D3/def2-TZVP quantum chemistry level. After scraping the data, sempiempirical calculations must be performed using geometries for the different chemical species specified in the database. Since there is the possibility that a fraction of the calculations cannot be achieved at the semiempirical level of theory, it was necessary to discard some of the data entries.

Having obtained data at the semiempirical level, it was then possible to obtain

the difference in activation energy between both methods, which is our target data. Then, since the ML model needs input data (the precise term is descriptors, which will be introduced in further detail in the next chapter) to produce a prediction, we had to extract this information from the database. In our case, these type of data involved finding new ways to encode the information encapsulated by a chemical reaction. It can range from electronic structure calculations applied to the reactants and products to topological and geometrical indices related to the transition state. That is, we eventually wanted to translate the language used by chemists into chunks that could be interpreted by a computer.

Upon completion of the dataset, it must be split into training, validation and test in order to perform experiments. The experiments involved using different ML algorithms to select the ones that produced better predictions. Then, the model parameters were optimized to enhance the outputs and compare its performance with results from state of the art models. The last step involved using different mathematical procedures that enable the interpretation of the model from a chemical perspective.

3 Theoretical Framework

3.1 Artificial Intelligence and Machine Learning

While the emergence of Artificial Intelligence (AI) might evoke a sensation of cutting-edge research and recent developments, it has been subject of active research for the past centuries. A more rigorous approach can be dated back to the early 20th century, as the philosopher Bertrand Rusell revolutionized formal logic in his work *Principia Mathematica* [11]. From that moment, researchers began to investigate how to formalize the mathematical reasoning previously depicted by philosophers. The next milestone arrived with Alan Turing's work on machine intelligence, defining a formalized version of a general purpose computer. Because of this research, Turing is usually referred to as the foundational father of AI. Skipping a few years of acute developments on the field, it is worth emphasizing the results from Frank Rosenblatt on his book *The Perceptron (1957)* [12]. He proposed the first implementation of an artificial neural network (ANN).

Fast forward until the 21st century, to the era of machine and deep learning. Faster and accessible computers enabled the usability of artificial intelligence techniques across different fields, which harnessed the power of data driven algorithms into both academic and industrial problems. Thus, thee current trend about using machine learning in every problem is justified by a series of achievements on the last decade.

This historical discussion grants an unique perspective on how and why AI is so widely used to solve research questions. Sometimes the terms AI and machine learning/deep learning are used interchangeably, but our next step will be to grant a concrete definition of each term and depict their boundaries. The relationship among these terms is usually visualized by means of the Venn diagram on Figure 1.

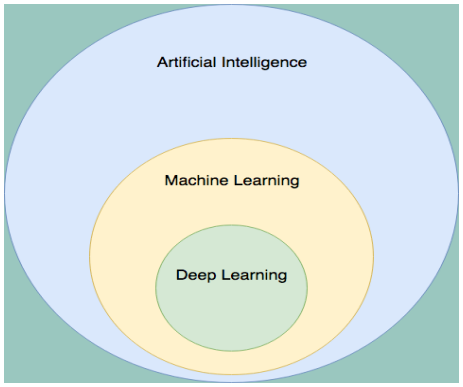


Figure 1: Venn Diagram on AI/ML/DL

It is important to bear in mind that machine learning is a subset within AI. In particular, we can define machine learning as the subset of AI which relies on data in order to learn how to perform a certain task. Then, deep learning is a subset of machine learning which deals with a particular type of algorithms.

Every deep learning algorithm can be classified as a machine learning algorithm, but most machine learning does not resort to deep learning algorithms. The difference between these classes can be more subtle, but it will become clear as we dive deeper on more details on how the algorithms work.

First, we will explain how machine learning works by describing the type of problems that it solves. There are two major distinctions on machine learning: supervised learning and unsupervised learning. For the scope of this work, we will restrict to supervised learning. Within supervised learning, there are two separate tasks that comprise every supervised problem: classification and regression.

Both classification and regression follow the same logic: we are granted a dataset containing predictor variables X or "features"[1] and a response variable Y. The main difference relies on the nature of the response variable Y. For a regression task, Y is a quantitative variable or continuous, while for classification it is qualitative or discrete. For the scope of this work, regression will be the main focus.

[1] In the applications of ML in chemistry, features are usually called descriptors.

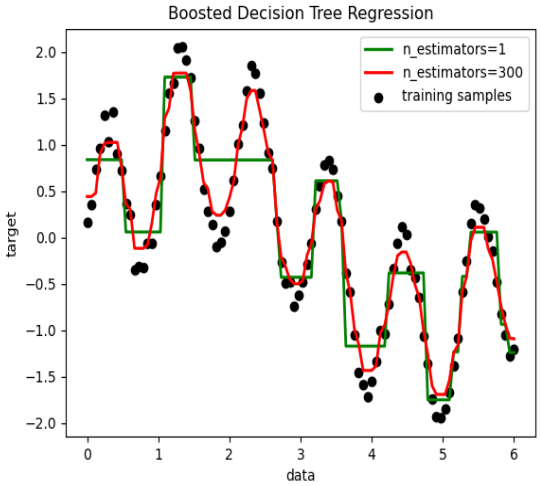


Figure 2: Regression Example using Decision Tree [13]

As an example of how a regression task works, we will use a one dimensional example problem, where the target data Y is a function of only one feature X, displayed on Figure 2. Thus, the objective is to obtain a model such that its predictions match Y . Ultimately, the procedure is equivalent to that of linear regression, but in reality data does not always follow linear behavior (in fact with ML it is not necessary to assume a functional relationship between features and target data) and the dimensionality is higher than one.

At this point, the model can be treated as a black-box which receives an input X and produces a prediction $\hat{y}$. In order to enhance its predictions, a loss function is used, which is another key concept in machine learning. In general, loss functions receive model predictions $\hat{y}$ and ground truth values Y as arguments and produce a number that can be understood as how the predictions differ from the ground truth. There are numerous loss functions used in ML and they can be as simple as the MSE, usually used for regression problems:

$$J(Y, \hat{y}) = \frac{\sum_{i=1}^{N} (Y_i - \hat{y}_i)^2}{N} \tag{1}$$

Taking into account this interpretation of the loss function as a measure for the discrepancy between our predictions and the values present in the response variable, it is clear that our objective should be to minimize the loss function. Then, we have reduced or simplified our problem into an optimization problem. There are different

mathematical optimizers available, but the underlying principle is the same. Figure 3 illustrates this situation, where the loss function is plotted as a function of the model parameters in a 1-dimensional setting. The generalization to higher dimensions is trivial, but the simpler case is easier to visualize.

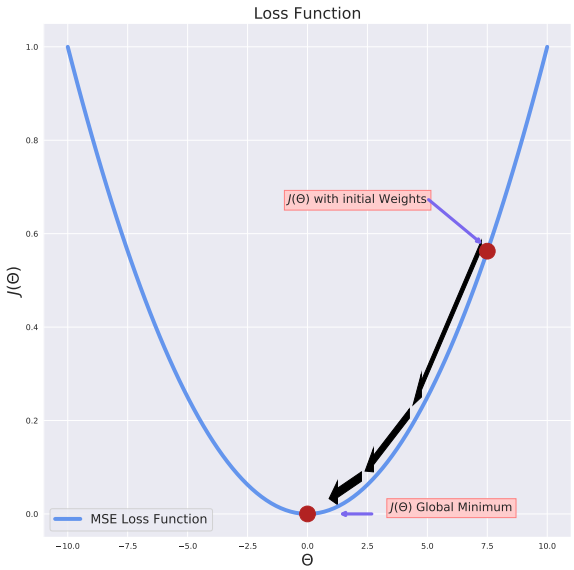


Figure 3: Loss function

Let us recapitulate the formulation of our model and how predictions are improved. First, we use our model to predict, then a loss function is calculated and model parameters are updated via an optimizer with respect to the loss function. This process is repeated iteratively until convergence of the loss function when it reaches a global minimum.

The next step will be to specify how parameters themselves are updated in order to solve the optimization problem. Minimization is attained by means of gradient descent [14] (there are numerous optimizers, but the fundamental processes are

inspired from gradient descent):

$$\theta_i = \theta_i - \alpha \frac{\partial J}{\partial \theta_i} \tag{2}$$

Where α is a parameter named learning rate. Introducing learning rate this way leads to a natural discussion about different type of parameters in machine learning models. Within a model, it is crucial to distinguish among parameters (such as weights and biases) and hyperparameters (such as learning rate, number of hidden layers or number of neurons). Parameters are those which can be trained with every iteration and hyperparameters are fixed from the beginning. They are susceptible to being optimized as well, but this process will be explained later.

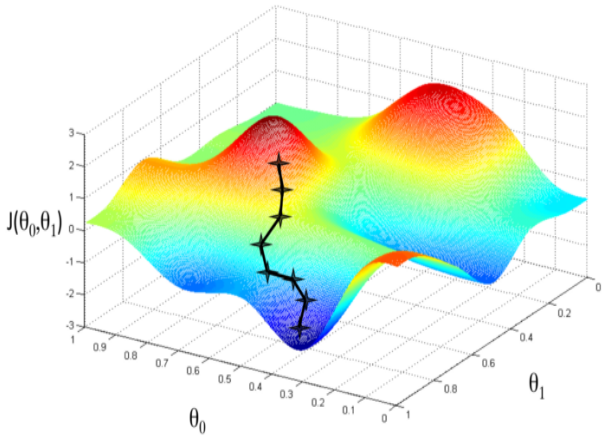


Figure 4: Gradient Descent

Figure 4 illustrates how gradient descent works in a pictorial way. It shows a 3-dimensional plot representing the cost function J in the z axis as a function of two parameters. The path drawn shows how parameters are updated in the direction of steepest descent with respect to the surface spanned by the loss function. Eventually, gradient descent stops at the global minimum. This plot serves as an example of the importance of selecting the appropriate learning rate. The significance of the learning rate is serving as a sensitivity on how much the parameters are updated in

every iteration. If the learning rate is too large, global minimum might be skipped and the optimization might land in a local minimum. If is too small, the learning optimization process can be too slow.

3.1.1 Artificial Neural Networks

Armed with the global vision on how machine learning problems work, now we should specify how parameters are updated and how predictions are made for the particular case of neural networks, even though that the same process is used for different models.

