Optimization of Multilevel Δ-Quantum Machine Learning Based on Neural Networks

van Wanrooij, Quinten M.P.M.

Award date:
2022

Master Thesis

Optimization of Multilevel Δ-Quantum Machine Learning Based on Neural Networks

Author:
Quinten van Wanrooij, 1000722

October 31, 2022

Supervisor:
Björn Baumeier

Abstract

The term Quantum Machine Learning (QML) refers to the application of machine learning methods for the prediction of properties of nanoscale objects, such as molecules. In a mathematical setting, these properties can, in principle, be derived from the solutions to the multi-electron Schrödinger equation. In practice, this equation cannot be solved exactly (except for simple systems) and one needs to resort to approximations of different computational complexity and varying accuracy. The machine learning used for this purpose, requires high-accuracy reference data to make predictions of the properties of interest purely based on the molecular structure (composition of atoms and their arrangement). This direct QML approach, targeting the highest quality reference, has shown to require quite a lot of, computationally expensive, datapoints.

In this project, an alternative QML approach, Δ-QML, will be investigated, in which the goal is to learn the difference $\Delta_b^t(m)$ between a low-accuracy baseline $D_b(m)$ and the high-accuracy property of interest $D_t(m)$, for molecule m, such that $E_t(m) = E_b(m) + \Delta_b^t(m)$. Specifically, we focus on multilevel Δ-QML, in which one (or more) intermediate quality reference calculations (with a lower computational cost) are introduced, i.e.

$$E_t(m) = E_b(m) + \Delta_b^i(m) + \Delta_i^t(m). \tag{1}$$

Where both Δ terms are obtained by two individually trained neural networks. In this report there is chosen to use 'SchNet', a neural network that is particularly used to model atomistic systems.

With the combined use of the multilevel Δ-QML method and SchNet, we investigated which intermediate step would result in the best balance between the mean absolute error (MAE) and computational cost. Furthermore, we looked into the possibility to use the Wasserstein distance – a metric used in optimal transport problems to measure the distance between two distributions – as a QML-free method to determine the intermediate step as well. Lastly, we investigated what reduction in computational cost can be achieved with the Δ-QML method combined with SchNet as neural network. This investigation suggests that (for the QM7b dataset) it is best to use the datapoints calculated with the MP2 quantum chemistry method under the numerical implementation of the cc-pVDZ basis set as intermediate step. Which is showed both by computational results and the Wasserstein distance. In total we managed to reduce the 543 days required to calculate all atomization energies on the reference level to 48 days of calculations to produce enough datapoints for training our SchNet models at chemical accuracy.

Contents

1 Introduction

Chemistry is all around around us, from producing materials, our food and developing new drugs. As of today, chemistry relies to a large degree on observations and experimentation, by trial and error. Eventually these experiments lead to new materials and molecules, such that they can be used for a new drug for instance. These experiments take much time before a chemical with the desired structure and properties is discovered. As the properties and structure of a molecule follows the laws of quantum mechanics, it might be beneficial to use a model to determine those rather than via experiments[Sch+17].

From a theoretical perspective, solutions to the exact Schrödinger equation of Quantum Mechanics are hard or impossible to obtain. Instead, approximate methods, such as the Hartree–Fock method, are used. However, even these are time consuming, especially when the size of the molecule, i.e. the number of atoms and electrons, and the number of molecules increases. Today, machine learning methods show great potential as they are able to incorporate pre-known quantum mechanical knowledge[Lu+19]. One of these machine learning methods is implemented in the SchNet package, which uses a neural network and is designed to model atomistic systems. In this thesis SchNet will be used to optimize Multilevel Δ-Quantum Machine Learning based on Neural Networks

To construct a neural network usable for predicting atomic properties with sufficient accuracy, (training)data needs to be generated. In Section 2 a concise overview of quantum mechanics, used to generate the data, will be given. Three approximate quantum chemistry methods and three basis sets for their numerical implementation are described. In this thesis, the QM7b dataset [Zas+19] will be used, consisting of 7211 different molecules, each molecule consists of a combination of at most seven of the C,N,O,S,Cl atoms. The details of this dataset are discussed in Section 3. This QM7b dataset will be used to train neural networks, which are chosen to be SchNet in this thesis (Section 4). SchNet is a neural network that is specifically designed to model atomistic systems. Therefore, SchNet is good at predicting energies and forces based on the atomic charges and positions of the atoms in the molecules. SchNet takes only the atoms and their arrangement as input values to predict the property of interest. To reach chemical accuracy for the predictions, one needs a large data set of accurate (we will define this later) data, constructed by quantum chemistry calculations at considerable cost. For instance, calculations of the data for QM7b at the highest possible accuracy take about 550 days.

To reduce the training data required, we will use a (cheap) baseline as starting point and train a neural network on the difference between the reference data and the baseline. We call this the Δ-Quantum Machine Learning method (Δ-QML method). We first consider the regular Δ-QML method, before proceeding to the multilevel Δ-QML method, based on the same principle, but takes one or more intermediate steps, with the aim to reduce the overall computational cost (Section 5).

With the multilevel Δ-QML method and SchNet as neural network, we will answer the following research questions:

- (R1) Which intermediate step results in the largest reduction of computational cost of the datapoints?

- (R2) Is it possible to use a QML-free method to rank the available intermediate steps?
- (R3) How much can we reduce the computational costs by predicting the effective averaged atomization energies instead of calculating all of them?

Note, research question R3 refers to the calculations performed with the CCSD(T) method under the cc-pVDZ basis set (see Section 2).

To answer research question R2, we describe the Wasserstein distance as a candidate to rank the available intermediate steps in a QML-free way (Section 6). In Section 7, we answer the research questions described above. By first using the Δ-QML method and Wasserstein distance to give us some insight about which level of calculation would be useful to use as intermediate step in the multilevel Δ-QML. We continue with the obtained 'best' intermediate step, to reduce the total computational cost for the datapoints significantly, by a simple trial and error method on the number of training samples used in the SchNet models. Lastly, we will summarize our findings and give recommendations for further research in Section 8. Specifically, regarding cases when the data is not already precomputed, i.e. active learning methods.

2 Background in Quantum Mechanics - a concise overview

To construct a neural network usable for predicting atomic properties with sufficient accuracy and reduced costs, it will be preferable to have data generated in multiple ways. In this section we will describe three quantum chemistry methods and describe three basis sets that are usable within the chemistry methods to obtain the datapoints. These quantum chemistry methods are all based on solving the wave function Ψ for a molecular system made out of M atoms and N electrons.

The coordinates $\mathbf{R}_\alpha$ of the individual nuclei with charges Z_α and $\mathbf{r}_i$ of the individual electrons are combined into the variables $\overline{\mathbf{R}} = (\mathbf{R}_1, \mathbf{R}_2, \ldots, \mathbf{R}_M)$ and $\overline{\mathbf{r}} = (\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_N)$, respectively, furthermore M_α denotes the mass of the nuclei. With this, the many-body Hamiltonian $\hat{H}$ reads

$$\begin{aligned}\hat{H} = &\underbrace{-\frac{1}{2}\sum_{\alpha=1}^{M} M_\alpha \Delta_{\mathbf{R}_\alpha}}_{\hat{T}_{\text{nuc}}} + \underbrace{\frac{1}{2}\sum_{\substack{\alpha,\beta=1,\\ \alpha\neq\beta}}^{M} \frac{Z_\alpha Z_\beta}{|\mathbf{R}_\alpha - \mathbf{R}_\beta|}}_{\hat{V}_{\text{nuc-nuc}}} \underbrace{-\frac{1}{2}\sum_{i=1}^{N} \Delta_{\mathbf{r}_i}}_{\hat{T}_{\text{el}}} + \underbrace{\frac{1}{2}\sum_{\substack{i,j=1,\\ i\neq j}}^{N} \frac{1}{|\mathbf{r}_i - \mathbf{r}_j|}}_{\hat{V}_{\text{el-el}}} \\ &\underbrace{-\sum_{\alpha=1}^{M}\sum_{i=1}^{N} \frac{Z_\alpha}{|\mathbf{r}_i - \mathbf{R}_\alpha|}}_{\hat{V}_{\text{nuc-el}}},\end{aligned} \tag{2}$$

where $\hat{T}$ and $\hat{V}$ are the the kinetic and potential energy operators, involving the nuclear (nuc) and electronic (el) subsystems.

The time evolution of the many-body wave function $\Psi(\overline{\mathbf{r}}, \overline{\mathbf{R}}, t)$ is obtained by solving the time-dependent Schrödinger equation [Sch26]

$$\hat{H}\Psi(\overline{\mathbf{r}}, \overline{\mathbf{R}}, t) = \mathrm{i}\frac{\partial}{\partial t}\Psi(\overline{\mathbf{r}}, \overline{\mathbf{R}}, t). \tag{3}$$

In practice however, (3) can only be solved exactly for $M = N = 1$, which corresponds to a hydrogen atom, so we will need to explore several approximations to make the problem tractable. The standard method of solving a partial differential equation such as (3) is the method of separation of variables in which one makes a product function *ansatz*, i.e., $\Psi(\overline{\mathbf{r}}, \overline{\mathbf{R}}, t) = \Phi(\overline{\mathbf{r}}, \overline{\mathbf{R}})U(t)$.

If the Hamiltonian in (2) is not explicitly time-dependent, its expectation value, the total molecular energy E_{mol}, is constant, and the time evolution of the wave function is given by $U(t) = C\exp(-\mathrm{i}E_{\text{mol}}t)$. The spatial component $\Phi(\overline{\mathbf{r}}, \overline{\mathbf{R}})$ of the wave function and the total energy are obtained as solutions of the stationary Schrödinger equation

$$\hat{H}_{\text{mol}}\Phi(\overline{\mathbf{r}}, \overline{\mathbf{R}}) = E_{\text{mol}}\Phi(\overline{\mathbf{r}}, \overline{\mathbf{R}}). \tag{4}$$

Note that in (4) both $\overline{\mathbf{r}}$ and $\overline{\mathbf{R}}$ are explicit variables of this eigenvalue problem. Since the nuclei are much heavier than the electrons, one can further assume that the electrons adjust instantaneously to the nuclear motion, i.e., the electrons move adiabatically. To express this situation in formal terms, we consider a *fixed arrangement* of nuclei $\overline{\mathbf{R}}$.

The Hamiltonian representing the electronic system that interacts with the fixed nuclear configuration reads

$$\hat{H}_{\text{el}} = \hat{H}_{\text{el}}(\overline{\mathbf{R}}) = \underbrace{\hat{T}_{\text{el}} + \hat{V}_{\text{nuc-el}}(\overline{\mathbf{R}})}_{\text{1-electron operator}} + \underbrace{\hat{V}_{\text{el-el}}}_{\text{2-electron operator}} . \tag{5}$$

In this situation, $\overline{\mathbf{R}}$ is no longer a variable of the electronic system, but a fixed parameter for the electronic degrees of freedom. The corresponding stationary electronic Schrödinger equation is given by

$$\hat{H}_{\text{el}}(\overline{\mathbf{R}})\Phi_\nu(\overline{\mathbf{r}};\overline{\mathbf{R}}) = E_\nu(\overline{\mathbf{R}})\Phi_\nu(\overline{\mathbf{r}};\overline{\mathbf{R}}), \tag{6}$$

where $\left\{\Phi_\nu(\overline{\mathbf{r}};\overline{\mathbf{R}})\right\}$ is a set of adiabatic electronic wave functions.

2.1 Hartree–Fock Theory

The electronic Schrödinger equation (6) is in practice still not solvable for many-body systems, due to the presence of the electron-electron interaction $\hat{V}_{\text{el-el}}$. Without it, the electronic Hamiltonian is simply the sum of non-interacting single-particle Hamiltonians, i.e., $\hat{H}_{\text{el}} = \sum_{i=1}^{N} \hat{h}_i(\mathbf{r}_i)$.

