MACHINE LEARNING
Science and Technology

PURPOSE-LED PUBLISHING™

Part of the MACHINE LEARNING Series

PAPER • OPEN ACCESS

Graph2Mat: universal graph to matrix conversion for electron density prediction

To cite this article: Pol Febrer *et al* 2025 *Mach. Learn.: Sci. Technol.* **6** 025013

MACHINE
LEARNING
Science and Technology

OPEN ACCESS

RECEIVED
8 August 2024

REVISED
15 March 2025

ACCEPTED FOR PUBLICATION
1 April 2025

PUBLISHED
14 April 2025

PAPER

Graph2Mat: universal graph to matrix conversion for electron density prediction

Pol Febrer[1], Peter Bjørn Jørgensen[2], Miguel Pruneda[1,3], Alberto García[4], Pablo Ordejón[1] and Arghya Bhowmik[2,*]

[1] Institut Català de Nanociència i Nanotecnologia (ICN2), UAB Campus, Bellaterra, Spain
[2] Department of Energy and Storage, Technical University of Denmark, 2800 Kongens Lyngby, Denmark
[3] Nanomaterials and Nanotechnology Research Center (CINN-CSIC), Universidad de Oviedo (UO), Principado de Asturias, 33940 El Entrego, Spain
[4] Institut de Ciència de Materials de Barcelona (ICMAB-CSIC), Bellaterra E-08193, Spain
[*] Author to whom any correspondence should be addressed.

E-mail: arbh@dtu.dk

Keywords: electronic structure, density matrix, euclidean neural network, graph neural network, electron density prediction, ml accelerated simulation

Abstract

The electron density is a fundamental observable of an atomic system from which all ground-state properties can be computed. As a prediction target for machine learning (ML) models, electron density is often represented on a dense real space grid, which is data heavy, or through density fitting approximations. In this work, we show the power of targeting the density matrix (DM), a linear-scaling sparse SE(3) equivariant matrix that encodes the exact density. We introduce Graph2Mat, a universal function for converting molecular graphs into equivariant matrices. We demonstrate how a ML model that combines this Graph2Mat approach with state-of-the-art molecular graph representations can accurately predict the DM of molecular systems. The models achieve state-of-the-art performance on electron density prediction by matching the accuracy of grid-based methods, while using datasets that are at least one order of magnitude smaller. Accurately predicted electron densities can also accelerate density functional theory (DFT) calculations by reducing the number of self-consistent field (SCF) iterations needed to converge. In this work, we get an average 40% reduction on the number of SCF steps in DFT calculations of QM9 molecules with SIESTA. The novel prediction model also allows for two new and promising measures of uncertainty (total charge error and self-consistency error) that will facilitate its practical usage, e.g. within active learning workflows. These results open the door for many applications using hybrid ML-accelerated DFT methodologies, and uncertainty aware single iteration *ab initio* molecular dynamics.

1. Introduction

Density functional theory (DFT) is the method of choice for efficient and often accurate calculation of the ground-state properties of molecules and materials. DFT simulations have enabled the rapid screening of new molecules [1] and materials [2] for all varieties of applications. To achieve accurate but computationally less expensive simulations, in recent years machine learning (ML) based surrogate methods have been developed [3, 4]. Those are frequently trained on DFT data to predict the total energy and atomic forces, enabling much larger time- and length-scale simulations. These methods show high accuracy on benchmark datasets, but often fail to generalize to new configurations [5] and require new DFT data collection [6] and training for exploration of newer geometries often based on uncertainty metrics [7].

In DFT, the electron density is a fundamental quantity from which all other ground-state properties of the system can be derived, including the energy and forces. We expect that predicting the electron density and then using physics laws to compute other target properties [8] will lead to models that generalize better to new configurations. Previous works also show the possibility of achieving data efficient learning of higher level theory energies [9], system size adaptability [10], and even usability with an unseen element containing systems [11]. This increase in robustness by incorporating physics into ML applications is not particular to electronic structure and has been observed across many fields [12]. The incorporation of physics also allows the system to learn a more comprehensive picture of the system's quantum mechanical state, moving away from the popular black-box predictive ML surrogate approach. Another interesting application for predicted electron densities is the acceleration of DFT. DFT starts from an initial guess of the electron density and then iterates through the self-consistent field (SCF) cycle modifying the density until it is consistent with the Kohn–Sham equations. The most typical initial guess is the density of the isolated atoms (atomic densities), which is a very rough estimate but usually leads to a stable convergence path. Using predicted electron densities as the initial guess offers the possibility to significantly reduce the number of SCF iterations, since the initial density will be much closer to the converged density. When using the predictions in this manner, there is no compromise in reliability for the simulation of new systems (unlike pure ML surrogate models) as DFT is still being used to converge the solution. The resulting acceleration can make it feasible to explore wider and newer phase spaces with certainty.

In this work, we present a novel deep learning method for predicting the electron density. Our method builds upon recent advances in equivariant graph neural networks (GNN) [13]. Previous approaches often have used a grid representation of the density to learn it with structure fingerprint [14], probe node representation on polarizable atom interaction GNN [8] or E(3)-equivariant GNN [15], image super-resolution [11]. Alternatively, models have been constructed for learning coefficients coming from density fitting from atomic fingerprints [16] or with GNN [17]. Taking a completely new direction, we predict the *density matrix* ($\rho_{\mu\nu}$ or DM) of the system.