An artificial neural network is a computing system which learns patterns from data and whose functioning rationale is inspired by how biological systems process and share information.

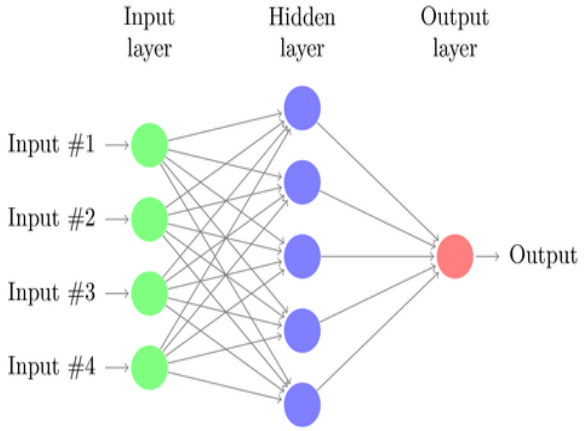


Figure 5: Artificial Neural Network representation

Every layer in a neural network is comprised of neurons, which can be understood as its building blocks. A collection of neurons comprise each of the layers shown in Figure 5. There are different types of layers, named input layer, hidden layers and output layer. Hidden layers are optional, and so are the number of hidden layers in a neural network. The purpose of each type of layer will become clear in the following paragraphs. At this point, we must focus on the building blocks. Each

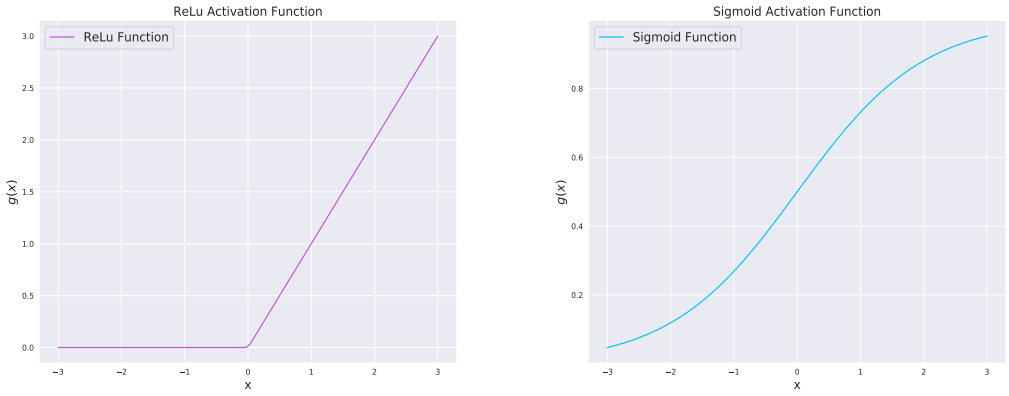
neuron contains two sets of parameters: weights (θ_i) and biases (b_i). When data is fed from the input layer, every neuron computes a linear function by using its weights and biases:

$$Z = \theta \cdot X + b \tag{3}$$

Note that if neural networks only performed linear calculations, they would be equivalent to linear regression. In order to be able to compute non linear patterns, it is necessary to introduce another function. In ML terminology, this (presumably) non linear function is often referred to as activation function. In general, it can be expressed as:

$$A = g(Z) = g(\theta \cdot X + b) \tag{4}$$

There are several activation functions, among those the most popular options are sigmoid or *ReLu* (Rectified Linear unit) functions, defined as:



a) ReLu Activation Function b) Sigmoid Activation Function

Figure 6: Activation Functions

$$g(x) = \frac{1}{1 + e^{-x}} \tag{5}$$

$$g(x) = max(0, x) \tag{6}$$

Where equations 6 and 5 refer to *ReLu* and sigmoid functions respectively. Figure 6 a) displays ReLu and b) Sigmoid activations functions. These two steps are performed sequentially, using the output from one layer as input for the next layer until output layer is reached. The output layer already produces our predictions, which will be used to calculate the loss function. Steps up to this point comprise what is usually known as feedforward step.

Next steps, which will be called backwards steps, are the reason why neural networks are so widely used. The underlying algorithm which explains how NN learn is backpropagation [15]. Backpropagation is a method to calculate partial derivatives of the cost function with respect to the weights (parameters) of the model. Keeping in mind that each neuron computes a complex function depending on the output from previous layers, it is clear that in order to calculate partial derivatives, this relationship must be considered by means of the chain rule. Thus, backpropagation becomes an efficient implementation which enables a rapid computation of gradients in parameter space.

$$\frac{\partial J}{\partial \theta^{[L]}} = \frac{1}{N} \frac{\partial J}{\partial Z^{[L]}} g(Z)^T = \frac{1}{N} \frac{\partial J}{\partial Z^{[L]}} \left(A^{[L-1]T}\right)^T \tag{7}$$

$$\frac{\partial J}{\partial b^{[L]}} = \frac{1}{N} \sum_{i}^{N} \frac{\partial J}{\partial Z^{[L](i)}} \tag{8}$$

$$\frac{\partial J}{\partial A^{[L-1]}} = \left(\theta^{[l]}\right)^T \frac{\partial J}{\partial Z^{[L]}} \tag{9}$$

Equations 7 to 9 show how backpropagation is performed, where the L index refers to the layer and the other quantities were already defined previously. This is the most general version, it must be noticed that the exact implementation will

depend on which activation functions and loss function are chosen. Nevertheless, we will not elaborate more on this topic, since the most relevant bit of information to retain and which must be emphasized is that backpropagation is just a method to compute gradients. Eventually the computation of these gradients are needed for gradient descent algorithm to perform optimization with respect to the loss function as explained in the previous section.

After revising the details and intricacies in neural networks, we will summarize how information flows within this model. First, predictor variables X are fed to the input layer, which initiates the feedforward step until it reaches the output layer, producing a prediction $\hat{y}$. Cost function is used by the neural network to learn the optimal parameters to optimize the prediction by means of backpropagation and gradient descent. The process is repeated for a number of epochs until convergence of the loss function. Equipped with the knowledge on how to train models, now we will move on to discuss how to perform machine learning experiments.

3.1.2 Experimentation in ML

Experiments in machine learning usually follow the same approach. First, the dataset is partitioned typically into train, validation and test sets by randomly selecting data entries and assigning a percentage of the dataset to each one. The train set, as the name suggests, is the one used to optimize the model parameters such that it learns how to perform the task. The validation set is used to measure how the model behaves with respect to a given metric and evaluates how the model works with unseen data. In the context of regression, the most common metric is arguably the mean absolute error (MAE , equation 10). Test set serves as an assessment on how the model is able to extrapolate to data which was not used during training nor as validation of the model performance. The difference between validation and test might seem subtle, but after performing an experiment it becomes

clearer.

$$MAE = \frac{\sum_i^N |Y_i - \hat{y}_i|}{N} \tag{10}$$

Another concern regarding the experimentation process can be raised with respect to the partition into train and validation sets. Since we have agreed that it must be done randomly, the final distribution might result in a pathological division which leads to particularly poor results or, conversely, to a division which places examples in the validation set which are remarkably easy to predict. In order to solve this issue, cross-validation can be used, which consists of randomly splitting the dataset into train and validation several times and then taking the average values. With this technique, extreme values are smoothed and results are more likely to be able to generalize to new samples.

3.2 Quantum Chemistry

In addition to the theoretical foundations on ML, a thorough review on quantum chemical calculations must be considered to understand what type of assumptions and limitations are needed in each case. For the scope of the present work, only semiempirical level of theory will be covered. Since DFT is only used as a reference method and due to the limited nature of this report, we will avoid expanding on the intricacies behind the method. Due to its increasing popularity, the essentials can be found in most introductory physical chemistry textbooks like Atkins [16].

3.2.1 Semiempirical Methods

Semiempirical methods are considered *ab initio* electronic structure methods which emerged as an efficient implementation of Hartree-Fock schemes. Nevertheless, Hartree-Fock methods require solving large matrices and integrals beyond

numerical feasibility. Assuming that the wavefunction can be approximated in the form of a Slater determinant containing the molecular orbitals as N one-electron wave functions, the next step would be to find the Slater determinant that minimizes the energy by means of the variational principle. That is, finding the wavefunction that minimizes the integral:

$$E = \min_{\Theta_S \to N} \langle \Theta_S | \hat{H} | \Theta_S \rangle \tag{11}$$

Where $\hat{H}$ is the electronic Hamiltonian and Θ_S is the Slater determinant. In the LCAO MO approximation, an electron can be described by means of a molecular orbital (OM), written as a linear combination of atomic orbitals (LCAO):

$$\varphi(\mathbf{r_i}) = \frac{1}{\sqrt{N_i}} \sum_l c_l^i \phi_i(\mathbf{r_i}) \tag{12}$$

Where $\phi_i(\vec{x})$ represents individual atomic orbitals and each c_i^l is a coefficient that must be determined by means of a variational method. The normalization constant is determined by:

$$N_i = \sum_l \sum_k c_k^i c_l^i S_{kl} \tag{13}$$

S_{kl} is the overlap integral between k and l atomic orbitals. At this point it is useful to define three integrals which are central to the Hartree-Fock framework. The overlap integral is calculated as:

$$S_{kl} = \int \varphi_k(\mathbf{r_2}) \, \varphi_l(\mathbf{r_2}) \, d^3 r_2 \tag{14}$$

Then, the Coulomb integral which accounts for the electronic repulsion is defined

as:

$$C_{kl} = \int \varphi_k \left(\mathbf{r_1}\right)^2 \left(\frac{1}{R_{kl}} + \frac{1}{r_{12}} - \frac{1}{r_{k1}} - \frac{1}{r_{l2}}\right) \varphi_l \left(\mathbf{r_2}\right)^2 d^3 r_1 d^3 r_2 \tag{15}$$

Finally, the exchange interaction integral which does not have a classical analogue with no clear physical explanation. It accounts for quantum mechanical phenomena:

$$J_{kl} = \int \varphi_k^* \left(\mathbf{r_1}\right) \varphi_l^* \left(\mathbf{r_2}\right) \left(\frac{1}{R_{kl}} + \frac{1}{r_{12}} - \frac{1}{r_{k1}} - \frac{1}{r_{l2}}\right) \varphi_l \left(\mathbf{r_1}\right) \varphi_k \left(\mathbf{r_2}\right) d^3 r_1 d^3 r_2 \tag{16}$$

A thorough explanation on how to expand this calculation can be found in a regular quantum chemistry textbook as it is out of the scope of this work. We will only present results, where we can skip to the following formula:

$$FC_k = \epsilon_k S C_k \tag{17}$$

Which is known as the Fock secular equation, where C_k is the matrix containing the coefficients. F matrix is the Fock matrix, where the ith element is computed as:

$$F_i = h_i + \sum_j^{N/2} (2C_j - J_j) \tag{18}$$

Where h_i is the corresponding one-electron hamiltonian. Then, the problem is reduced to a diagonalization exercise, which requires computationally expensive integral evaluations in order to obtain both F and S matrices.