As $\left[\hat{h}_i(\mathbf{r}_i), \hat{h}_j(\mathbf{r}_j)\right] = 0$, the corresponding N-electron wave function Φ_ν^0 is a product of single-particle functions, also known as the Hartree product,

$$\Phi_\nu^0(\overline{\mathbf{r}}) = \prod_{i=1}^{N} \phi_{\nu_i}^0(\mathbf{r}_i) \tag{7}$$

and the total energy is given by

$$E_\nu^0 = \sum_{i=1}^{N} \varepsilon_{\nu_i}^0. \tag{8}$$

However, the Pauli principle tells us that the electronic wave functions must be antisymmetric with respect to particle exchange, and therefore must change sign whenever the coordinates of two electrons are interchanged. To meet this requirement, the electronic wave function is constructed from single-particle functions ϕ_i as a so-called Slater determinant [Sla28]:

$$\Phi(\overline{\mathbf{r}};\overline{\mathbf{R}}) = \frac{1}{\sqrt{N!}} \begin{vmatrix} \phi_1(\mathbf{r}_1) & \cdots & \phi_1(\mathbf{r}_N) \\ \phi_2(\mathbf{r}_1) & \cdots & \phi_2(\mathbf{r}_N) \\ \vdots & \ddots & \vdots \\ \phi_N(\mathbf{r}_1) & \cdots & \phi_N(\mathbf{r}_N). \end{vmatrix} \tag{9}$$

The idea of the Hartree–Fock theory is that instead of starting from predetermined single-particle functions and enforcing antisymmetry, we can start from the requirement of antisymmetry and use the variational principle to derive a set of equations that determine suitable effective single particles for the interacting case. Let us take a closer look at the

expression for the total electronic energy (suppressing the parametric dependence on $\overline{\mathbf{R}}$) for any given wave function Φ:

$$E[\Phi] = \int \Phi^*(\overline{\mathbf{r}})\hat{H}_{\mathrm{el}}\Phi(\overline{\mathbf{r}})\mathrm{d}^{3N}\overline{\mathbf{r}}. \tag{10}$$

The variational theorem states that this energy functional is minimal for the true ground-state wave function, i.e., $E[\Phi] \geq E_0$, where E_0 is the ground-state energy. Denote now Φ^{HF} as the many-body wave function in Hartree–Fock theory as a Slater-determinant ansatz. By a variational principle, the energy as a functional of the determinant approximates the true ground-state energy E_0:

$$E[\Phi^{\mathrm{HF}}] = \frac{\langle\Phi^{\mathrm{HF}}|\hat{H}_{\mathrm{el}}|\Phi^{\mathrm{HF}}\rangle}{\langle\Phi^{\mathrm{HF}}|\Phi^{\mathrm{HF}}\rangle} \geq E_0. \tag{11}$$

Minimizing the above energy functional via the effective single-particle functions ϕ_j under the constraint that they are normalized is achieved by

$$\frac{\delta}{\delta\phi_j^*(\mathbf{r})}\left\{E[\Phi^{\mathrm{HF}}] - \sum_{i=1}^{N}\varepsilon_i^{\mathrm{HF}}\left(\int \phi_i^*(\mathbf{r})\phi_i(\mathbf{r})d\mathbf{r} - 1\right)\right\} = 0 \tag{12}$$

and yields a set of equations that allow to determine the functions ϕ_j:

$$\begin{aligned}\left\{-\frac{\Delta_{\mathbf{r}}}{2} + \hat{V}_{\mathrm{ext}}(\mathbf{r}) + \int n(\mathbf{r}')v_{\mathrm{C}}(\mathbf{r},\mathbf{r}')d\mathbf{r}'\right\}\phi_j^{\mathrm{HF}}(\mathbf{r}) \\ - \int n(\mathbf{r},\mathbf{r}')v_{\mathrm{C}}(\mathbf{r},\mathbf{r}')\phi_j^{\mathrm{HF}}(\mathbf{r}')d\mathbf{r}' = \varepsilon_j^{\mathrm{HF}}\phi_j^{\mathrm{HF}}(\mathbf{r}),\end{aligned} \tag{13}$$

with $v_{\mathrm{C}} = |\mathbf{r} - \mathbf{r}'|^{-1}$. Here, we have also introduced the electronic densities

$$n(\mathbf{r}) = \sum_{i=1}^{N}\phi_i^*(\mathbf{r})\phi_i(\mathbf{r}) \qquad n(\mathbf{r},\mathbf{r}') = \sum_{i=1}^{N}\phi_i^*(\mathbf{r})\phi_i(\mathbf{r}'). \tag{14}$$

The first integral in (13) corresponds to the classical Hartree integral [Har28] of the Coulomb interaction $V_{\mathrm{H}}(\mathbf{r}) = \int n(\mathbf{r}')v_{\mathrm{C}}(\mathbf{r},\mathbf{r}')d\mathbf{r}'$, and the second integral defines the exchange potential operator $\hat{V}_{\mathrm{x}}(\mathbf{r}) = \int n(\mathbf{r},\mathbf{r}')v_{\mathrm{C}}(\mathbf{r},\mathbf{r}')\cdots d\mathbf{r}'$. The N-electron problem has thus been mapped on a set of effective single-particle problems with the Hartree–Fock potential

$$\hat{V}_{\mathrm{HF}}(\mathbf{r}) = \hat{V}_{\mathrm{ext}}(\mathbf{r}) + \hat{V}_{\mathrm{H}}(\mathbf{r}) + \hat{V}_{\mathrm{x}}(\mathbf{r}). \tag{15}$$

Considering the double counting of i and j interactions ϕ_i^{HF} and ϕ_j^{HF}, the total energy of the ground state is

$$E_0^{\mathrm{HF}} = \sum_{i=1}^{N}\varepsilon_i^{\mathrm{HF}} - \frac{1}{2}(E_{\mathrm{H}} + E_{\mathrm{x}}) \tag{16}$$

where

$$\begin{aligned}E_{\mathrm{H}} &= \int n(\mathbf{r})v_{\mathrm{C}}(\mathbf{r},\mathbf{r}')n(\mathbf{r}')\mathrm{d}\mathbf{r}\mathrm{d}\mathbf{r}' \\ E_{\mathrm{x}} &= -\int n(\mathbf{r},\mathbf{r}')v_{\mathrm{C}}(\mathbf{r},\mathbf{r}')n(\mathbf{r}',\mathbf{r})d\mathbf{r}d\mathbf{r}'.\end{aligned} \tag{17}$$

In summary, the Hartree–Fock theory assumes that the many-electron wave function takes the form of a Slater determinant. Since the exact wave functions cannot be expressed as single determinants, the problem with this assumption is that Hartree–Fock methods cannot fully represent the solution of the exact many-electron Schrödinger equation (6) and the corresponding total energy differs from the true ground-state energy. This difference is often referred to as *correlation* energy. There are multiple methods that aim to improve on this Hartree–Fock method. These methods improve the accuracy of the solutions to the Schrödinger equation, however that is paired to a larger computational cost to obtain the solutions. Furthermore, the accuracy and cost depends on the basis set that is used to describe the waves of the electrons. We will discuss two of the post-Hartree–Fock methods and three different basis sets in the remainder of this section.

2.2 Møller–Plesset perturbation theory

The Møller–Plesset (MP) perturbation theory enhances the Hartree–Fock method by adding electron correlation effects by using the Rayleigh–Schrödinger perturbation theory. This theory adds a small disturbance $\hat{P}$ to an unperturbed Hamiltonian $\hat{H}_0$:

$$\hat{H} = \hat{H}_0 + \lambda \hat{P}. \tag{18}$$

Here, λ is a real parameter, which determines the size of the disturbance added to the Hamiltonian. In the case of the Møller–Plesset perturbation, the unperturbed Hamiltonian $\hat{H}_0$ is given by:

$$\hat{H}_0 = \sum_{i=1}^{N} f_i. \tag{19}$$

Where f_i denotes the Fock operator for the i-th electron in the system and is defined in terms of the exchange operator $K_j(i)$, which defines the quantum effect produced by exchanging two electrons, and Coulomb operator $J_j(i)$, which defines the repulsive force between two electrons:

$$f_i = h_i + \sum_{M_o} 2J_{M_o}(i) - K_{M_o}(i) \tag{20}$$

where we only sum over the occupied molecular orbitals.

The Hartree–Fock ground state wavefunction Φ^{HF} is an eigenfunction of $\hat{H}_0$ with an eigenvalue $E_0^{(0)}$. The perturbation $\hat{P}$ is taken as:

$$\hat{P} = H - \hat{H}_0, \tag{21}$$

with H being the electronic Hamiltonian (Equation 5).

This perturbation determines the first-order energy correction $E_0^{(1)}$, as the energy correction is equal to the expectation value of the perturbation. By adding $E_0^{(0)}$ and $E_0^{(1)}$, we end up with the result of the Møller–Plesset perturbation theory. Note that $E_0^{(0)}$ is the same as E_0^{HF} in Equation 16. Therefore, the Møller–Plesset theory adds a correction to the Hartree-Fock method based on the perturbation. There are different variants of this method available, dependent on the order of the correction that is added. For example, MP2 includes the second-order energy correction in the ground state energy calculation.

This second order energy correction $E^{(2)}$ is defined as follows:

$$E^{(2)} = \sum_{J \neq 0} \frac{\langle \Phi_J | H^{(1)} | \Phi_0 \rangle \langle \Phi_0 | H^{(1)} | \Phi_J \rangle}{E_0^{(0)} - E_J^{(0)}}. \tag{22}$$

Where, $E_J^{(0)}$ is the eigenvalue of Φ_J, a multiply excited determinant and a eigenfunction of $\hat{H}_0$. The method can be extended to include third order (MP3), fourth order (MP4) or even higher orders for the energy correction. Resulting in better results for the Hartree–Fock equations, but also becoming more complicated to perform the calculations.

There are advantages and drawbacks to the Møller–Plesser method. Clearly it enhances the results with comparison to the Hartree–Fock method, as the method is designed tackle one of the weaknesses of the Hartree–Fock method. However, this enhancement comes at a computational cost to calculate the perturbation and higher-order energy corrections. Furthermore, the method lacks the infinite-order effects of the coupled cluster method, which we will discuss in the next subsection, meaning the latter method will be more accurate than the Møller–Plesset method.

2.3 The coupled-cluster method

The coupled-cluster (CC) method relates the exact electronic wavefunction Φ to the Hartree–Fock wavefunction Φ_0 by a cluster operator C:

$$\Phi = e^C \Phi_0, \tag{23}$$

where the exponential e^C is defined by the series expansion:

$$e^C = 1 + C + \frac{1}{2}C^2 + \frac{1}{3!}C^3 \dots. \tag{24}$$

Cluster operator C is equal to the sum $\sum_{i=1}^{N} C_i$, where C_i is the effect of the i-electron excitation operator, where

$$C_1 \Phi_0 = \sum_{a,p} t_a^p \Phi_a^p, \quad C_2 \Phi_0 = \sum_{a,b,p,q} t_{ab}^{pq} \Phi_{ab}^{pq}, \quad \dots \tag{25}$$

are satisfied. Dependent on the variant of the coupled-cluster method, we know which excitation operators C_i are exactly used. Most common are the abbreviations S, D, T and Q to distinguish respectively between single, double, triple and quadruple excitations. Meaning that CCDT stands for the coupled cluster method where C is given by $C_2 + C_3$.

Another variant, CCSD(T), is used to obtain datapoints within the QM7b dataset, on which we will elaborate in Section 3. In the name of the method, the brackets around 'T' mean that an estimate is used for the connected triples. So in CCSD(T), the cluster operator C consists of C_1, C_2 and C_3, however the contribution of the triple excitations is calculated non-iteratively using perturbation theory.

2.4 Numerical implementation and basis sets

To represent the molecular orbitals in the Hartree–Fock method exactly, we need a complete set of basis functions χ to perform the calculations on. However, this is not computationally feasible due to the basis set being infinite. Therefore, it is important to use a

basis set that minimizes the number of two-electron integrals to evaluate by keeping the amount of basis functions low; and choose these basis functions smart to minimize the cost of calculating the integrals.

One choice for the basis functions are the Slater-type orbitals (STO), which depend on the nucleus of the atom with atomic number Z and on quantum numbers n,l and m_l:

$$\Psi_{nlm_l}(r,\theta,\phi) = Nr^{N_{\text{eff}}-1}e^{-Z_{\text{eff}}\rho/n_{\text{eff}}}Y_{lm_l}(\theta,\phi). \tag{26}$$

Where N is a normalization constant, $\rho = r/a_0$ and Y_{lm_l} is a spherical harmonic. Furthermore, n_{eff} and Z_{eff} are respectively the effective principal quantum number and the effective nuclear charge. A complete basis set consists of all possible STOs (see section 9.4 in), which is infeasible to use, so a basis set consists usually only of a small number of STOs. However, for the Hartree–Fock method such a basis set is still impractical to use, because of the amount of two-electron integrals.

To reduce the number of two-electron integrals, Guassian-type orbitals (GTO) were introduced. These Gaussian orbitals are of the form:

$$g_{ijk}(\mathbf{r}) = Nx^i y^j z^k e^{-ar^2}. \tag{27}$$

Here $\mathbf{r}$ represent the coordinates of the nucleus and N is a normalization constant. Furthermore, α is a positive exponent and lastly i,j,k are non-negative integers. There are different types of these Gaussian orbitals, dependent on the sum of the values for i,j,k. If $i+j+k=0$, then we have s-type orbitals; when they sum to one, we are using p-type orbitals; and so on. The advantage of GTOs is that the product of two Gaussians can be written as one Gaussian. Reducing the computational costs to evaluate the two-electron integrals. Despite the upside of the Gaussian orbitals, they need a larger basis set to represent the orbitals around the nucleus to achieve the same accuracy as STOs. To reduce the computational cost of the expanded basis set, we can group several GTOs together by taking a linear combination of them. This grouping of GTOs is known as 'contracted Gassian functions'.

Clearly, there are multiple possibilities when choosing which GTOs should contract with each other. Let n_g denote the number of Gaussians in the initial basis set. By using a least square fit of the n_g Gaussians to a set of, already optimized, STOs. The basis set consisting of these n_g optimized Gaussians is then used as the STO-n_gG basis set. In the dataset used in this paper (see Section 3) STO-3G, the variant with $n_g = 3$, is one of the available basis sets. Another basis set based on contracting Gaussians is the 6-31G basis set. In this contraction method the orbital of the inner shell and the valence shell are contracted differently. For the inner shell, a single contracted Gaussian, composed of six primitive orbitals, is used. For the valence shell the orbital is given by one contracted Gaussian consisting of three original orbitals, with the addition of a single diffuse primitive. This basis set is more accurate than the STO-3G basis set, however the number of basis set functions increases quickly when used on larger molecules. As a consequence, the computational time becomes larger than with the STO-3G basis. Lastly, we will explain the most accurate basis set available in the dataset, cc-pVDZ. This basis set extends on the 6-31G basis set, by using twice as many basis functions for the valence shells. Thus the amount of basis functions increases with respect to the 6-31G basis set, resulting in an increase of computational cost.

3 Overview of the QM7b dataset

The research performed in this report is on the molecules in the QM7b dataset. It consists of 7211 molecules, where each molecule consists of a maximum combination of at most seven C, N, O, S, Cl atoms, saturated with H atoms. In other words the molecules consist of C, N, O, S, Cl, H atoms with seven or less 'heavy' atoms and enough H atoms to satisfy the preferable amount of bonds for each atom. As an example, in Figure 1, one of the molecules in the dataset is shown. This molecule adheres the constraints to belong to the QM7b dataset, since there are only five 'heavy' atoms: one nitrogen atom and four carbon atoms. Furthermore, the hydrogen atoms ensure that each carbon atom has four bonds and the nitrogen has three bonds.

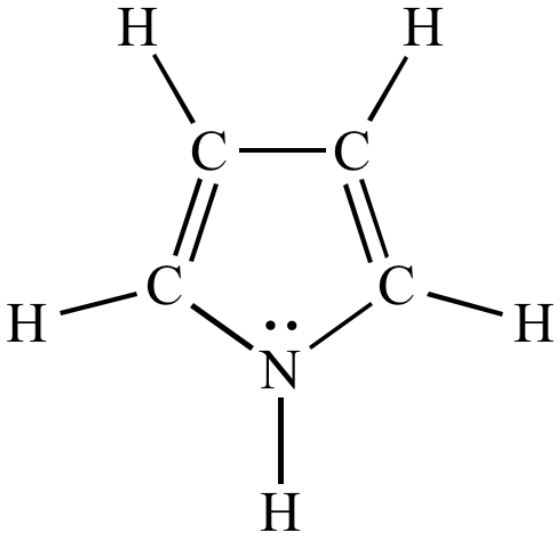


Figure 1: Example of a molecule in QM7b dataset.

For these 7211 molecules, the 'effective averaged atomization energies' and 'total energies' are calculated with nine different levels of accuracy. We will refer to these nine levels as the nine 'levels of theory', which consist of a combination of one of the basis sets and quantum chemistry methods described in Section 2. For the basis set, we have the option between STO-3G, 6-31G and cc-pVDZ and for the quantum chemistry method between HF, MP2 and CCSD(T). In Table 1 these combinations are shown with their notation.

	basis set			
↑	cc-pVDZ	$D_{H,c}$	$D_{M,c}$	$D_{C,c}$
	6-31G	$D_{H,6}$	$D_{M,6}$	$D_{C,6}$
	STO-3G	$D_{H,s}$	$D_{M,s}$	$D_{C,s}$
		HF	MP2	CCSD(T)
		quantum chemistry method		
		→		

Table 1: The nine combinations of basis sets and methods with their notation, where the arrows indicate an increase in accuracy and computational cost.

As mentioned in Section 2, the accuracy and cost of the calculations needed to generate the data depend on the method and basis set. The arrows, next to the table above, indicate an increase in computational cost to obtain the datapoints, accordingly the accuracy of the datapoints also increases. Therefore, $D_{C,c}$ will be used as the target value in the Δ-QML model. We discuss the computational costs of these methods in more details in Section 5 (Figure 3). In the rest of this paper we denote the effective averaged atomization energy of a molecule m calculated at level $D_{q,b}$ by $E_{q,b}(m)$ for $q \in \{H, M, C\}$ and $b \in \{s, 6, c\}$. The effective averaged atomization energy of molecule m is defined as:

$$E(m) = E_{\text{total}}(m) - \sum_{I \in m} n_I(m) \cdot e_I(m), \tag{28}$$

where $n_I(m)$ is the number of atom I in molecule m and $e_I(m)$ is the effective atomic energy of atom I. The values $e_I(m)$ are determined by a linear fit of $E_{\text{total}}(m) = \sum_{I \in m} n_I(m) \cdot e_I(m)$ over all molecules m in the dataset. The total energies E_{total} are calculated as described in Section 2 and thus dependent on the combination of quantum chemistry method and basis set. Note that in the remainder of this report, we denote the effective averaged atomization also with just 'atomization energy'.

The dataset can be found as ZIP at [Zas+19]. The ZIP file is placed under the 'Supporting Info' tab and can be freely downloaded from there.

4 Description of SchNet

In order to predict the target values $D_{C,c}$ based on a representation of the molecules, a neural network will be used. There are a lot of different networks available, but in this thesis SchNet will be used. This choice is largely based on the fact that SchNet is specifically designed to model atomistic systems. Therefore, SchNet is good at predicting energies and forces based on the atomic charges and positions of the atoms in the molecules. SchNet is freely available as option within the 'SchNetPack' package on GitHub[Sch]. The description of SchNet that follows in the rest of this section is largely based on the description of the inventors, found in the papers about SchNet[Sch+18] and SchNetPack[Sch+19]. For further details about the possibilities within SchNetPack and the results that they achieve, we refer you to these papers.

Neural networks consist of 2 components: a representation block and a prediction block. Here representation means: the way that the available data gets stored to be usable within the neural network. An example of a representation for neural networks can be the Coulomb Matrix $\mathbf{C}$[Rup+12]. This matrix depicts the interaction between atoms in the following way:

$$\mathbf{C}_{i,j} = \begin{cases} 0.5 Z_i^{2.4} & \text{for } i = j \\ \frac{Z_i \cdot Z_j}{R_{i,j}} & \text{for } i \neq j \end{cases},$$

where Z_i is the atomic charge Z of atom i and $R_{i,j}$ the distance between atoms i and j. In the case of the Coulomb Matrix, the representation block will simply consist of the calculations of the values in each of the matrices. A neural network can be trained on the available molecules based on the values in the Coulomb Matrix. The Coulomb Matrix is easy and fast to calculate, however it has some drawbacks. One of them is the lack of uniqueness, i.e. the same molecule can have multiple valid Coulomb Matrices, depending on the ordering of the atoms.

Another variant for the representation block is used within SchNet: since SchNet is an end-to-end architecture, both the representation and the prediction block will be trained on the data. So there is no predetermined representation like the Coulomb matrix, but a flexible one which adapts to the available data. Therefore, SchNet trains to predict a predetermined property P_m of molecule m, consisting of n atoms, by only using nuclear charges $Z^m = (Z_1^m, \ldots, Z_n^m)$ and positions $R^m = (\mathbf{r}_1^m, \ldots, \mathbf{r}_n^m)$ as input, where n is the number of atoms in molecule m.

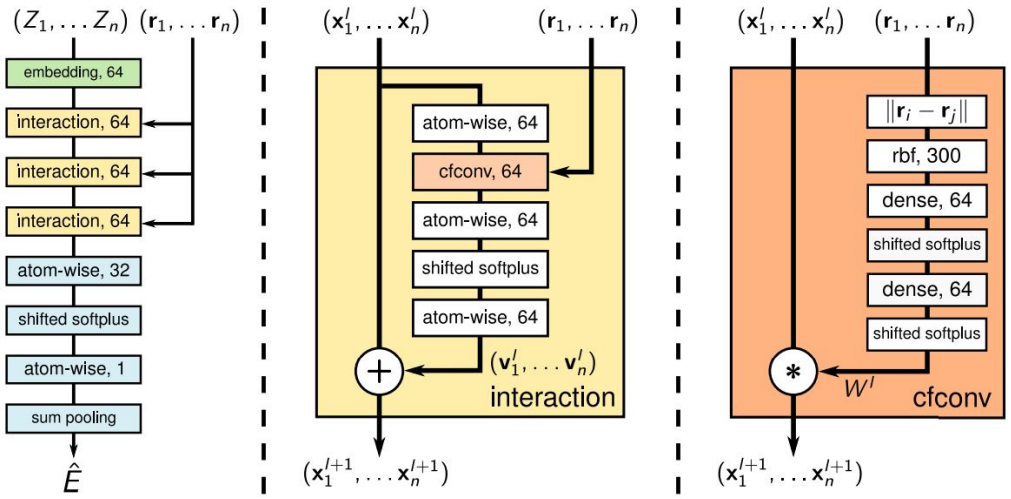


Figure 2: Overall architecture of SchNet[Sch+18].

Figure 2 shows the architecture of the neural network SchNet. The network consists of a few building blocks that reappear multiple times throughout the model. When we look at the left hand side of Figure 2, we see the overall structure of SchNet that starts with an embedding layer to initialize the representation block. Next, a predetermined number of interaction blocks will be used. These interaction blocks are displayed in more detail in the middle part of the figure. Each interaction block consists of continuous-filter convolutions, for which its composition is shown on the right hand side in Figure 2. The process of training ends with sum pooling, which simply sums the atomic contributions, in order to get to the final prediction.