It is important to emphasize the final objective of this method. We want to find a diagonal matrix C_k which contains the coefficients that grant a functional form for the wavefunction. Calculating the energy by means of equation 11 leads to the ground state energy.

Since evaluating the integrals needed for the variational method are extremely expensive from the computational perspective, there were several attempts to elude them. The prevalent version of these approximations is the so called NDDO type of semiempirical methods. NDDO stands for neglect of diatomic differential overlap and its purpose is to equate a certain type of integral to zero. In summary, it neglects the overlap over two basis functions $\varphi_i \varphi_j d\mathbf{r}$ where the functions $\varphi_i(\mathbf{r}-\mathbf{R_M})$ and $\varphi_n(\mathbf{r}-\mathbf{R_N})$ are centered on different nuclei. The consequence is that the number of repulsion integrals that must be computed is drastically reduced [17].

In summary, semiempirical methods are Hartree-Fock based schemes where an additional approximation is considered in order to elude the calculation of an enormous number of integrals. This is where semiempirical methods attract more attention compared to DFT. DFT scaling law follows cubic complexity $\mathcal{O}(n^3)$ while semiempirical NDDO methods as those incorporated in MOPAC lead to linear complexity $\mathcal{O}(n)$.

While there is a wide variety of semiempirical methods currently available, PM7 [18], included in MOPAC, is corroborated as the best performing method. PM7 was parametrized to achieve greater performances with molecules outside of the subset of molecules employed for the parametrization. With respect to the approximations and assumptions made for PM7, two major modifications were reviewed. First, it enhanced the description of non-covalent interactions by means of including dispersion and hydrogen bonds. Then, errors under the NDDO formalism were corrected.

Discussions regarding quantum chemical methods open the opportunity to clarify a term widely used in this work. When we refer to activation energies, we are actually referring to barrier heights. Nevertheless, after an extensive literature research, the term coined for barrier heights in the context of machine learning is activation energies. True activation energies are dependent on the temperature and would pose a much more intricate problem.

4 Experimental Procedure

4.1 Database Curation

As mentioned previously, the first task we need to carry out after downloading the database contents is to perform the calculations at the semiempirical level. Our target data is $Y = E_a^{DFT} - E_a^{PM7}$, which makes necessary the calculation of barrier heights at both the DFT and PM7 levels of theory. The database contains 11960 folders including output files produced after the optimizations for reactant, transition state and product. These output files were generated from Q-Chem software, at the ωB97X-D3 level of DFT theory using def2-TZVP basis expansion. Barrier heights are directly obtained by subtracting the TS final energy minus the reactant energy, considering ZPE.

On the other hand, obtaining barrier heights at the semiempirical level entailed a much more complex process, as several sanity checks are required. Figure 7 presents a flow chart diagram explaining how the calculations are performed step by step. Further explanations are needed and the following paragraphs are allocated to understanding the diagram.

First step, labeled as TS optimization in Figure 7 consists on parsing transition state coordinates from the database and performing an optimization step using MOPAC at semiempirical level of theory starting from the database geometry. After this step, it is crucial to check whether this TS connects reactants and products present in the database. That is, we must inspect if the optimized TS with MOPAC corresponds to the same reaction as per the database.

In order to verify which reactants each TS leads to, the Intrinsic Reaction Coordinate (IRC) must be followed. The reaction path IRC method consists on finding the reaction path corresponding to the steepest descent curve from the first-order saddle point. That is, the path which maximizes the gradient starting from the TS

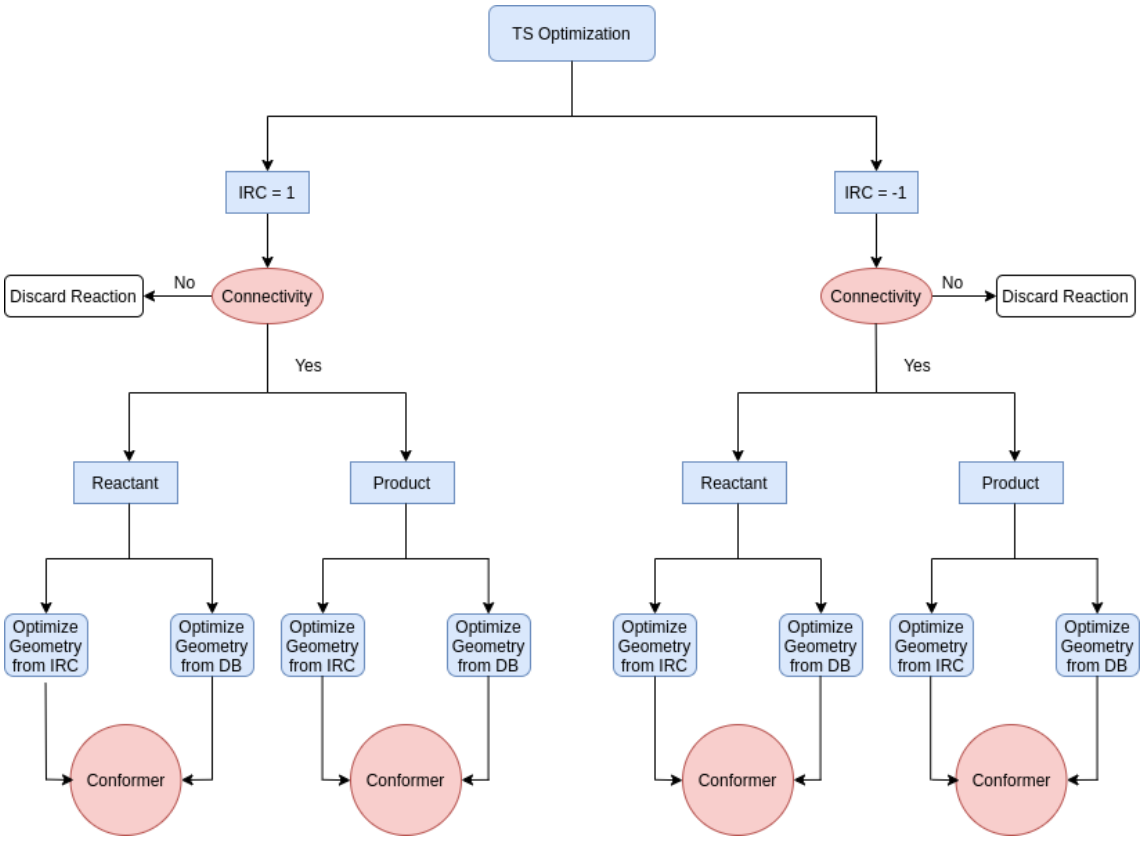


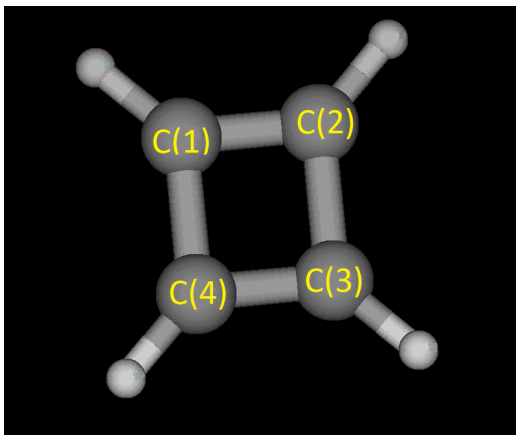
Figure 7: Dataset generation flow diagram

coordinates within the potential energy surface and eventually arriving at a minimum. This steepest descent reaction path in mass weighted coordinates is usually what is referred to by IRC [19].

This calculation can be performed in different directions starting at the TS geometry. Each direction is represented in the flow diagram by means of the keyword *IRC = 1* or *IRC = -1*. Ideally, after performing this step, the minima obtained for each TS should correspond to the reactants and products present in the reaction involving that specific TS. Thus, the next step boils down to comparing different geometries and resolving whether they represent the same molecule. This step can be automated by resorting to spectral graph theory.

A graph is the natural mathematical representation of a molecule, encoding how atoms are connected (bonds) in matrix form. The most relevant matrix is the

adjacency matrix, which can be easily visualized by means of the example displayed in Figure 8. This figure represents a cyclobutadiene molecule with carbon atoms labeled. The adjacency matrix is a 4×4 matrix where the diagonal is always filled with zeros and the elements outside of the diagonal are 1 at position a_{ij} if the atoms ij are bonded and 0 otherwise. In this particular case, carbon number 1 is attached both to carbon number 2 and carbon number 3. Thus, elements $\{a_{12}, a_{13}, a_{21}, a_{31}\}$ are non-zero.



$$\begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{pmatrix} \quad (19)$$

Adjacency Matrix

Figure 8: Cyclobutadiene

It is clear from this example that the shape of the adjacency matrix is dependent on the labelling of the molecular graph. Thus, a graph invariant must be used to compare whether two different matrices represent the same molecule. This graph invariant can be obtained by substituting each diagonal element by the atomic number of the atom that it represents. Then, eigenvalues for this modified adjacency matrix are computed. The eigenvalues are labelling invariant and only depend on the molecule connectivity. We have found a graph invariant to automate the verification of whether IRC reactants and products match those present on our database.

If the connectivity differs, the reaction must be discarded. If they are the same molecule, both the geometry from the database and the geometry from the IRC calculation are optimized at PM7 level of theory. Finally, the last step on the flow diagram refers to the corroboration of both molecules being the same conformer.

This last step is performed in a similar manner using methods from spectral graph theory.

In conclusion, this database curation lead to the discard of part of the reactions present in the database, either because the species could not be optimized at PM7 level of theory or the IRC calculation lead to different reactants and products. From the initial 11960 reactions, 8355 survived this screening process, meaning that roughly 70 % of the samples could be utilized.

Considering these available data entries, we could calculate the activation energy. Equations 20 and 21 refer to how the barrier height from reactants to transition state and product respectively are computed.

$$E_a = (H_F^{TS} + \mathrm{ZPE}_{TS}) - (H_F^R + \mathrm{ZPE}_R) \tag{20}$$

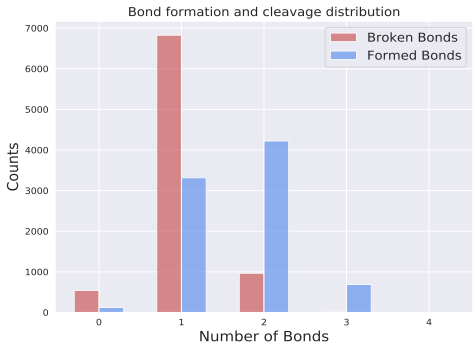
$$\Delta H = (H_F^P + \mathrm{ZPE}_P) - (H_F^R + \mathrm{ZPE}_R) \tag{21}$$

Where H_F refers to the heat of formation, which stands for the total electronic energy optimized by MOPAC and ZPE is the Zero-point energy, the vibrational energy corresponding to 0 K. This accounts for the differences between how *ab initio* and semiempirical methods represent the energy of a chemical system.

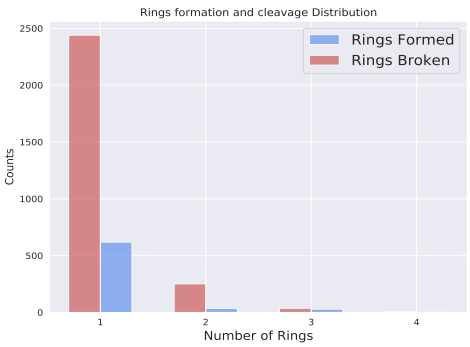
Both of these quantities will be our target variables. Remember that our purpose is to predict differences involing DFT and semiempirical level of theory, which can be written as $Y = (E_a^{DFT} - E_a^{PM7}, \Delta H^{DFT} - \Delta H^{PM7})$. Even though we are mainly interested in predicting activation energy, predicting both of these properties simultaneously tend to enhance the training process (usually known as Multitask Learning [20]). At this point, the dataset is already curated and cleansed to start with the machine learning pipeline.

4.2 Dataset Description

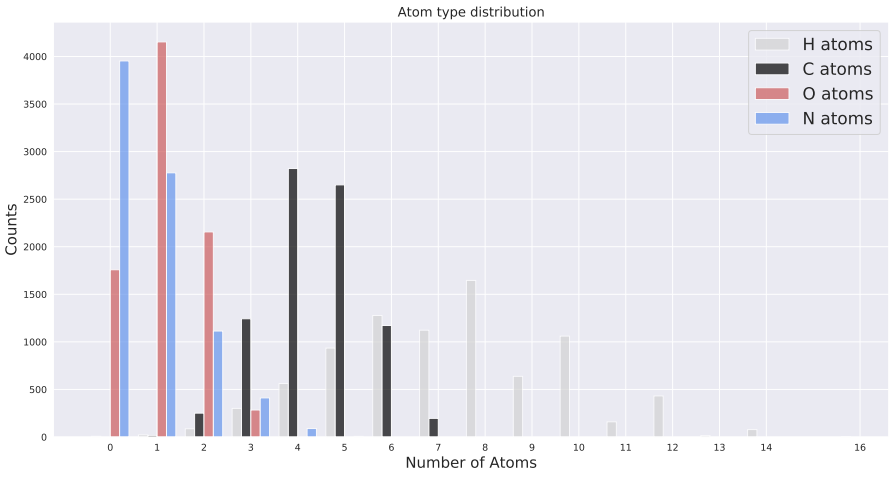
The first step in a machine learning problem is to investigate and analyze the data which will be used. In this case, exploratory data analysis (EDA) was performed on the Grambow et al. (2020) database. Figure 9 shows histogram plots depicting different characteristics of the reactions in the database. Figure 9 a) refers to the number of bonds either broken or formed in the reaction and b) the number of rings being formed in the product or broken. Last picture, Figure 9 c) shows the number of atoms per element present in each reaction.



a) Number of Broken/Formed Bonds



b) Number of Broken/Formed Rings



c) Number of atoms involved per reaction

Figure 9: Exploratory Data Analysis of the reactions present on the database

These histograms provide a powerful insight on the type of compounds present in the database and the statistical distribution of the type of molecules depending on the elements as well as bonds involved in the reactions.

This analysis grants a vision on the chemical nature of the compounds present in the database. Since the database is comprised of reactions connecting reactants with products, it is interesting to visualize a selection of reactions to understand what type of transformations we are working with. Figure 10 displays different examples, with the type of reactions ranging from ring opening/formation to tautomerisms (enolization).

Reactions in the database contain up to seven heavy-atoms (that is, carbon, nitrogen or oxygen) per molecule and consist on unimolecular reactants that can yield multi-molecular products (even though most reactions are isomerizations). The purpose with this dataset was to represent the widest spectrum of reactions while covering a broad range of activation energy values.

All these reactions are in gas-phase, with every reagent sampled from the chemical space spanned by GDB-7 dataset [21]. It is worth noting that for this type of reactions, it is possible to include the reverse reactions. Since our dataset contains information for every species involved in the reaction, we could extract reverse barrier heights.

Including direct and reverse barrier heights leads to doubling the number of reactions accessible. This procedure is a form of the common technique in ML known as data augmentation [22]. Data augmentation allows an increment on the number of training samples without the need for further electronic structure calculations.

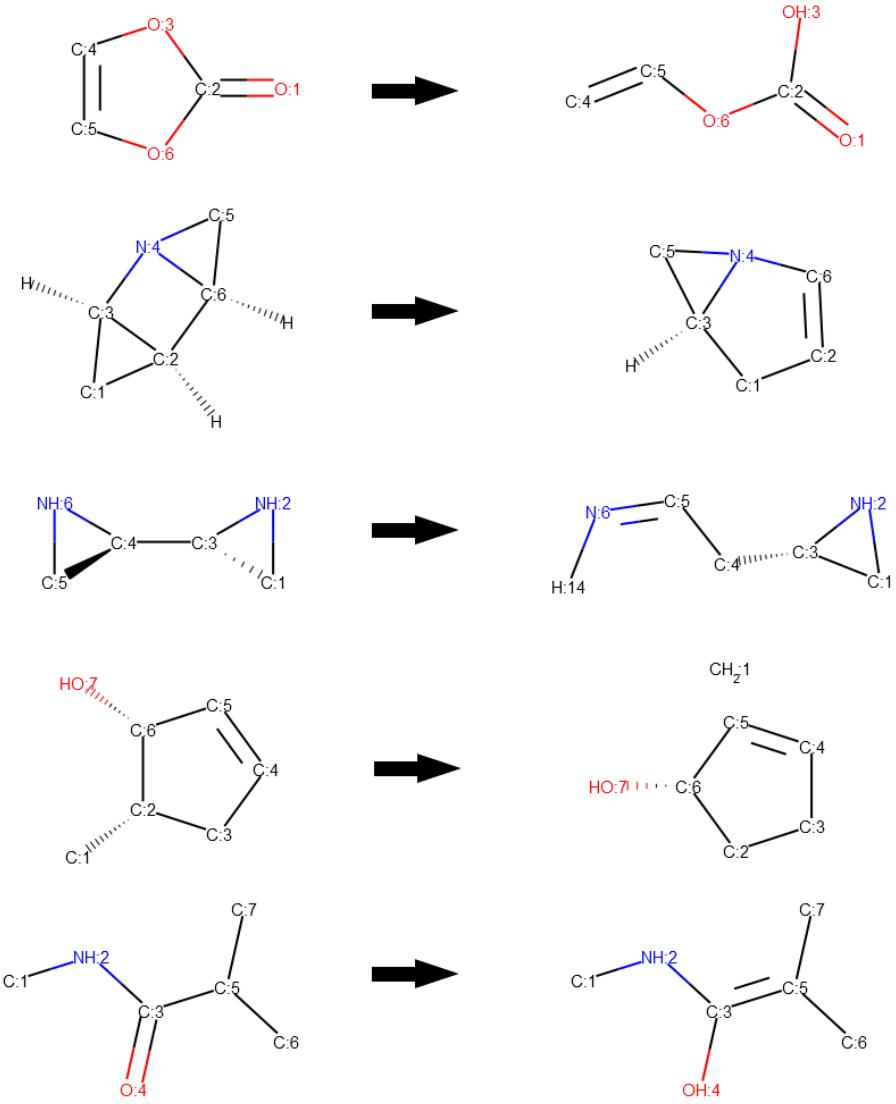


Figure 10: Reaction examples from the database entries

4.3 Descriptor Calculation

The most crucial step in this project is calculating a set of descriptors that encode the most useful information present in every reaction. These descriptors are the dependent variables X that serve as input for the model to predict the target variable Y. The nature of these descriptors range from purely topological (based on each

molecule connectivity) to electronic structure (obtained from MOPAC calculations).

In chemoinformatics, descriptors are usually calculated using RDKit library [23]. RDKit provides a set of descriptors based mostly on heuristics and demanding the molecule structure as the only input. In this project, descriptors were calculated both for products and reactants and the final descriptors used involved the difference $X_{RDKit} = X_{RDKit}^{Product} - X_{RDKit}^{Reactant}$. If the specific descriptor remained invariant both in products and reagents (molecular weight, for instance), the difference would always yield zero. In this case, the raw value was used.

In order to provide an intuition on some of the other descriptors provided by RDKit, we name some examples. Some desriptors can be as simple as the molecule's molecular weight or as sophisticated as calculating the topological polar surface area (TPSA) [24]. TPSA refers to the surface occupied by polar atoms, calculated by means of an heuristic method described on RDKit documentation [23].