SchNet represents the atoms of molecule m in each layer by a feature vector $\mathbf{x}$ of a preset size F per layer. Let molecule m consist of n atoms, with nuclear charges Z and positions R. Furthermore, let $X^l = (\mathbf{x}_1^l, \dots, \mathbf{x}_n^l)$ be the representation of the n atoms at layer l. This representation is just a way to document the values of the different neurons in the network in a convenient way. The number of features F is simply the number of used neurons. Note that the numbers in Figure 2 are the number of output neurons in that layer and hence having the same lengths as the feature vectors $\mathbf{x}_i$, F. Next, the interaction of the different kind of layers on the representation vectors is explained in more detail.

Embedding layer: This layer is the initializing layer of SchNet and is solely based on the atomic charges Z_i:

$$\mathbf{x}_i^0 = \mathbf{a}_{Z_i}, \ \forall i \in \{1, \dots, n\}.$$

The embeddings $\mathbf{a}_Z$ are initialized at random and optimized during training. Note that if atoms i and j are the same, the feature vectors for these vectors will also be the same, i.e. if $Z_i = Z_j$, then $\mathbf{x}_i^0 = \mathbf{x}_j^0$.

Dense layer: This layer, as the name indicates, is a dense layer. Dense layers in neural networks are layers that are fully connected to the previous layer. Hence, the output vector $\mathbf{x}_i^{l+1}$ is determined by all entries of feature vector $\mathbf{x}_i^l$. This is simply done by a weighted sum of the entries with the addition of an additional bias $\mathbf{b}$:

$$\mathbf{x}_i^{l+1} = \mathbf{x}_i^l W^T + \mathbf{b}^l, \ \forall i \in \{1, \dots, n\}.$$

Atom-wise layer: These layers share a lot of similarities with the dense layers. However, they differ in the way that weights are applied to the entries of $\mathbf{x}_i^l$. Instead of the weighted sum used in the dense layers, a weighted function is used. Meaning that $\mathbf{x}_i^{l+1}$ is obtained by a function, which takes all entries of $\mathbf{x}_i^l$ as input. Furthermore, a bias b is added for each neuron, giving us the following result for x_i^{l+1}.

$$\mathbf{x}_i^{l+1} = W^l \mathbf{x}_i^l + \mathbf{b}^l, \ \forall i \in \{1, \dots, n\}.$$

Note, W^l is a function and not a matrix. These functions and their biases are fixed within each layer l, which ensures that SchNet remains scalable with respect to the number of atoms in molecule m.

Shifted softplus: This layer simply applies an activation function to the output of the previous layer. More concretely, the shifted softplus function is used as activation function:

$$ssp(\mathbf{x}_i^l) = \ln\left(0.5e^{\mathbf{x}_i^l} + 0.5\right).$$

The shift in the function makes sure that $ssp(0) = 0$, which enhances the convergence of SchNet.

Sum pooling: This is the last step in the structure of SchNet and results in the prediction for molecule m. Before this step the number of features is reduced by the use of several atom-wise layers. As shown in Figure 2 the initial number of features (64) is first reduced to 32 and then to only one. So for last layer L we end up with a feature vector $X = (\mathbf{x}_1^L, \dots, \mathbf{x}_n^L)$, where x_i^L is only a number and not a vector anymore. Applying sum pooling on this feature vector, simply means summing all the x_i:

$$\widehat{E} = \sum_{i=1}^{n} x_i^L.$$

The SchNet structure consists of a combination of all these different layers, which occur in the different building blocks of SchNet. As said before, SchNet produces its prediction by training the representation and the prediction blocks. The representation block gives the intermediate feature vector X^l after all the interaction blocks. After which the prediction block proceeds by transforming this feature vector into the eventual prediction. On the left hand side of Figure 2, the green and yellow blocks are the components of the representation block and the blue parts form the prediction block.

This leaves us with the explanation of the **continuous-filter convolutional layers** (cfconv). These layers are specifically designed by the developers of SchNet to handle atomic interactions within the molecules. To achieve this, the convolution between atom i and its surrounding atoms is taken into account. Let us denote the set of neighbours of atom i by $nbh(i)$. Then convolution of atom i in SchNet becomes:

$$\mathbf{x}_i^{l+1} = \sum_{j \in \{nbh(i)\}} \mathbf{x}_i^l \odot W^l(||\mathbf{r}_j - \mathbf{r}_i||).$$

Here W^l is a filter-generating network, that maps atomic distances to filter values, i.e. $W^l : \mathbb{R} \mapsto \mathbb{R}^F$. Furthermore, '$\odot$' is the element wise multiplication between $\mathbf{x}_i$ and W^l.

Therefore, the convolutions are all calculated feature-wise, which is computationally more efficient. Note that cross-feature processes are performed by the atom-wise layers.

This filter-generating network can be used to incorporate chemical knowledge into the system. For example, known invariances of molecules are included into SchNet. More precisely the model is rotational invariant by using the pairwise distances $||\mathbf{r}_j - \mathbf{r}_i||$ between atoms, instead of the relative positions of the atoms.

5 Δ-QML method

In this section we give an overview of the computational costs per combination of chemistry method and basis set. With the computational costs at hand, we will describe the role of Δ-QML in this project. 'regular' QML, Δ-QML and multilevel Δ-QML and their differences will be described in this section. Furthermore, we will explain our choice for using multilevel Δ-QML in this project.

5.1 Δ-QML

As mentioned before, SchNet will be used as neural network to eventually predict the reference values $D_{C,c}$ from the input of the molecular structure. Additionally, we want to achieve chemical accuracy ($\approx$ 1kcal/mol) as cheap as possible, since the goal is to reduce the computational costs for generating high-quality data. To achieve the reduction in computational cost, we want to use inexpensive data as predictor for the 'effective averaged atomization energies', instead of having to perform the high cost data calculations. Therefore, we will first look at the cost bound to the different combination of basis set and method. Note that the cost is measured as the computation time (CPU) in seconds. We split the dataset in 7 groups, where each element in a group is determined by the number of 'heavy' atoms and denote them by CONSCli. Here i denotes the number of 'heavy' atoms and ranges from one to seven, for example CONSCl2 is the subset containing all the molecules with exactly 2 non-hydrogen atoms. The distinction between the different CONSCl levels is logical to make, as the calculations for the datapoints largely depend on the number of 'heavy' atoms.

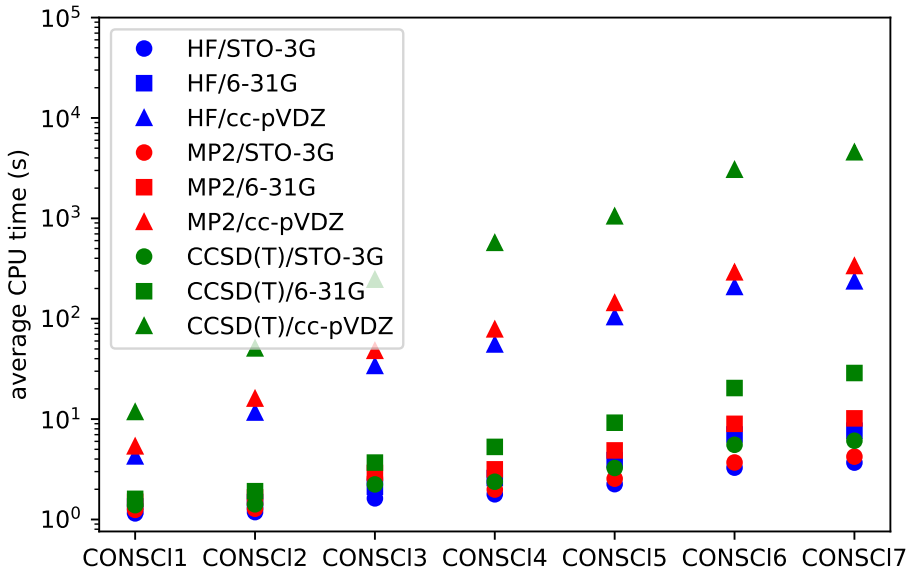


Figure 3: Average computational time per combination of chemical method and basis set, depends on the number of 'heavy' atoms.

The figure above shows that datapoints on $D_{C,c}$ level are extremely expensive with respect to the other combinations of basis set and method. On the other hand datapoints on $D_{H,s}$ are as expected the cheapest to obtain. Note that the difference in basis set influences the computational costs more than the difference in the chemistry method. To analyze the reduction in costs later in this report, we assume that the computational costs are on a linear line with respect to the CONSCl level, when a log-scale y-axis is used, i.e. the computational costs follow an exponential curve (with base 10). With this assumption, we have an average cost (a_c) per CONSCli level for calculating the atomization energy. To make the analysis of the results easier, we take the averaged sum of a_c over the number of molecules in each CONSCl level. As a result we have a single value as computational cost, shown in Table 2. Note that this procedure is repeated for each combination of chemistry method and basis set individually. The advantage of using a single cost value for all molecules is that we do not have to check which datapoints are exactly in the training's samples for the SchNet model.

$D_{H,s}$	$D_{H,6}$	$D_{H,c}$	$D_{M,s}$	$D_{M,6}$	$D_{M,c}$	$D_{C,s}$	$D_{C,6}$	$D_{C,c}$
3.5	7.5	322.0	4.1	9.9	462.2	5.9	26.3	6509.4

Table 2: The nine different combinations of chemistry method and basis with respect to their computational cost in seconds.

Using the value for $D_{C,c}$ in the table above, we calculate the total computational cost: $6509.4 \cdot 7211 = 46939283.4$ seconds, which is over 543 days, to obtain the atomization energies with the highest accuracy. To reduce the computational time, we want to predict atomization energies by using the help of only a subset of the available data. By default, we can use SchNet to predict the atomization energy directly. Meaning we train the neural network from a single level of datapoints. Obviously, it will give the best results if the training data is obtained on the same level as the target data, i.e. on the $D_{C,c}$ level. However, the amount of expensive datapoints is then entirely determined by the training size. Let $S_{C,c}^{N,v}$ denote a trained SchNet model on level $D_{C,c}$ of training size N and validation set of size v. Then we denote the predicted atomization energy $\widehat{E}_m$ for molecule m by:

$$\widehat{E}_m \approx S_{C,c}^{N,v}(m).$$

To limit the training size, we will use more initial information before we start training the neural network. One way to achieve this, is to use a baseline as starting point and train a neural network on the difference between the reference data and the baseline. Since we train the model on differences of quantum mechanical properties, we call this the Δ-Quantum Machine Learning method or Δ-QML method in short. As the baseline we can for instance take the energies calculated at $D_{H,s}$ level, but another combination between basis set and chemistry method can form the baseline as well. Let $\Delta_{H,s}^{C,c}(m)$ denote the atomization energy difference $E_{C,c}(m) - E_{H,s}(m)$ for molecule m. Note that the difference can become negative when $E_{H,s}$ is larger than $E_{C,c}$. Let $S[\cdot]^{N,v}$ denote a trained SchNet model on the differences, where N, v are respectively the training and validation sizes. This gives us the following notation for the predicted atomization energies for molecule m:

$$\widehat{E}_m \approx E_{H,s}(m) + S[\Delta_{H,s}^{C,c}(m)]^{N,v}.$$

For both these methods, the direct ML and the Δ-QML, we can use a SchNet model

to see how the learning curves differ from each other. Figure 4 shows these learning curves over the number of training samples N.