There was still plenty of information that is not encapsulated in RDKit descriptors. Information regarding how the reaction occurs was not being employed. Then, another set of descriptors consisted on counting the number of broken and formed bonds for each reaction, as well as counting the number of bonds per type of atom involved. That is, number of carbon-carbon bonds broken, number of oxygen-hydrogen bonds formed, etc. Even the count of atoms per type of element was used as a descriptor, since the parametrizations in semiempirical methods are highly dependent on the type of elements.

The next set of descriptors was focused on extracting the most relevant information from the transition state itself. First, we decided to include descriptors that represent the TS connectivity and topology. From its adjacency matrix, several topological indexes can be computed. These topological indexes are graph invariants (remember the importance of using invariant objects). Using a more concrete example, one of the indexes employed was Randić's molecular connectivity index [25], which is a measure on the branching for an organic compound.

Apart from how atoms are connected in a molecule, it is also important to know what type of bonds are involved. In order to encode this information, we parsed the bond order matrix from MOPAC output files and calculated its eigenvalues. The eigenvalues of this bond order matrix served as new descriptors.

There was still room for capitalizing on the information provided by MOPAC calculations. By providing special keywords, it is possible to compute electrophilic and nucleophilic delocalizabilities, as well as charge densities [26]. These quantities provide an estimation on the energy stabilization upon the attack from an electrophilic or nucleophilic agent respectively. In general, they provide a picture of how the charge is distributed within a molecule.

4.4 Model Selection

After obtaining the complete set of descriptors, it is important to investigate if there are highly correlated variables. Correlated features would only add statistical noise to the model and the usual procedure is to remove descriptors that exceed a correlation threshold. The correlation measure employed was Pearson correlation coefficient:

$$r = \frac{\sum_i (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_i (x_i - \bar{x})^2 \sum_i (y_i - \bar{y})^2}} \tag{22}$$

Where x_i are the values for descriptor x and y_i for descriptor y. Considering every possible descriptor, a correlation matrix can be built where each entry represents the pairwise Pearson correlation coefficient between every pair of descriptors. A correlation threshold was established such that if a pair of descriptors exceed this value, one of them had to be dropped from the dataset. This threshold was optimized by cross-validation and the final value was set to $\rho = 0.9$.

The next step, as outlined in the previous chapter regarding experimentation in

ML, involves partitioning the dataset into training, validation and test sets. Following common practices in ML literature as well as considering the size of our dataset, we decided to use 85 % of the data for training purposes, 5 % for validation and 10 % for the test set.

After separating the test set, it is time to select the optimum model which provides the best performance in the validation set. Following the procedure outlined in the previous chapters, in order to obtain the best performing model we need to optimize its hyperparameters.

In the context of deep learning, there are several hyperparameters that can be fine-tuned. Since searching for the optima for every combination of hyperparameters simultaneously would be extremely computationally expensive, a sequential approach was carried out.

First, it was necessary to elucidate the number of hidden layers and the number of neurons per layer. The minimum was obtained by resorting to a grid search. This method entails trying every possible combination and examining the one that minimizes a given metric. For our experiment, we decided to use MAE as the optimizing metric. It is important to emphasize that this metric is validated on the samples not used for training, meaning that this metric is computed for the validation set.

With the aim of obtaining more general results, we used cross-validation. Cross-validation involves testing the model performance with different train/validation set partitions with the purpose of preventing pathological cases. Table 1 shows MAE for the different combinations that assemble various forms of model architecture.

Number of Hidden Layers	Number of Neurons per hidden layer	MAE (kcal/mol)
4	$128 \times 256 \times 64 \times 32$	5.04 ± 0.34
4	$128 \times 128 \times 64 \times 32$	5.14 ± 0.33
4	$128 \times 64 \times 64 \times 32$	5.21 ± 0.33
5	$128 \times 256 \times 128 \times 64 \times 32$	8.1 ± 4.3
5	$128 \times 128 \times 64 \times 64 \times 32$	5.13 ± 0.40
5	$128 \times 64 \times 32 \times 64 \times 32$	5.19 ± 0.36
6	$128 \times 256 \times 128 \times 64 \times 64 \times 32$	8.1 ± 4.5
6	$128 \times 128 \times 64 \times 32 \times 64 \times 32$	5.14 ± 0.28
6	$128 \times 64 \times 32 \times 16 \times 64 \times 32$	5.21 ± 0.23

Table 1: Neural Network Architecture hyperparameter tuning

Uncertainties were assigned by means of the standard deviations obtained because of the cross-validation procedure. More precisely, 3-fold cross-validation was employed, meaning that per every combination of hyperparameters, 3 different dataset partitions were used and the presented values refer to the mean value. After selecting 4 hidden layers as the optimum value with the corresponding number of neurons, we could fix what is usually referred to as the network architecture. Apart from obtaining the minimum MAE from this architecture, there is another reason to choose the model with the lowest possible number of hidden layers and neurons.

In the situation where two different models perform equally on the same dataset, it is always preferable to choose the simplest model. In this case, that would be the model with the least number of parameters, which depends on the number of neurons. The simpler the model is, the better it will extrapolate to data entries outside of the training and validation sets. Occam's razor is commonly mentioned as the argument to avoid high complexity models on deep learning approaches [27].

Reducing model complexity serves as a hedge against another undesirable phenomenon which can happen in machine learning settings. This issue is overfitting, which consists of training a model which learns to predict data on the training set to an extremely high accuracy but achieving poor performance on the test set. The main evidence that a model is overfitting is that it performs drastically better on the training set than on the test set and the more iterations, the worse result is obtained in the test set.

There are several approaches to avoid overfitting apart from increasing model simplicity. The most widely used resource is introducing dropout [28] within the model architecture. Dropout involves ignoring some connections between hidden layers with a given probability, such that statistical noise is introduced in the model in order to prevent the model from learning to excessively adapt to the training data. Dropout probability can be included as a hyperparameter as well which must be fine-tuned in a similar approach as network architechture.

Dropout probability was optimized together with another pair of hyperparameters: learning rate and batch size. Batch size in interpreted as the number of data entries that are passed down to the input layer per iteration and learning rate is a measure on the magnitude of parameter update in every gradient descent iteration.

Table 2 shows the results from validation set performances for different combinations of the hyperparameters using 3-fold cross-validation again. Since parameter space is still tractable , we decided to perform discretized grid search again, pursuing every possible set of parameters.

		Batch Size			
Learning Rate	Dropout Rate	128	256	512	
0.01	0.15	14.26 ± 0.19	14.24 ± 0.15	14.23 ± 0.17	MAE kcal/mol
0.001	0.15	4.61 ± 0.44	4.41 ± 0.40	4.38 ± 0.39	
0.0001	0.15	4.34 ± 0.35	4.30 ± 0.36	4.34 ± 0.36	
0.01	0.25	14.23 ± 0.16	14.24 ± 0.16	14.25 ± 0.15	
0.001	0.25	4.75 ± 0.31	4.57 ± 0.37	4.59 ± 0.32	
0.0001	0.25	4.52 ± 0.45	4.50 ± 0.37	4.60 ± 0.40	
0.01	0.35	14.24 ± 0.16	14.25 ± 0.14	14.23 ± 0.16	
0.001	0.35	4.99 ± 0.44	4.81 ± 0.30	5.01 ± 0.30	
0.0001	0.35	4.65 ± 0.34	4.78 ± 0.49	4.85 ± 0.35	

Table 2: Hyperparameter Optimization

From this experiment we can conclude that the optimum model is obtained by using 0.15 as dropout rate, 0.0001 as learning rate and a batch size of 128 samples. At this moment, it was interesting to include the data augmentation technique mentioned earlier. That is, including reverse barrier heights to increase the training data.

Using twice the training data lead to results that were within the confidence interval spanned by the optimized model: (4.34 ± 0.35) kcal/mol . We concluded that data augmentation in this case does not lead to better performances while penalizing computational expenditure. A sensible explanation is that our descriptors are heavily dependent on the transition state and reverse reactions share transition states with direct ones, introducing noise during training. This approach was abandoned and only direct reactions were finally used.

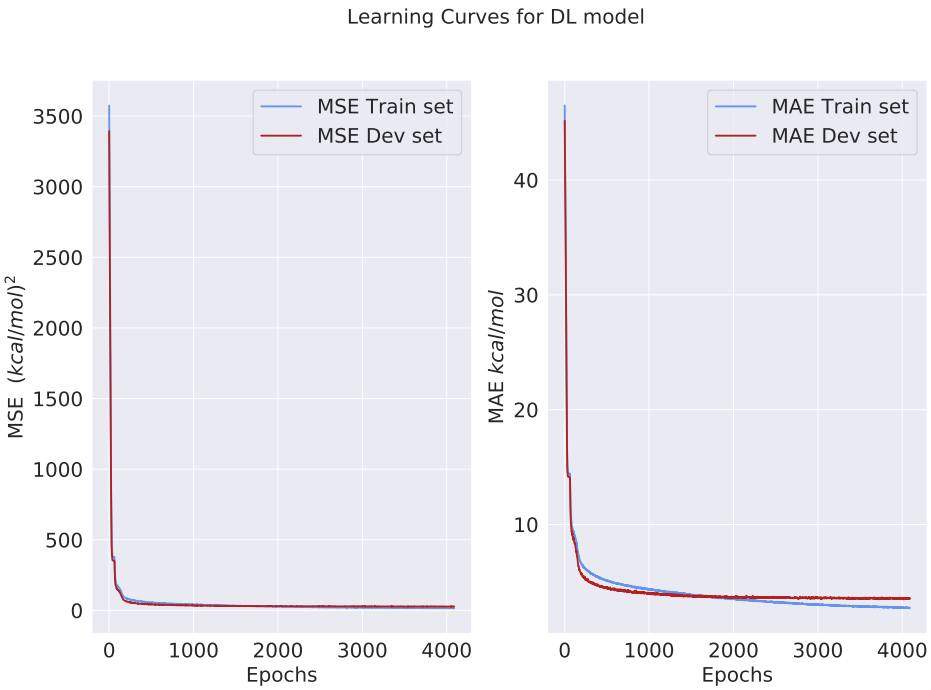


Figure 11: Learning Curves

Figure 11 represents how MSE and MAE evolve on the training and validation (dev) sets. This procedure serves to identify and troubleshoot potential problems during training process. Since both metrics evolve similarly for every iteration, with the validation loss function only diverging in the end, we can state that the model effectively learns to predict samples from the training set and it is able to extrapolate to the validation set successfully. In other words, from evaluating this plot we can explore whether underfitting or overfitting is involved in this learning problem.

In order to visualize the final model architecture and dimensions, Figure 12 displays a representation containing all the information needed to reproduce this model.

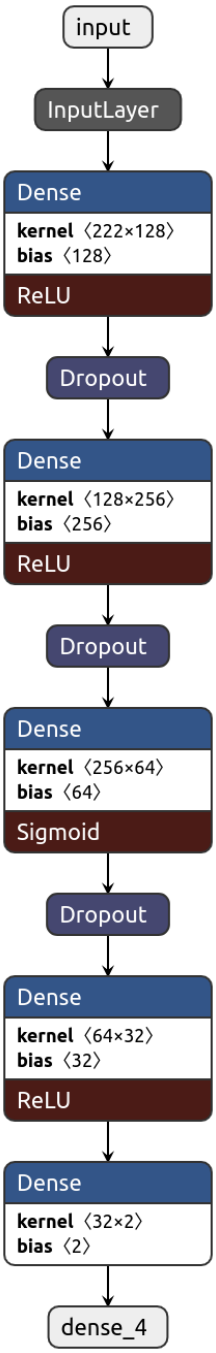


Figure 12: Final model architecture

Apart from the number of neurons per layer, the activation function used is also included, as well as how dropout was implemented. Dense layers refer to hidden layers where every neuron is fully connected. Note that the input layer requires

entries to have 222 dimensions, which corresponds to the number of descriptors needed. On the other hand, the output layer produces two dimensions, due to the fact that we are simultaneously predicting both activation energies and product-reactants energy differences.

Upon obtaining a model with the optimum hyperparameters, it is time to compare its performance with respect to simply using PM7 calculations. Figure 13 shows on the left activation energies calculated at DFT level vs activation energies at PM7 level for the test set. Using MAE as a metric, the value obtained is 11.24 kcal/mol, with semiempirical calculations inducing systematic errors, as PM7 underestimates high activation energies (for higher values of activation energies, the majority of the points lie above the $y = x$ black line).

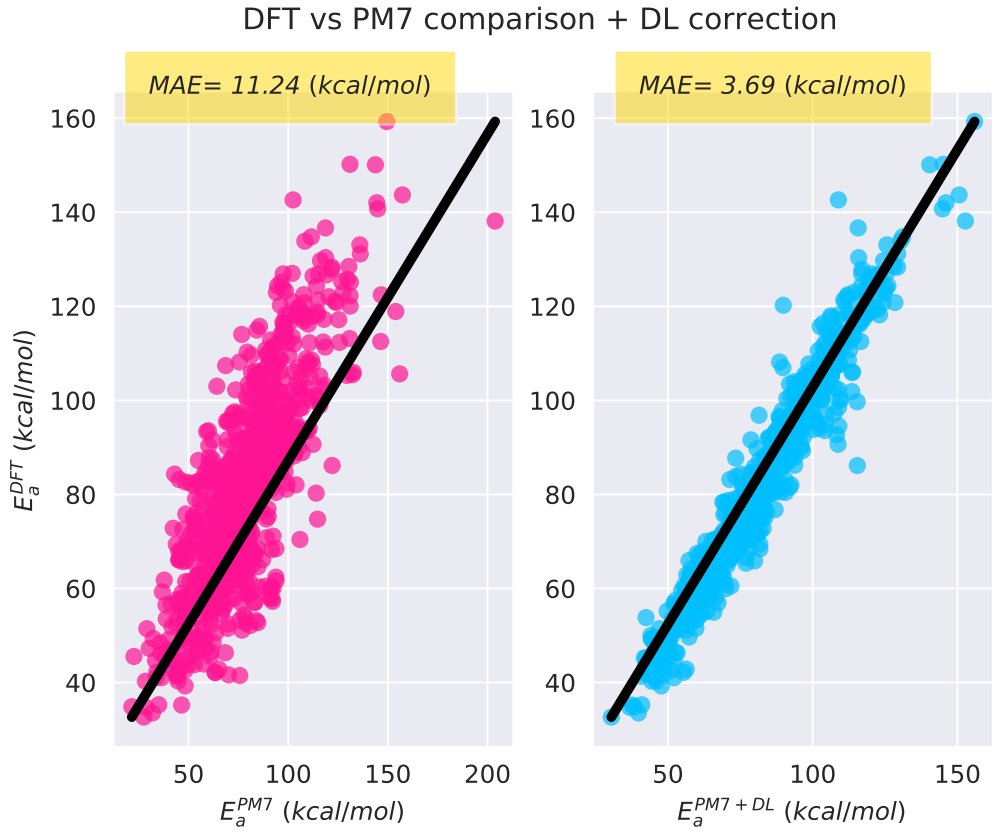


Figure 13: DL corrected activation energies

On the right, activation energies at DFT level vs DL-corrected activation energies.

The model performs equivalently well in every energy range, with no systematic bias, obtaining a MAE on the test set of 3.69 kcal/mol.

The previous plot does not provide an accurate picture on how the model performs for the test set, because most of the samples from the test set are around 100 kcal/mol. Figure 14 shows how the correction works with a color code representing the number of data points lying in the same pixel. With this type of graph, it is clear that there is a compact density of points whose activation energies are accurately corrected.

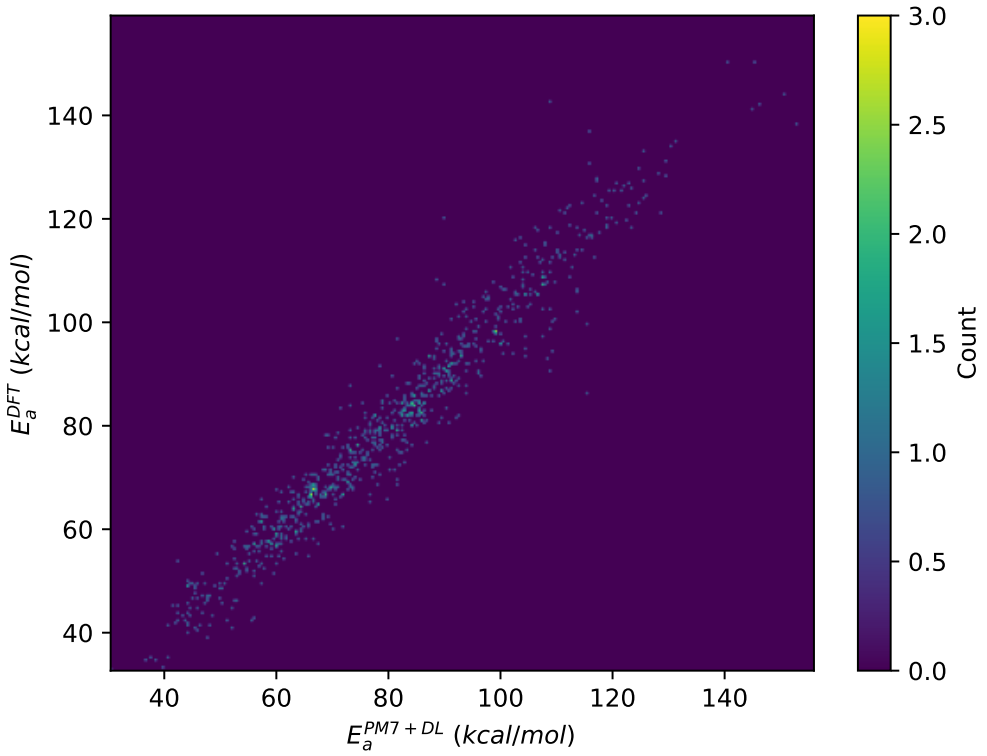


Figure 14: Density of corrected activation energies values.

Apart from harnessing the power of this model with respect to its predictive power, it is equally important to asses how interpertable the model is (as opposed to black-box models). In order to extract more information from this model, we will employ Shap values [29]. Shap values is an approach that helps understanding how

a machine learning model output is produced. Its foundation lays in the domain of game theory, but we will not provide a detailed explanation since it is out of the scope of this work. In order to comprehend the intuition behind this, it tries to make a prediction without employing a certain descriptor and validating how the model performs without this descriptor. Thus, we can obtain a measure of the most important descriptors and how they affect the prediction.

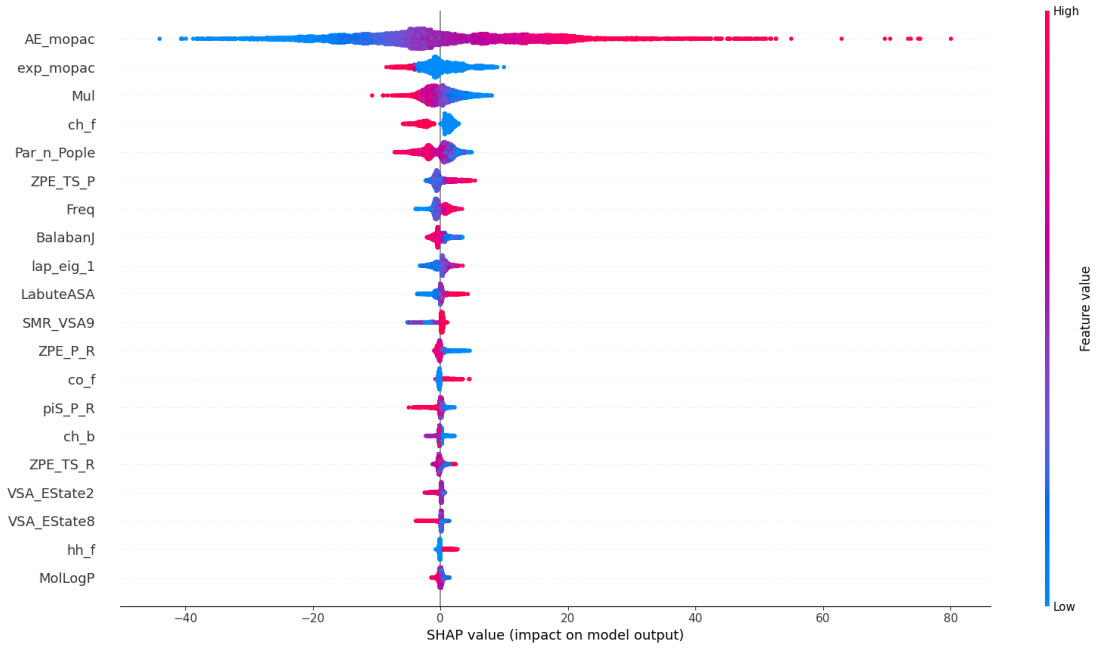


Figure 15: Shap values sorted by descriptor importance

Figure 15 represents the most relevant descriptors sorted by importance. It is interesting to notice how the activation energy computed at low level (MOPAC) as well as the exponential of this value represent the most important descriptors. Then, another point that emphasizes the interest behind our approach is how the most important features were novel descriptors proposed in this work rather than descriptors obtained from RDKit library.