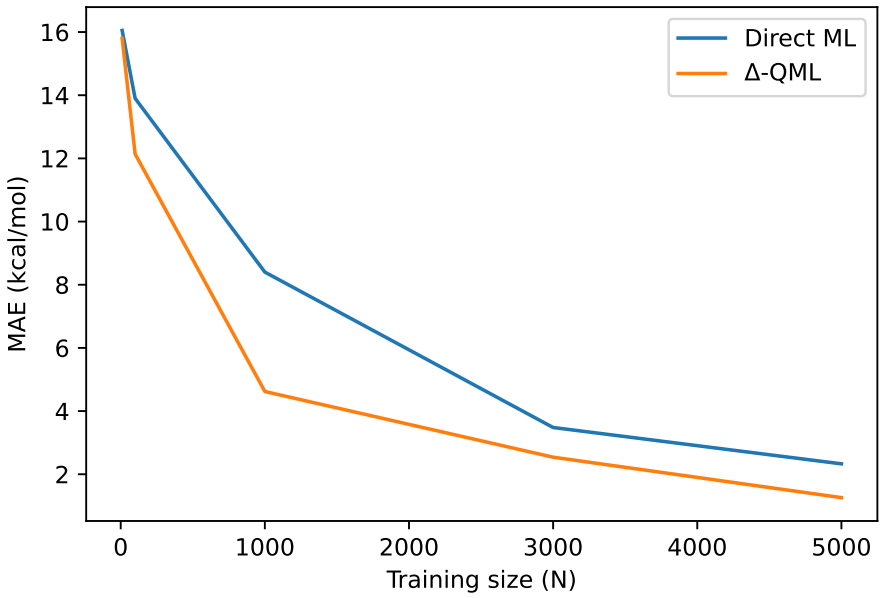


Figure 4: Learning curves of the direct QML and Δ-QML methods.

One can notice that the learning curve of the Δ-QML method starts with a steeper slope, meaning that the MAE on the predictions decreases faster on the training size than with the direct ML method. As an example, to get below a MAE of 4 kcal/mol the direct ML method needs 3000 datapoints and the Δ-QML method needs only 2000. So the Δ-QML method already improves on applying a SchNet model directly to the data, however we can improve on the Δ-QML method as well. Despite the addition of pre knowledge in the form of a baseline to start from, which reduces the amount of training data N to achieve the same mean absolute error (MAE), we remain at the point that all training data uses its atomization energy at $D_{C,c}$ cost to calculate the necessary differences. To move away from the restriction that all needed training samples use their atomization energy at $D_{C,c}$ level, we will use a multilevel Δ-QML method.

5.2 Multilevel Δ-QML

The multilevel Δ-QML method is based on the same principle as the Δ-QML method, but takes one or more intermediate steps. Let us consider only one intermediate step for now. Again we can start with baseline $D_{H,s}$ and reference values $D_{C,c}$. For simplicity, let us denote the intermediate level with i, which is one of the 7 remaining combinations in Table 1. As we have one intermediate level, we end up with 2 differences. Namely, the difference between the baseline and the intermediate calculations and between the

intermediate level and the reference data:

$$\Delta^i_{H,s}(m) = E_i(m) - E_{H,s}(m),$$
$$\Delta^{C,c}_i(m) = E_{C,c}(m) - E_i(m).$$

Let S_1 denote the SchNet model trained on the first distances $\Delta^i_{H,s}$ and S_2 the neural network subject to the second distances $\Delta^{C,c}_i$ with respectively N_1, N_2 as training sizes and v_1, v_2 as validation set sizes. Then the prediction for the atomization energy of molecule m becomes:

$$\widehat{E}_m \approx E_{H,s}(m) + S_1[\Delta^i_{H,s}(m)]^{N_1,v_1} + S_2[\Delta^{C,c}_i(m)]^{N_2,v_2}.$$

Comparing this to the single Δ-QML method, we hope to have a more accurate baseline: $E_{H,s}(m) + \Delta^i_{H,s}$ and less irregular $\Delta^{C,c}_i$, since the computations on D_i level are more accurate to train a neural network on. Both these advantages result in N_2 being smaller than N, therefore, needing less samples from the expensive $D_{C,c}$ level. However, we need at least $\max\{N_1, N_2\}$ training samples on the D_i level of accuracy. If we require that both SchNet models are trained on (fully) independent datapoints, we need $N_1 + N_2$ samples on D_i level. Whereas, by creating as much overlap between the samples, we only have a training size of $\max\{N_1, N_2\}$ at that accuracy.

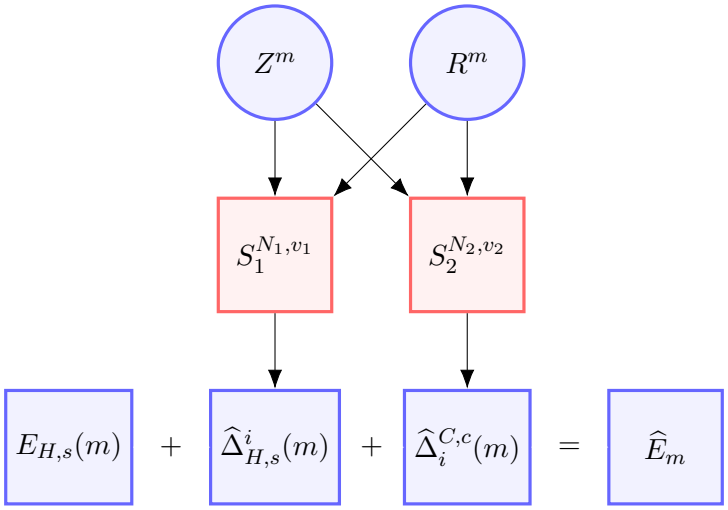


Figure 5: Overview of how the multilevel Δ-QML method arrives at its predictions.

The figure above shows an overview of how we use SchNet in combination with the Δ-QML method. The red blocks in Figure 5 are trained SchNet models on the respective distances. These models take the atomic charges Z^m and atomic positions R^m of molecule m as input (blue circles) to get predictions $\widehat{\Delta}^i_{H,s}$ and $\widehat{\Delta}^{C,c}_i$ for molecule m. These predictions are summed together with the available atomization energy at $D_{H,s}$ level to get to the final value $\widehat{E}(m)$.

In conclusion, we will use the multilevel Δ-QMl method in combination with SchNet to predict the 'effective averaged atomization energy' of a molecule with the intention to reduce the computational costs associated with generating the atomization energies on a high accuracy level. With the data that is available to us, we have different possibilities for the intermediate step in the multilevel Δ-QML method. In the following sections of

this report we investigate, which intermediate step performs the best and whether we can use the Wasserstein distance as machine learning-free ranking of the intermediate step. Furthermore, we will determine the computational reduction one can achieve by using the combination of SchNet and the multilevel Δ-QML method, instead of applying the CCSD(T) method with cc-pVDZ basis to all the molecules.

6 Wasserstein distance

As mentioned, we want to achieve chemical accuracy with the lowest computational cost for the datapoints. Therefore, we need to determine which intermediate step will be the best one. As we train neural networks on differences, we basically are training the networks to transport data from one distribution to another one. This transport of data has a lot of similarities to an optimal transport problem, as our SchNet models basically try to transport the distribution of the baseline to the distribution of the intermediate step with the least amount of work and the same for the distribution of the intermediate step to the reference data. In Figure 6, the histogram of the available data on $D_{H,s}$ level and the kernel density estimate (kde) on $D_{C,c}$ level are shown. Clearly the kde does not belong to the histogram, since they are not perfectly on top of each other. However, both $D_{H,s}$ and $D_{C,c}$ represent the effective averaged atomization energies for the same molecules. Thus we would like to adjust the histogram on $D_{H,s}$ level in such way that it represents the values of the $D_{C,c}$ level, i.e. shift the histogram to match the kde on $D_{C,c}$ level.

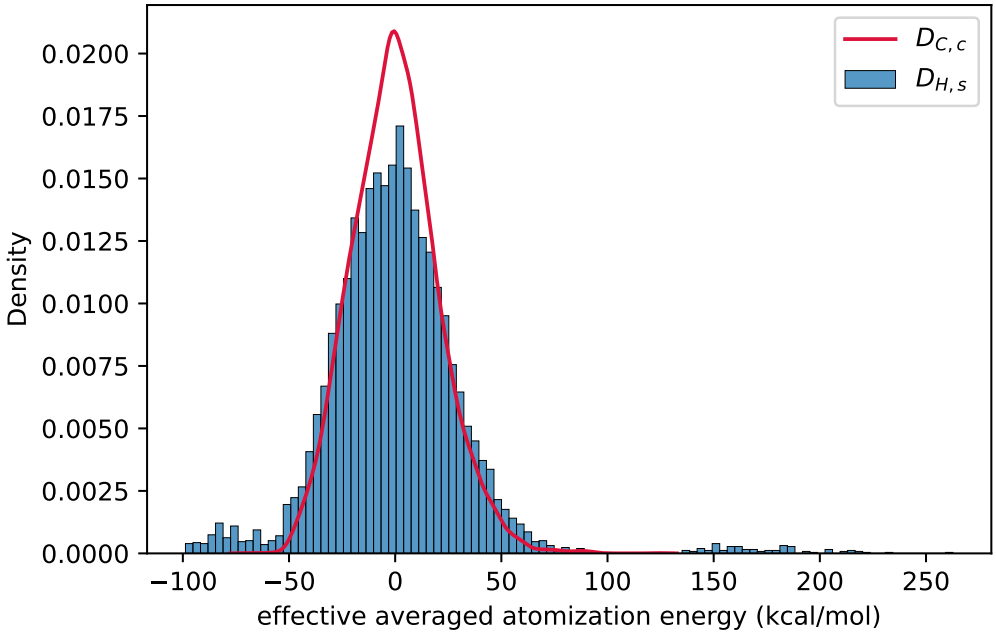


Figure 6: Histogram of the atomization energies calculated on $D_{H,s}$ and kernel density estimate of the atomization energies calculated on $D_{C,c}$ level.

This gives us 2 options to determine the best intermediate step. Firstly, we can indeed see it as an optimal transport problem and use some metric from that field to derive a choice. Secondly, we can train SchNet on each of the remaining options and investigate which intermediate step gives the best results, i.e. trial and error. Let us explain the more theoretical option: the optimal transport problem. As explained above, the optimal transport problem arises when one wants to move one distribution onto another distribution while minimizing the amount of work. This can either be performed discrete-wise (on histograms) or continuously (on densities). Clearly our data is not continuous, so we will focus on the discrete case.

We can see each histogram as a probability vector $\mathbf{a}$, where its length corresponds to the number of bins in the histogram and the value of $\mathbf{a}_i$ is equal to the probability of an observation being in bin i. This probability is just the ratio between the frequency of observations being in bin i and the total amount of observations. Clearly $0 \leq \mathbf{a}_i \leq 1 \ \forall i$ and $\sum_i \mathbf{a}_i$=1, hence $\mathbf{a}$ is indeed a probability vector. Now we let $\mathbf{a}$ denote the probability vector corresponding to the histogram we want to transport to a second histogram with probability vector $\mathbf{b}$. Note that both probability vectors not necessarily need to have the same length, since the histograms could consist of a different number of bins. For now we assume that vectors $\mathbf{a}$ and $\mathbf{b}$ respectively have k and l entries.

Furthermore, we define matrix $\mathbf{C}$ as the cost matrix, where $\mathbf{C}_{i,j}$ denotes the cost to move from $\mathbf{a}_i$ to $\mathbf{b}_j$ and matrix $\mathbf{P}$ as the coupling matrix, where $\mathbf{P}_{i,j}$ denotes the amount of mass flowing from $\mathbf{a}_i$ to $\mathbf{b}_j$. With this coupling matrix we can define the admissible transportations by $\mathbf{U}(a, b)$:

$$\mathbf{U}(\mathbf{a}, \mathbf{b}) = \{\mathbf{P} \in \mathbb{R}_+^{k \times l} | \sum_j (\mathbf{P}_{i,j}) = \mathbf{a} \text{ and } \sum_i (\mathbf{P}_{i,j}) = \mathbf{b}.\}$$

Hence, $\mathbf{U}(\mathbf{a}, \mathbf{b})$ is the set of possible couplings that move vector $\mathbf{a}$ to $\mathbf{b}$. Using this notation we can construct Kantorovich's optimal transport problem:

$$\mathbf{L}_\mathbf{C}(\mathbf{a}, \mathbf{b}) = \min_{\mathbf{P} \in \mathbf{U}(\mathbf{a}, \mathbf{b})} \sum_{i,j} \mathbf{C}_{i,j} \mathbf{P}_{i,j}.$$

Using Kantorovich's description of the optimal transport problem, we derive the p-Wasserstein distance in the following way: We assume that probability vectors $\mathbf{a}$ and $\mathbf{b}$ have the same length n. Furthermore, let the cost matrix $\mathbf{C}$ be equal to $\mathbf{D}^p$ for some distance matrix $\mathbf{D}$. Where matrix $\mathbf{D}$ is a distance matrix if the following three properties hold:

$$\begin{aligned}
&\mathbf{D} \in \mathbb{R}^{n \times n}, \\
&\mathbf{D}_{i,j} = 0 \leftrightarrow i = j, \\
&\mathbf{D}_{i,k} \leq \mathbf{D}_{i,j} + \mathbf{D}_{j,k}, \forall (i, j, k) \in \{0, \ldots, n.\}^3
\end{aligned}$$

Then the p-Wasserstein distance[PC20] $\mathbf{W}_p(\mathbf{a}, \mathbf{b})$ is defined as:

$$\mathbf{W}_p(\mathbf{a}, \mathbf{b}) = \mathbf{L}_{\mathbf{D}^p}(\mathbf{a}, \mathbf{b})^{1/p}.$$

7 Results

In this section we will answer the research questions stated in the introduction. We will start with the ordering of intermediate steps that the Wasserstein distance gives us. Furthermore, we will investigate which intermediate step in the multilevel Δ-QML method is the most applicable for us. We proceed by using this intermediate step in a more extensive SchNet model to see what cost reduction we can achieve. Lastly, we will investigate the possibility to use sub-samples of the data in the Wasserstein distance.