The model weights descriptors related to charge distribution as high priority descriptors, such as Parr and Pople absolute hardness (Par_n_Pople) and Mulliken electronegativity (Mul), which are both HOMO and LUMO linear combinations.

Then, descriptors related to the topology of the transition state are next in importance, such as the first frequency (*freq*) and the second eigenvalue of the laplacian matrix (*lap_eig_1*), which grant information regarding how tight or loose the TS state is. Last, it is also worth underlining that bonds participating in the reaction can serve as powerful descriptors, such as formed carbon-hydrogen bonds (*ch_f*) or formed carbon-oxygen bonds (*co_f*).

4.5 Alternatives

After seeing the success achieved using artificial neural networks type of models, we investigated the implementation of more sophisticated models. At this point, given the proven record of successes in the field of computer vision [30], we opted for using Convolutional Neural Networks (CNNs). The intuition behind how CNNs work with images is that by performing a convolution operation on different regions of the input images, it is possible to extract new features that resemble how humans interpret visual information.

CNNs are incredibly powerful at pattern recognition from matrices that represent images. The idea behind this approach is encoding molecules as matrices. Then, our model would recognize patterns such as functional groups and their influence in the prediction on activation energies. We decided to use Coulomb matrices, defined by equation 23.

$$C_{ij} = \begin{cases} 0.5Z_i^{2.4} & i = j \\ \frac{Z_i Z_j}{||R_i - R_j||} & \forall i \neq j \end{cases} \tag{23}$$

Where Z_i refers to the atomic number or nuclear charge on atom i and R_i is its position in atomic units (Bohr radii a_0). Thus, the purpose of Coulomb matrices is to account for the electronic (Coulombic) repulsion. It is important to note that these matrices are translation and rotation invariants but they depend on atom

permutations. In order to solve this issue, rows must be sorted by their L2-norm. Figure 16 shows a representation of the coulomb matrix of an organic molecule.

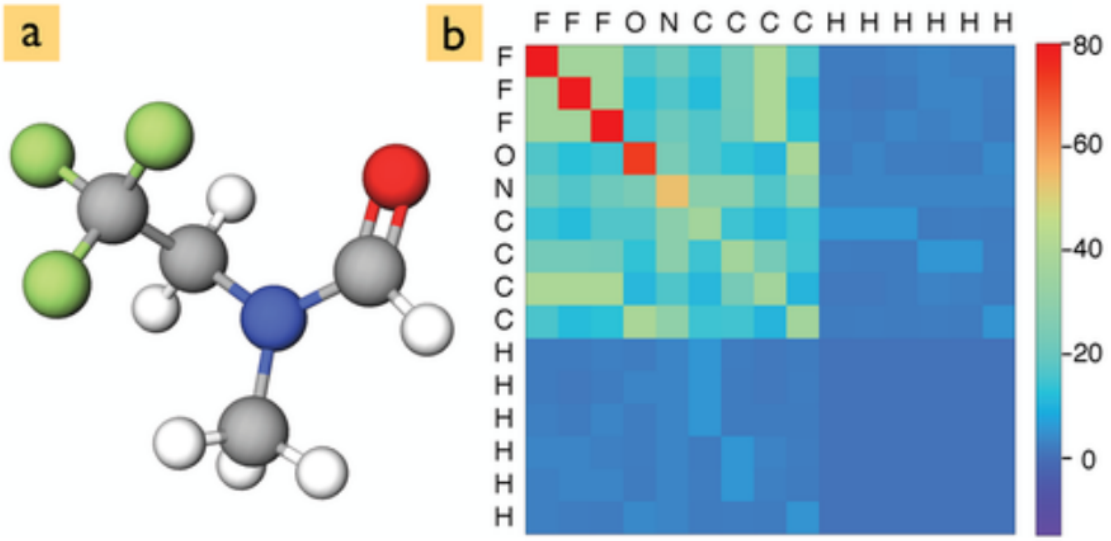


Figure 16: Coulomb Matrix representation [31].

After generating a dataset containing Coulomb matrices for each transition state, we decided to train a CNN model with the architecture described by Andrew Rabinovich et al. (2014) [32]. Even though this type of model performs extraordinarily well for image classification tasks, it can render robust results for regression tasks as well.

Results from training this model are displayed on Figure 17. On the left, MSE for both train and dev sets are shown and MAE on the right. Both figures evidence an important problem with this model. As more iterations progress, the model learns to predict activation energies from the training set surprisingly well. For the validation set, not only does it not improve its accuracy with every iteration, but actually provides worse results. This situation confirms that the model fails to extrapolate to unseen data and suffers from overfitting.

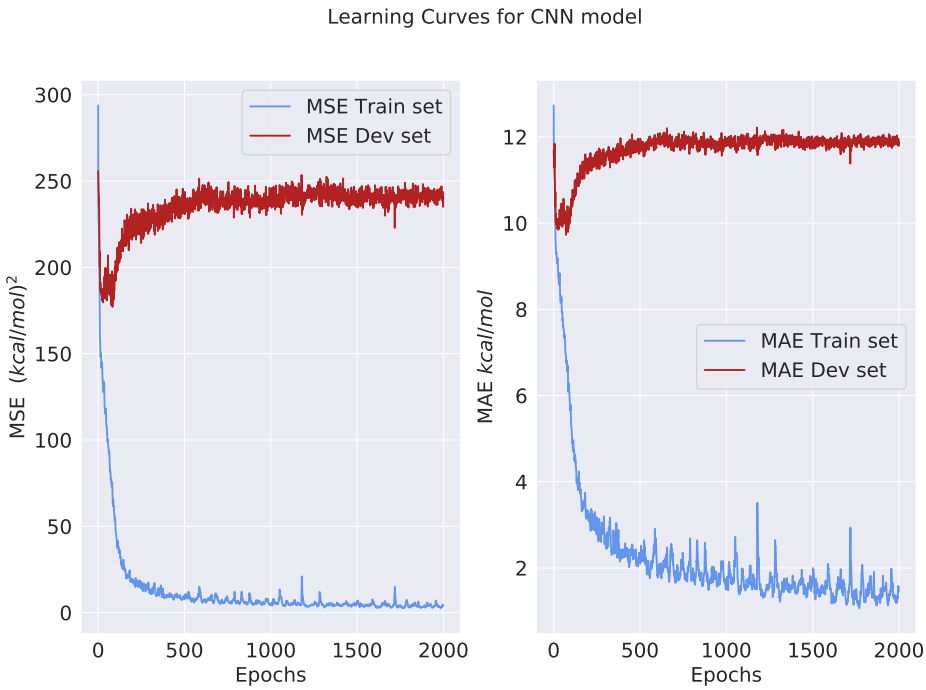


Figure 17: DL corrected activation energies

When working with CNNs, there is another approach when the dataset is limited. Transfer learning consists on using a pre-trained model, with parameters trained on a different task. Then, the output layer is extracted and replaced by a new one, customized for the problem that we are trying to solve. Then, we used VGG-16 model [33], trained on a large dataset containing images. This approach lead to the same results shown in Figure 17, suffering from the same overfitting issue and unable to generalize to molecules outside of the training set. In order for CNN models to work for this problem, we would need either to generate a larger dataset or to obtain a pre-trained model from a closer application. For image recognition tasks, dataset usually rely on millions of samples, which is orders of magnitude above our database.

5 Conclusions

This work firmly argues in favor of all the excitement around machine learning and artificial intelligence in the physical sciences. By resorting to machine learning, we could predict activation energies at DFT level with the computational expenditure of semiempirical calculations.

Apart from beating state of the art models when it comes to quantitative reaction properties prediction, there are other features of equal importance in this work. First, we demonstrated how low level, computationally inexpensive semiempirical calculations serve as a stepping stone to provide more accurate models.

With respect to descriptors, it is important to emphasize the interest behind using information contained in transition states, a detail vastly omitted in common literature. On top of that, our custom descriptors, based on graph-theoretic aspects and electron density (amongst others), can be adapted by practitioners to expand on current data mining techniques used in chemistry.

On the other hand, we introduced the utility behind explainable machine learning models in the context of chemoinformatics which connects descriptors calculated and model outputs. Not only this approach leads to insights on the most important features, but on how they influence predictions. This line of work can benefit further parametrizations on semiempirical methods. It is worth noting how analyzing the most relevant descriptors, the model prioritized our custom designed descriptors rather than the ones used from common literature.

Further work on this topic can be extended from project, as there are several aspects that were unexplored. The most straight-forward solution to enhance model accuracy and its ability to extrapolate to different systems is to expand the database by performing more high-level calculations and increasing the number of reactions contained. In particular, it would be interesting to include molecules which are sufficiently different such that the underlying statistical distribution comprises a

wider variety of systems and became less homogeneous.

As the size of the dataset increases, more sophisticated models can be used which require less information (less descriptors) to achieve chemical accuracy, such CNN previously described in. As a concluding remark for this project, we can state that while machine learning approaches are already present in computational chemistry research, there is still room for exploiting the gargantuan amount of data available in this domain.

5.1 Conclusiones

Este trabajo proporciona evidencia a favor de la exaltación entorno al uso de *machine learning* e inteligencia artificial en las ciencias físicas. Al recurrir a *machine learning*, conseguimos predecir energías de activación a nivel DFT con el coste computacional de cálculos semiempíricos.

Además de superar modelos que suponen el estado del arte en cuanto a predicción cuantitativa de propiedades de reacción, presentamos otros resultados de importancia equivalente en este trabajo. Primero, demostramos como un cálculo a bajo nivel, barato computacionalmente sirve como base para obtener modelos más precisos.

En cuánto a los descriptores utilizados, es importante enfatizar el interés en recurrir a la información encapsulada en los estados de transición, un detalle omitido generalmente en la literatura. Además de esto, nuestros descriptores adaptados, basados en aspectos de la teoría de grafos y la densidad electrónica (entre otros), pueden ser adoptados por los usuarios para expandir las técnicas de minería de datos empleadas habitualmente en química.

Por otra parte, introducimos el interés de utilizar modelos de *machine learning* explicables o interpretables en el contexto de la química. Esta forma de abordar el problema no sólo proporciona los descriptores más importantes, además nos informa de la influencia de cada descriptor sobre la predicción. Esta línea de trabajo puede

beneficiar próximas parametrizaciones de métodos semiempíricos. Es importante destacar como al analizar los descriptores que el modelo considera más importantes, encontramos que los descriptores propuestos en nuestro trabajo son los priorizados por el modelo en vez de los extraídos de la literatura.

Futuro trabajo en este tema puede extenderse a partir de este proyecto, ya que todavía quedan alternativas por explorar. La solución más directa para mejorar la precisión del método y su habilidad para extrapolar a sistemas diferentes es expandir la base de datos mediante más cálculos a alto nivel para aumentar el número de reacciones contenidas. En particular, sería interesante incluir moléculas suficientemente diferentes para que la distribución estadística subyacente tuviese en cuenta una mayor variedad de moléculas y fuese menos homogénea.

A medida que se incrementa el tamaño del conjunto de datos, es posible recurrir a modelos más sofisticados que requieren menos información (menos descriptores) para conseguir precisión química, como las CNN probadas en este trabajo. Como una nota concluyente para este proyecto, podemos afirmar que aunque las soluciones basadas en *machine learning* ya están presentes en investigación en química computacional, todavía se puede explotar más la ingente cantidad de datos de los que disponemos en este campo.

5.2 Conclusións

Este traballo proporciona evidencia a favor da exaltación entorno ao uso de *machine learning* e intelixencia artificial no eido das ciencias físicas. Ao recorrer ao *machine learning*, conseguimos predicir enerxías de activación a nivel DFT co custo computacional de cálculos semiempíricos.

Ademais de superar modelos que supoñen o estado da arte en canto a predición cuantitativa de propiedades de reacción, presentamos outros resultados de importancia equivalente neste traballo. Primeiro, demostramos como un cálculo a baixo nivel,

barato computacionalmente serve como base para obter modelos máis precisos.

En canto aos descritores empregados, é importante enfatizar o interese en empregar a información encapsulada nos estados de transición, un detalle omitido xeralmente na literatura. Ademais disto, os nosotros descritores adaptados, baseados en aspectos da teoría de grafos e a densidade electrónica (entre outros), poden ser adoptados polos usuarios para expandir as técnicas de minaría de datos usadas habitualmente na química.

Por outra parte, introducimos o interese de empregar modelos de *machine learning* explicables ou interpretables no contexto da química. Esta forma de abordar o problema non só proporciona os descritores máis importantes, ademáis infórmanos da influenza de cada descritor sobre a predición. Esta liña de traballo pode beneficiar próximas parametrizacións de métodos semiempíricos. Cómpre salientar como ao analizar os descritores que o modelo considera máis importantes, atopamos que os descritores propostos no noso traballo son os priorizados polo modelo en vez dos extraídos da literatura.

Futuro traballo neste tema pode extenderse a partir deste proxecto, xa que aínda quedan alternativas por explorar. A solución máis directa para mellorar a precisión do método e a súa habilidade para extrapolar a sistemas diferentes é expandir a base de datos mediante máis cálculos a alto nivel para aumentar o número de reaccións contidas. En particular, sería interesante incluir moléculas suficientemente diferentes para que a distribución estatística subxacente tivese en conta unha maior variedade de moléculas e fose menos homoxénea.

A medida que se incrementa o tamaño do conxunto de datos, é posible recorrer a modelos máis sofisticados que requiren menos información (menos descritores) para acadar precisión química, como as CNN probadas neste traballo. Como nota concluínte, podemos afirmar que aínda que as solucións baseadas en *machine learning* xa están incorporadas na investigación en química computacional, podemos explotar máis a inxente cantidade de datos da que dispoñemos neste campo.

References

[1] A.J. Duke and R.F.W. Bader. "A Hartree-Fock SCF calculation of the activation energies for two SN2 reactions". *Chemical Physics Letters* (1971).

[2] James J. P. Stewart. "Optimization of parameters for semiempirical methods I. Method". *Journal of Computational Chemistry* (1989).

[3] W. Kohn and L. J. Sham. "Self-Consistent Equations Including Exchange and Correlation Effects". *Phys. Rev.* (1965).

[4] Luke W. Bertels, Joonho Lee, and Martin Head-Gordon. "Polishing the Gold Standard: The Role of Orbital Choice in CCSD(T) Vibrational Frequency Prediction". *Journal of Chemical Theory and Computation* (2021).

[5] Adam C. Mater and Michelle L. Coote. "Deep Learning in Chemistry". *Journal of Chemical Information and Modeling* (2019).

[6] Ana M. Jiménez-Carvelo, Antonio González-Casado, M. Gracia Bagur-González, and Luis Cuadros-Rodríguez. "Alternative data mining/machine learning methods for the analytical evaluation of food quality and authenticity – A review". *Food Research International* (2019).

[7] Logan Ward, Ankit Agrawal, Alok Choudhary, and Christopher Wolverton. "A general-purpose machine learning framework for predicting properties of inorganic materials". *npj Computational Materials* (2016).

[8] Michael Gastegger and Philipp Marquetand. "High-Dimensional Neural Network Potentials for Organic Reactions and an Improved Training Algorithm". *Journal of Chemical Theory and Computation* (2015).

[9] Thomas C. Allison. "Application of an Artificial Neural Network to the Prediction of OH Radical Reaction Rate Constants for Evaluating Global Warming Potential". *The Journal of Physical Chemistry B* (2016).

[10] Colin A. Grambow, Lagnajit Pattanaik, and William H. Green. "Reactants, products, and transition states of elementary chemical reactions based on quantum chemistry". *Scientific Data* (2020).

[11] Alfred North Whitehead and Bertrand Russell. *Principia Mathematica*. Cambridge University Press, 1925–1927.

[12] F. Rosenblatt. *The Perceptron, a Perceiving and Recognizing Automaton Project Para*. Report: Cornell Aeronautical Laboratory. Cornell Aeronautical Laboratory, 1957.

[13] Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, Edouard Duchesnay, and Gilles Louppe. "Scikit-learn: Machine Learning in Python". *Journal of Machine Learning Research* (2012).

[14] Sebastian Ruder. "An overview of gradient descent optimization algorithms". *arXiv preprint arXiv:1609.04747* (2016).

[15] Henry J Kelley. "Gradient theory of optimal flight paths". *Ars Journal* (1960).

[16] Peter Atkins and Julio Paula. *Atkins' Physical chemistry*. Oxford University press, 2008.

[17] Wolfhard Koch. "Neglect of Diatomic Differential Overlap (NDDO) in Non-Empirical Quantum Chemical Orbital Theories". *Zeitschrift für Naturforschung A* (1993).

[18] James J. P. Stewart. "Optimization of parameters for semiempirical methods VI: more modifications to the NDDO approximations and re-optimization of parameters". *Journal of Molecular Modeling* (2013).

[19] Ramon Crehuet and Josep Maria Bofill. "The reaction path intrinsic reaction coordinate method and the Hamilton–Jacobi theory". *The Journal of Chemical Physics* (2005).

[20] Rich Caruana. "Multitask Learning". *Machine Learning* (1997).

[21] Lars Ruddigkeit, Ruud van Deursen, Lorenz C. Blum, and Jean-Louis Reymond. "Enumeration of 166 Billion Organic Small Molecules in the Chemical Universe Database GDB-17". *Journal of Chemical Information and Modeling* (2012).

[22] Luke Taylor and Geoff Nitschke. "Improving Deep Learning with Generic Data Augmentation". 2018.

[23] Greg Landrum. "RDKit: Open-Source Cheminformatics Software" (2016), Visited 1/07/2021. URL: `https://github.com/rdkit/rdkit/releases/tag/Release_2016_09_4`.

[24] Peter Ertl, Bernhard Rohde, and Paul Selzer. "Fast Calculation of Molecular Polar Surface Area as a Sum of Fragment-Based Contributions and Its Application to the Prediction of Drug Transport Properties". *Journal of Medicinal Chemistry* (Sept. 2000).

[25] Milan Randic. "Characterization of molecular branching". *Journal of the American Chemical Society* (1975).

[26] Gerrit Schüürmann. "QSAR analysis of the acute fish toxicity of organic phosphorothionates using theoretically derived molecular descriptors". *Environmental Toxicology and Chemistry* (1990).

[27] Ke Sun and Frank Nielsen. "Lightlike Neuromanifolds, Occam's Razor and Deep Learning". *arXiv:1905.11027* (2021).

[28] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. "Dropout: A Simple Way to Prevent Neural Networks from Overfitting". *Journal of Machine Learning Research* (2014).

[29] Scott M Lundberg and Su-In Lee. "A Unified Approach to Interpreting Model Predictions". *Advances in Neural Information Processing Systems*. Ed. by I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett. Curran Associates, Inc., 2017.

[30] Waseem Rawat and Zenghui Wang. "Deep Convolutional Neural Networks for Image Classification: A Comprehensive Review". *Neural Computation* (2017).

[31] Kunal Ghosh, Annika Stuke, Milica Todorović, Peter Bjørn Jørgensen, Mikkel N. Schmidt, Aki Vehtari, and Patrick Rinke. "Deep Learning Spectroscopy: Neural Networks for Molecular Excitation Spectra". *Advanced Science* (2019).

[32] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. "Going Deeper with Convolutions". *arXiv:1409.4842* (2014).

[33] Karen Simonyan and Andrew Zisserman. "Very Deep Convolutional Networks for Large-Scale Image Recognition". *3rd International Conference on Learning Representations*. 2015.