7.1 Δ-QML

Let us start with the Δ-QML method to give us some insight about which level of calculation would be useful to use as intermediate step in the multilevel Δ-QML. First let us look at what the 1-Wasserstein distance is between the different level of calculations. We calculate these distances with respect to $D_{C,c}$:

cc-pVDZ	0.59	0.36	0
6-31G	3.14	2.55	2.05
STO-3G	6.68	4.90	4.49
	HF	MP2	CCSD(T)

Table 3: The 1-Wasserstein distance with respect to $D_{C,c}$.

Table 3 shows that the distance grows, if the calculations are either from another basis set or with another quantum chemistry method. The choice of the latter seems to influence the distance less than the choice in basis set.

In order to investigate, whether the 1-Wasserstein distance is a good predictor of the Δ-QML performance, we calculate the explicit mean absolute error (MAE) for different baselines. We have taken all 8 possible levels, other than $D_{C,c}$, as baseline and calculated the mean absolute error after training SchNet on 1000, 3000 and 5000 molecules.

	1000	3000	5000	1-W
$D_{H,s}$	4.62	2.54	1.26	6.68
$D_{M,s}$	5.2	2.8	1.17	4.90
$D_{C,s}$	4.21	1.88	1.24	4.49
$D_{H,6}$	3.09	1.22	0.92	3.14
$D_{M,6}$	2.03	1.64	0.73	2.55
$D_{C,6}$	1.9	1.08	0.92	2.05
$D_{H,c}$	1.0	0.6	0.47	0.59
$D_{M,c}$	0.8	0.34	0.26	0.36

Table 4: Mean absolute errors of SchNet with different baseline and training size next to the Wasserstein distance.

By using Wasserstein and SchNet, we can make an ordering of the baselines with respect to the expectation of the performance. These are shown below in Table 5 and

Table 6

cc-pVDZ	2	1	-
6-31G	5	4	3
STO-3G	8	7	6
	HF	MP2	CCSD(T)

Table 5: Ordering based on the 1-Wasserstein distance.

cc-pVDZ	2	1	-
6-31G	5	4	3
STO-3G	7	8	6
	HF	MP2	CCSD(T)

Table 6: Ordering based on the mean absolute error.

One may notice that these orderings are almost identical apart from $D_{H,s}$ and $D_{M,s}$. This shows that reducing the mean absolute error by SchNet and the Optimal Transport Problem might indeed be linked to each other. This link between the two problems might help us to determine the intermediate step in multilevel Δ-QML. For instance, we can fix the baseline to the $D_{H,s}$ calculations and find the shortest path to $D_{C,c}$ of length 2. This shortest path indicates, which intermediate step would minimize the total amount of 'work' to shift the histogram of $D_{H,s}$ to D_i, where i is the intermediate level, and then from D_i to $D_{C,c}$.

cc-pVDZ	6.16	6.37	6.68
6-31G	3.74	4.28	4.74
STO-3G	0	1.81	2.34
	HF	MP2	CCSD(T)

Table 7: The 1-Wasserstein distance with respect to $D_{H,s}$.

cc-pVDZ	6.75	6.73	-
6-31G	6.88	6.83	6.80
STO-3G	-	6.71	6.84
	HF	MP2	CCSD(T)

Table 8: The total 1-Wasserstein distance corresponding to its intermediate level.

In Table 8, one can see that $D_{M,s}$ has the lowest Wasserstein distance as intermediate step between $D_{H,s}$ and $D_{C,c}$. Shortly followed by the datapoints that are calculated on the cc-pVDZ basis set. However, the difference between the furthest and shortest path is only $6.88 - 6.71 = 0.17$. Hence, the Wasserstein distances are very close to each other. Note, that all distances in Table 8 are higher than the distance of the direct path between $D_{H,s}$ and $D_{C,c}$: 6.68, as can be seen in Table 7. That the direct path has a lower Wasserstein distance is no surprise, as each distance metric should adhere the triangle inequality, i.e. $W(a,b) \leq W(a,c) + W(c,b)$ for any a, b, c.

One may note, that the use of the Wasserstein distance is only possible if all the datapoints are already precomputed for all combinations of chemistry method and basis set. Which contradicts with the task of minimizing the computational costs of the data generation. To counter this problem of the Wasserstein distance, we will investigate whether sub-samples of the data give representative Wasserstein distances in Subsection 7.3.

To answer research question R2: 'Is it possible to use a QML-free method to rank the available intermediate steps?', it indeed seems to be possible to use a QML-free method to rank the intermediate steps. In the case that the Wasserstein distance is used as such a QML-free method, we basically get the same ranking as the ranking obtained by looking at the MAE corresponding to using different baselines. When we use the Wasserstein distance

as a predictor for which intermediate step to take in the multilevel Δ-QML method, we see that the distances are all quite close to each other. In the next subsection, we investigate if the lowest Wasserstein distance in Table 8 ($D_{M,s}$) indeed give the best combined results between the MAE and computational cost. Besides the Wasserstein distance, there may exist other metrics that can rank the intermediate steps with a more distinct difference between the methods. We leave this as a possibility for further research.

7.2 Multilevel Δ-QML

To get a first impression, whether $D_{M,s}$ is indeed an useful intermediate step, we train with a SchNet model which has the same parameters as described in the tutorial on the QM9 dataset by SchNetPack[Sch]. We use this configuration for the parameters, as this limits the time needed to train the SchNet model, approximately 20 minutes for training on 3000 training samples. In Table 9 some of the values used in the SchNet models are displayed.

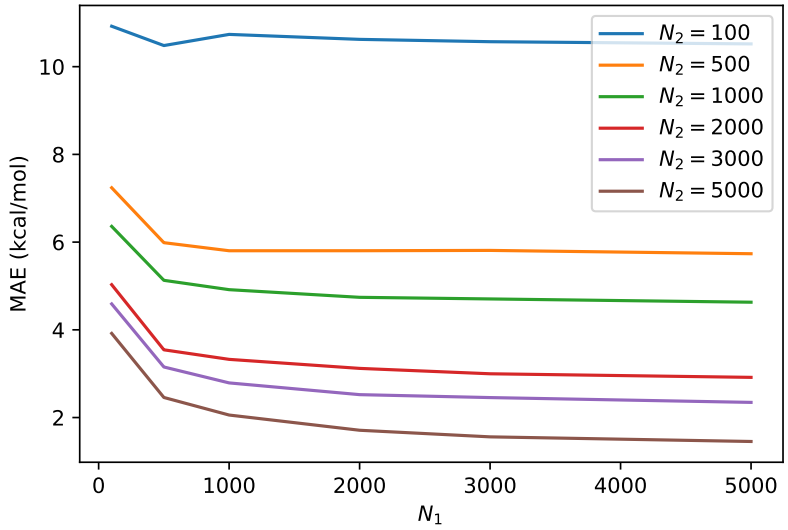
variable name	**value**
n atom basis	30
n filters	30
n gaussians	20
interaction blocks	5
cutoff	4
learning rate	10^{-2}
patience	5
epochs	200
validation size	500

Table 9: Overview of some of the values assigned to variables within SchNet used for tutorial like applications.

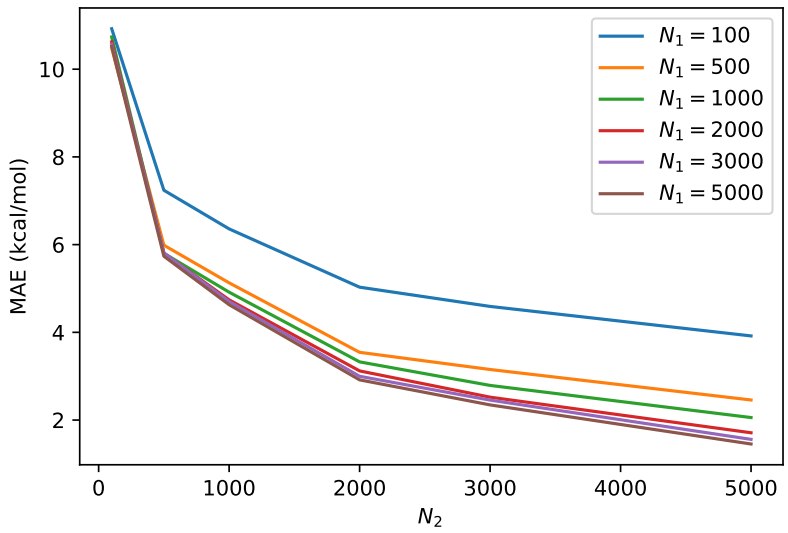
We will start by investigating the ability of SchNet to learn from the data. We investigate this by using training sets with different sizes and calculate the mean absolute error on a fixed test set. These mean absolute errors obtained by different training sets give us the learning curve. For this curve, we expect that the mean absolute error decreases when the number of training samples increases, because then SchNet has more data to learn from and thus should be able to make more accurate predictions. In Figure 7a and Figure 7b, the mean absolute error of the multilevel Δ-QML method is shown, where respectively N_1 and N_2 are on the x-axis and the different colored lines represent different values for N_2 and N_1 respectively.

One may note that the lines in Figure 7b indeed seem like regular learning curves, but the lines in Figure 7a seem to flatten at a 1000 training datapoints. This might be explainable by the Wasserstein distance between $D_{H,s}$ and $D_{M,s}$ as shown in Table 7, which is only 1.81. This relatively small distance indicates that the two distributions are quite similar and therefore we suspect that only a limited number of training samples on $D_{M,s}$ level is needed to get representative predictions.

Furthermore, we can also investigate the cost C of the data generation. We determine



(a)



(b)

Figure 7: $D_{M,s}$ MAE with respect to training sizes N_1 and N_2.

the total cost based on the training sizes N_1, N_2 and the validation size v in the following way:

$$C(N_1, N_2, v) := c_1(N_1 + N_2 + v) + 6509.4(N_2 + v).$$

The value c_1 is determined by Table 2 and depends on the intermediate step we use. The formula for the cost is derived from the amount of datapoints we use for both SchNet models. The first model is trained on $\Delta^i_{H,s}$, where the cost is only determined by the amount of training samples N_1, since we assume that the baseline datapoints $D_{H,s}$ are freely available to us. For the second model, trained on $\Delta^{C,c}_i$, we need N_2 datapoints on $D_{C,c}$ level but also on D_i level, as we need the differences between the values of both levels. Therefore, we need $N_1 + N_2$ datapoints with cost determined by the intermediate step and N_2 datapoints with a cost of 6509.4 seconds. Furthermore, both SchNet models use a validation set of size v, so we need to take them into account as well, leaving us with the equation above with variable c_1 dependent on the intermediate level. Since we have $D_{M,s}$ as intermediate step, we see in Table 2 that $c_1 = 4.1$. In Figure 8 the MAE is depicted against the computational cost.

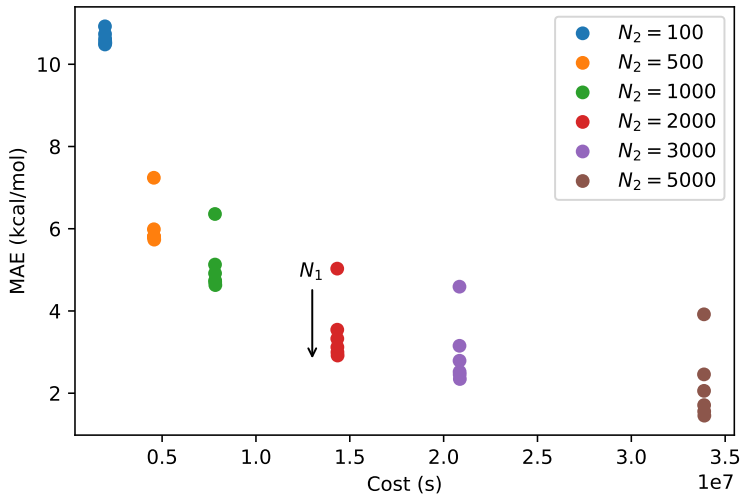
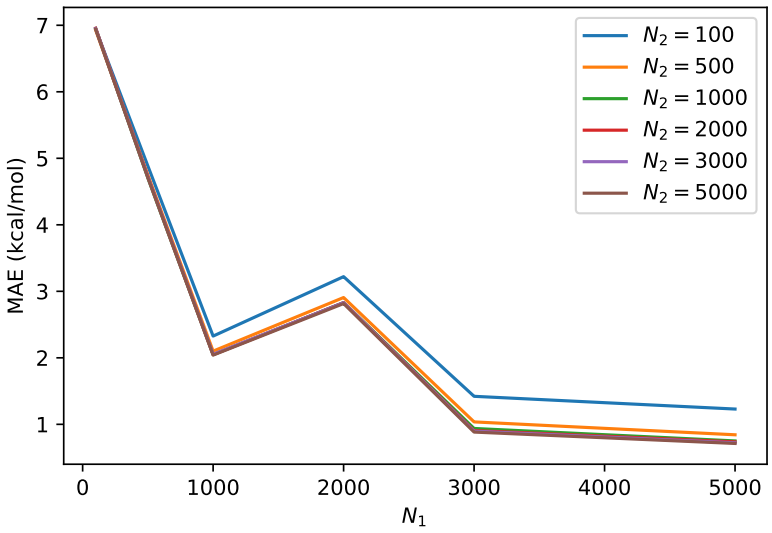


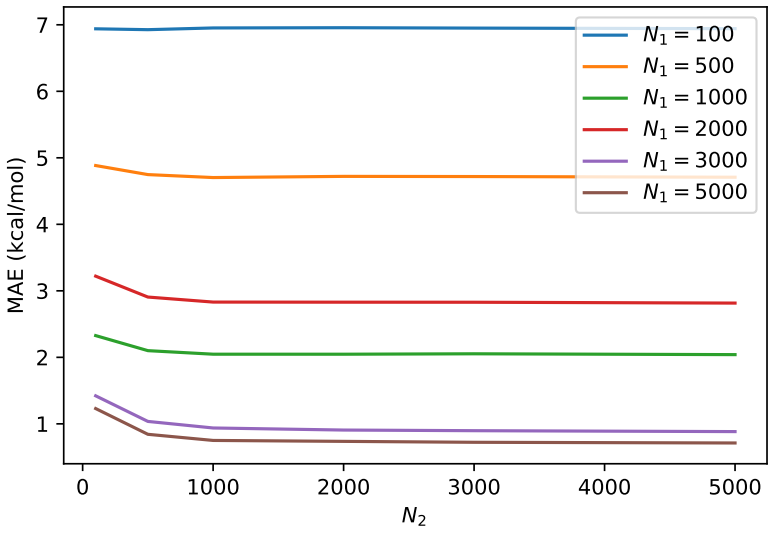
Figure 8: $D_{M,s}$ MAE with respect to the computational cost.

As can be seen in the figure above, the number of training samples N_2 largely determine the total cost for the data generation. As explained before this is due to the large cost associated to the data generation of $D_{C,c}$. To get a reasonable mean absolute error, we need a lot of these datapoints and thus there will be a high computational cost. However, we have seen that if the Wasserstein distance is small, the learning curve flattens out quite quickly. So we expect that we need less N_2 datapoints if we use $D_{M,c}$ as intermediate level, since the Wasserstein distance is only 0.36 between $D_{M,c}$ and $D_{C,c}$ as shown in Table 3. We check this claim by training the SchNet models using the same configuration as before (Table 9) on the distances $\Delta^{M,c}_{H,s}$ and $\Delta^{C,c}_{M,c}$ respectively.

In Figure 9a and Figure 9b, we show the obtained learning curves again. On the left we have the mean absolute error against N_1 and on the right against N_2. As expected the lines in Figure 9b flatten out over the number of training samples.



(a)



(b)

Figure 9: $D_{M,c}$ MAE with respect to training sizes N_1 and N_2.

In Figure 9a, we notice an unexpected bump at $N_1 = 2000$ in the learning curves. This bump is unforeseen as we expect that more training samples result in a lower MAE, since the SchNet model has more data to learn from. One explanation might be that the model over-fitted itself on the 2000 N_1 datapoints, i.e. the model is able to deliver predictions close to the actual atomization energies for the molecules it is trained on, but is unable to generalize the energies to the other molecules in the dataset. Another plausible explanation is that the molecules used for training are not representative for the entire dataset, meaning the model is unable to capture the underlying structure of the data. To investigate if the bump is an incident, we trained an untrained SchNet model for the

$\Delta_{H,s}^{M,c}$ on different training sets of size 2000. The investigation suggests that it is indeed an incidence and not a recurring problem. However, we should keep in mind that those bumps can occur on our SchNet models.

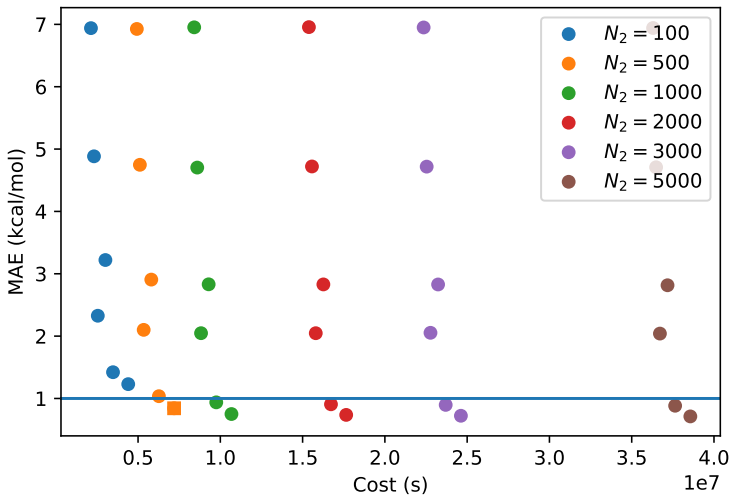


Figure 10: $D_{M,c}$ MAE with respect to the computational cost.

In Figure 10 the mean absolute errors with respect to their cost are shown for $D_{M,c}$ as intermediate step. The vertical difference between points of the same color (same N_2 value) shows that using more N_1 data points decrease the mean absolute error significantly. Furthermore, the addition of more N_2 datapoints makes less of a difference, just as we expected. Now we need only a 1000 N_2 datapoints to get below the threshold value of 1 kcal/mol (see orange square in Figure 10) and using $D_{M,s}$ as intermediate step we needed at least 5000 N_2 datapoints. Therefore, the computational cost decreases from $3.4 \cdot 10^7$ s to only $0.9 \cdot 10^7$ s. This indicates that $D_{M,c}$ is indeed a better intermediate step than $D_{M,s}$, to reduce the computational cost for generating the data.

With the results obtained by the Wasserstein distance before and the MAE corresponding to the different intermediate steps, we can formulate an answer to research question R1: 'Which intermediate step results in the largest reduction of computational cost of the datapoints?'. Based on the MAE obtained by using different intermediate steps for the SchNet models, we determine that $D_{M,c}$ is a better intermediate step than $D_{H,s}$. A first reason for this conclusion is that with $D_{M,c}$ as intermediate step, the SchNet models are able to produce a MAE smaller than 1 kcal/mol, i.e. below the chemical accuracy threshold. Intermediate step $D_{H,s}$ is unable to achieve MAE below the chemical accuracy, even when $N_1 = N_2 = 5000$. Furthermore, the learning curves in Figure 9b are almost flat over N_2, i.e. the MAE are already close to the smallest MAE for low N_2 values, which limit the computational costs for the datapoints. We believe that these flat learning curves are a result of the small Wasserstein distance between $D_{M,c}$ and $D_{C,c}$. So based on the Wasserstein distance we recommend using $D_{M,c}$ as intermediate step over all other possible intermediate steps, as less N_2 datapoints are needed as training data.

Moving forwards with $D_{M,c}$ as our choice for the intermediate step in the multilevel

Δ-QML method, we can investigate whether the amount of data points can be reduced, by adjusting the parameters of SchNet. The parameters are chosen such that they match with the parameters used by the creators of SchNet to achieve the best results on the QM9 dataset. Table 10 shows some of the values for the parameters used in the SchNet models. With these parameters the SchNet models need fewer training samples to obtain a MAE below 1 kcal/mol as shown in Figure 11, however the time spent training the models increases significantly as well. To train a SchNet model on 3000 training samples with these parameters, we need at least 80 minutes.

variable name	**value**
n atom basis	128
n filters	128
n gaussians	50
interaction blocks	6
cutoff	10
learning rate	10^{-4}
patience	25
epochs	100000
validation size	200

Table 10: Overview of some of the values assigned to variables within SchNet used for the benchmark results on the QM9 dataset.

Figure 11 shows the MAE with respect to computational cost when we use more potential from SchNet. In this figure, every color shows the number of N_2 values used while training as shown in the legend, i.e. $N_2 = 40 \cdot i$ for $i \in \{1, \ldots, 9\}$. For each color, the dot on the left represents $N_1 = 3400$ and each subsequent dot is an increase of 200, resulting that the most right dot corresponds to $N_1 = 5600$.

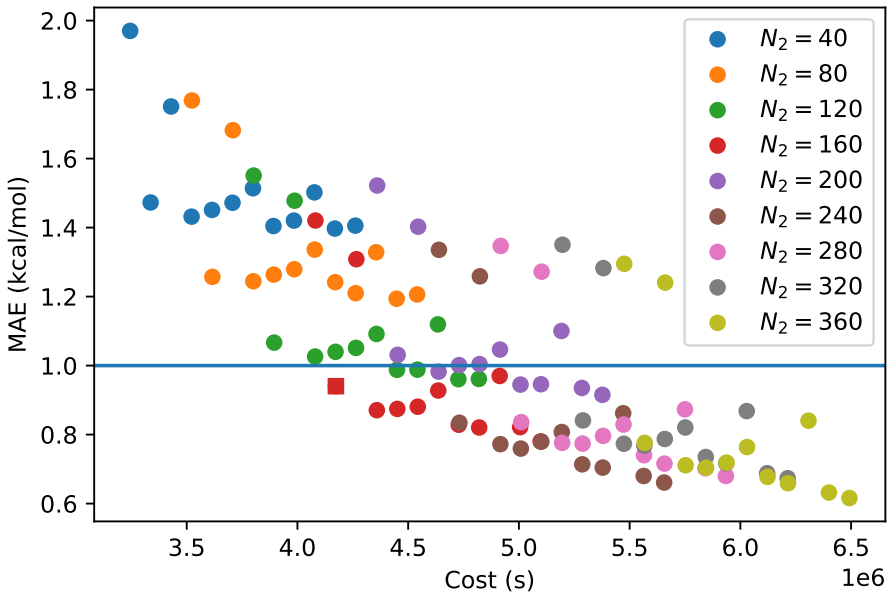


Figure 11: MAE with respect to computational cost

As can be seen in the figure above, the least amount of computational cost, while maintaining an MAE below the threshold of 1 kcal/mol (represented with the red square in Figure 11), is achieved when using (only) 160 data points to train the second SchNet model, i.e. the model trained on $\Delta^{C,c}_{M,c}$. Additionally, we need 3600 datapoints to train the other SchNet model. Note that on top of the training sizes N_1 and N_2 both models use 200 validation datapoints, so we need to include them as well when we calculate the computational cost. In total the computational costs will be $4.2 \cdot 10^6$ s, which is approximately half of the $0.9 \cdot 10^7$ s needed in the previous case.

Let us answer research question R3:'How much can we reduce the computational costs by predicting the effective averaged atomization energies instead of calculating all of them?'. By using the multilevel Δ-QML method and using SchNet as neural network to predict the differences $\Delta^{M,c}_{H,s}$ and $\Delta^{C,c}_{M,c}$, we managed to reduce 543 days of generating atomization energies to only 48 days. There might be even more cost reduction possible, for instance by playing around with the parameters of the used SchNet models. By playing around with the parameters, we might be able to get better SchNet models, in the sense that they achieve the same MAE with less training samples. However, checking all combinations for the parameters is a time consuming task, so we will leave that as a possibility for future research. In particular, there can be looked further into the role of the validation set in SchNet models. Especially for the model trained on the distances $D^{C,c}_{M,c}$, since the validation set is larger than the training size used to obtain the lowest computational cost. Therefore, even a small reduction of the validation size may have an impactful result on the total computational cost. One can also research whether multiple intermediate steps in the multilevel Δ-QML method yield more cost reduction. In this report we focused on only a single intermediate step ($D_{M,c}$), however it is also possible to use an intermediate step between $D_{H,s}$ and $D_{M,c}$, i.e. using two intermediate steps. How-

ever, three SchNet models are needed in that case as we have three differences. Therefore, the runtime of the SchNet models on the computer will increase as more models have to be trained. Despite the increase in runtime, it might still be beneficial to consider two or even more intermediate steps. Lastly, the use of a minimization algorithm, for instance a genetic algorithm, might be interesting to investigate. The algorithm should navigate over the values N_1 and N_2 both in the range $\{1, \ldots, 7211\}$, with objective minimizing the computational cost under the constraint that the MAE is smaller than chemical accuracy. One advantage of using an algorithm for the N_1 and N_2 values, is that these values should converge to the best possible combination, instead of us just trying certain N_1 and N_2 values. On top of that, the algorithm might be able to reduce the number of SchNet models that are trained, because it does not have to use all possible values for the training sizes. Therefore, not only the computational cost will be minimized, but also the time spent training SchNet models. However, if the algorithm is free to take any value for the training size, for example N_2 as 7000, then the purpose of our research is negated, as we still need to compute 7000 datapoints. A simple solution is to bound both N_1 and N_2 from above, but then we remain with the question: what should the upper-bound be. Still it might be interesting to further investigate the possibility of using some sort of optimization algorithm.

7.3 Wasserstein distance with sub-sampling

The choice for $D_{M,c}$ as intermediate step in the multilevel Δ-QML method is based on the Wasserstein distance and supported by the experimental results. However, for the Wasserstein distance we used all available data, so basically we are still using 7211 datapoints on $D_{C,c}$ level. Since we used the Wasserstein distance as indication for the intermediate step, it will be interesting to investigate whether we can limit the datapoints used in the Wasserstein distances. As a test we take differently sized sub-samples of the available data and calculate the distances on these samples, note that for this comparison it is important to use data only from the same molecules.

As the values of the Wasserstein distance become more unreliable, when we use less datapoints, it might be more beneficial to look only at the order of D_i based on the Wasserstein distances, just like Table 5. We suspect that the distances rely on the specific subset of molecules that is being taken. Therefore, we have to repeat the process of sub-sampling multiple times, to get meaningful results. We take 10000 times a different sub-sample of size n_s, where $n_s \in \{100 \cdot k | k = 1, \ldots, 50\}$. For each of these individual sub-samples, we check how often the ordering remains the same as in Table 5. In Figure 12 the percentage of times the ordering stays the same is shown for each n_s value.

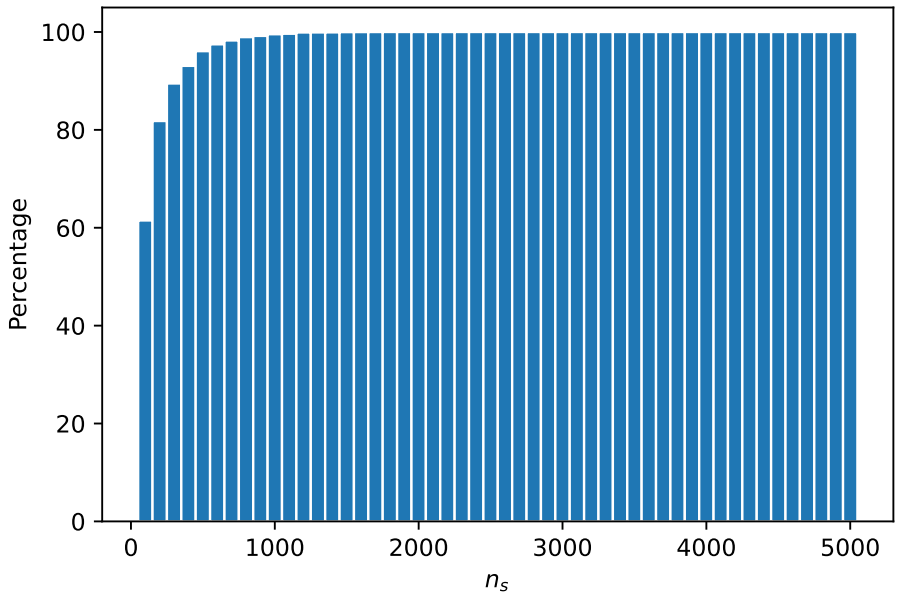


Figure 12: The percentage of times that the sub-sample size n_s gave the same ordering as in Table 5.

One can see in the figure above, that the percentage becomes close to 100% quite quickly. It seems that using at least 1100 datapoints for the Wasserstein distances gives the same ordering every time. This is quite an increase when compared to the 360 datapoints we needed to train and validate the second SchNet model. However, at 500 datapoints we get a 96% success rate, so there is an option to take only 500 datapoints for the Wasserstein distances. As an advantage we reduce the amount of datapoints by more than half, however we only have a 96% possibility that the ordering is the right one.

8 Conclusion

In this report we investigated the possibility to downscale the computational time needed to calculate the atomization energy with the CCSD(T) method under the cc-pVDZ basis set. For the entire QM7 dataset these calculations take 543 days in total. In particular, we investigated the use of Δ-QML in combination with the neural network SchNet to reduce the amount of necessary datapoints and still obtain valuable datapoints. Since Δ-QML still relies mostly on the amount of high accuracy datapoints, we used a multilevel variant, which takes at least one intermediate step. This variant comes at the cost of having to use two individual SchNet models, since we have to train them both on $\Delta_{H,s}^{i}$ and $\Delta_{i}^{C,c}$.

First we examined which intermediate step to take, both by experimental results as by the Wasserstein distance. These methods indicated that $D_{M,c}$ as intermediate step yield the best results, since the number of datapoints N_2 for the second SchNet model are minimized. In the case of the available dataset and goal of the project, the minimization over N_2 is more important than for N_1 as the data generation costs significantly more for obtaining the $\Delta_{i}^{C,c}$ values. After we fixed the intermediate level to $D_{M,c}$, we analyzed the mean absolute errors when using a more sophisticated SchNet model. Where this variant of the SchNet model differs mostly in the size of the feature vector $\mathbf{x}$ and the amount of epochs.

To conclude, the combination of using SchNet in the Δ-QML method can indeed reduce the computational cost of obtaining all atomization energies. The effectiveness of this method is determined by the used parameters within SchNet and which intermediate step we use. We already managed to reduce the computational time from 543 days to only 48 days.

However, it might be possible to reduce these costs even further. Firstly, the parameters within the SchNet models are chosen to be the same as when the model would be used for the QM9 dataset. Since we use a different dataset and we have a different objective, it is possible that changing some of the parameters yield better results. We chose not to investigate the effect in this report as it is a time-consuming task. Secondly, the Δ-QML could be exploited further with respect to the number of intermediate steps that are taken. As shown in Section 7, the amount of datapoints needed in the SchNet models decreases, when the Wasserstein distance becomes smaller. We have quite a large distance between $D_{H,s}$ and $D_{M,c}$, so using another intermediate step might reduce the total cost again. However, we suspect this reduction to be less impactful than the already used intermediate step, since the order of the data generation cost is significantly smaller than the cost for $D_{C,c}$.

References

[Har28] D. R. Hartree. "The Wave Mechanics of an Atom with a Non-Coulomb Central Field. Part II. Some Results and Discussion". en. In: *Mathematical Proceedings of the Cambridge Philosophical Society* 24.1 (Jan. 1928), pp. 111–132. ISSN: 0305-0041, 1469-8064. DOI: 10.1017/S0305004100011920. URL: https://www.cambridge.org/core/product/identifier/S0305004100011920/type/journal_article (visited on 05/24/2022).

[Lu+19] Chengqiang Lu et al. *Molecular property prediction: A Multilevel Quantum Interactions Modeling Perspective*. June 2019. URL: https://arxiv.org/abs/1906.11081.

[PC20] Gabriel Peyré and Marco Cuturi. *Computational Optimal Transport*. Mar. 2020. URL: https://arxiv.org/abs/1803.00567v4.

[Rup+12] Matthias Rupp et al. *Fast and accurate modeling of molecular atomization energies with machine learning*. Jan. 2012. URL: https://journals.aps.org/prl/abstract/10.1103/PhysRevLett.108.058301.

[Sch] K. T. Schütt et al. *Atomistic-machine-learning/Schnetpack: Schnetpack - deep neural networks for atomistic systems*. URL: https://github.com/atomistic-machine-learning/schnetpack.

[Sch+17] Kristof T. Schütt et al. *Quantum-chemical insights from deep tensor neural networks*. Jan. 2017. URL: https://www.nature.com/articles/ncomms13890.

[Sch+18] K. T. Schütt et al. "SchNet – a deep learning architecture for molecules and materials". In: *The Journal of Chemical Physics* 148.24 (Mar. 2018), p. 241722. DOI: 10.1063/1.5019779.

[Sch+19] K. T. Schütt et al. "Schnetpack: A deep learning toolbox for atomistic systems". In: *Journal of Chemical Theory and Computation* 15.1 (2019), pp. 448–455. DOI: 10.1021/acs.jctc.8b00908.

[Sch26] E. Schrödinger. "An Undulatory Theory of the Mechanics of Atoms and Molecules". en. In: *Physical Review* 28.6 (Dec. 1926), pp. 1049–1070. ISSN: 0031-899X. DOI: 10.1103/PhysRev.28.1049. URL: https://link.aps.org/doi/10.1103/PhysRev.28.1049 (visited on 05/24/2022).

[Sla28] J. C. Slater. "The Self Consistent Field and the Structure of Atoms". en. In: *Physical Review* 32.3 (Sept. 1928), pp. 339–348. ISSN: 0031-899X. DOI: 10.1103/PhysRev.32.339. URL: https://link.aps.org/doi/10.1103/PhysRev.32.339 (visited on 05/24/2022).

[Zas+19] Peter Zaspel et al. "Boosting Quantum machine learning models with a multilevel combination technique: Pople diagrams revisited". In: *Journal of Chemical Theory and Computation* 15.3 (2019), pp. 1546–1559. DOI: 10.1021/acs.jctc.8b00832.