The DM, formally an operator representing the projection on the occupied subspace, can be seen in practice as a decomposition of the electron density into contributions from pairs of basis functions (ϕ_μ, ϕ_ν):

$$\rho\left(\vec{x}\right) = \sum_{\mu}\sum_{\nu}\rho_{\mu\nu}\phi_{\mu}\left(\vec{x}\right)\phi_{\nu}\left(\vec{x}\right). \tag{1}$$

For atom-centered basis sets, ϕ_μ and ϕ_ν are centered on atoms, but not necessarily both on the same atom. The DM can be divided into atom-pair blocks ρ_{ij} containing the coefficients for all products $\phi_\mu\phi_\nu$ where $\mu \in i, \nu \in j$. This structure naturally lends itself to be modeled as a GNN where the task is to predict nodes (ρ_{ii}) and edges ($\rho_{ij}, i \neq j$). Furthermore, if the basis functions have a finite range, the only relevant elements for the electron density are those for which ϕ_μ and ϕ_ν overlap. This sparsity pattern is naturally captured by the graph structure and makes the task scale linearly with the number of atoms, which makes it tractable for large systems. This relation between the matrix and an atomic graph can only be made if the basis functions are atom centered (e.g. in DFT codes like SIESTA [18] or FHI-aims [19]), but not for other cases (e.g. for planewave based DFT codes).

From the point of view of equation (1), $\rho_{\mu\nu}$ can be seen as the coefficients of a density fitting expansion using $\phi_\mu\phi_\nu$ (two-center products of basis functions) as the basis functions. In practical terms, we can treat our prediction problem like the prediction of such a density expansion. However, the DM represents the exact density of a DFT simulation with an atom centered basis set. If the training data comes from such a simulation, the model is being trained on the exact DFT density, not a fitting. Another motivation to learn the DM instead of a simple density fitting (with single atom-centered basis functions) is that the DM contains higher symmetry information. We expect that less data is required for training and the models should be able to extrapolate better by considering the full set of symmetries. Compared to grid-based approaches, the compact encoding of the DM as well as the symmetry information in it makes its learning much more data-efficient. The disadvantage of any basis-based approach is that the coefficients depend on the basis, and therefore a change of basis would require retraining the model.

The case where the basis functions' angular dependence is dictated by spherical harmonics, e.g. $\phi_(\vec{r}) = R(r)Y_l^m(\theta, \varphi)$, is particularly interesting. The DM then is SE(3) equivariant. That is, it transforms predictably under translations and rotations of the system. In particular it transforms as the product of spherical harmonics. Previous works using polarizable atom interaction GNN [8], E(3)-equivariant GNNs for learning probe node density [15] or density fitting coefficients [10] have shown that taking into account the equivariance of the data has many benefits for the learning process, including the need for less training data. The equivariance of the matrix is significantly more complex than the equivariance of energies (they are invariant) or vector properties like forces (they rotate along with the system coordinates), which have been successfully exploited in interatomic potentials based on graph edge based GNNs [20] and many body message based GNNs [13]. Previous works have exploited matrix equivariance to successfully fit Hamiltonian matrices with atom centered basis sets based on SE(3) GNN [21], atomic cluster expansion [22] full E(3)

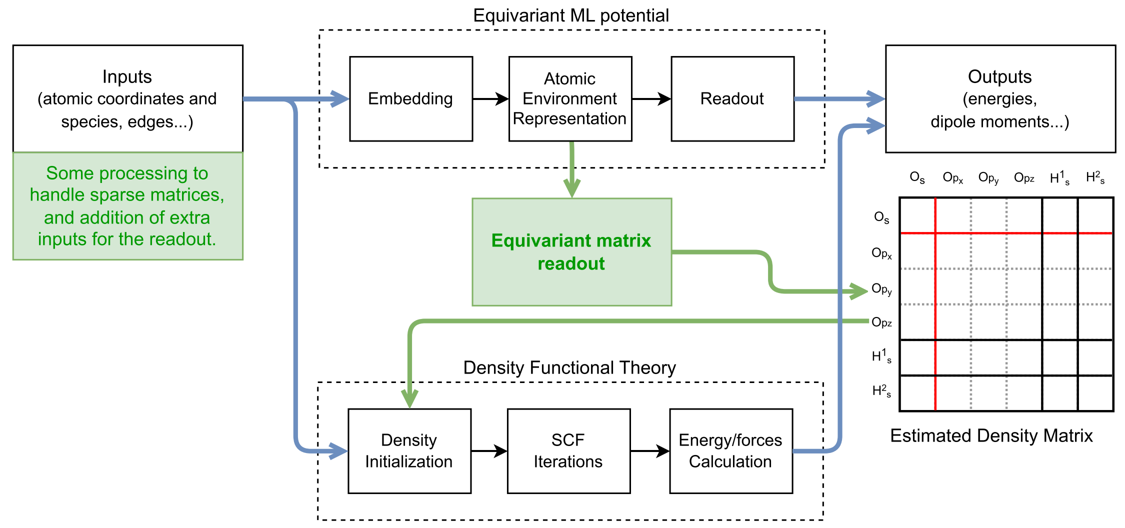

Equivariant ML potential
Inputs
(atomic coordinates and species, edges...)
Some processing to handle sparse matrices, and addition of extra inputs for the readout.
Embedding
Atomic Environment Representation
Readout
Outputs
(energies, dipole moments...)
Equivariant matrix readout
Density Functional Theory
Density Initialization
SCF Iterations
Energy/forces Calculation
O_s
Op_x
Op_y
Op_z
H^1_s
H^2_s
Estimated Density Matrix

Figure 1. Interplay between our method (displayed in green), molecular graph representation learning and density functional theory (DFT) based simulations. The atomic environment representation is converted to a density matrix. Then, the predicted DM is fed into a physics-based engine (in this case DFT) that can compute many system properties. DFT can take the prediction as the truth (hybrid data-driven/physics-based method) or refine the density until its converged (accelerated physics-based method). The displayed density matrix illustrates the anatomy of a matrix for a water molecule with a single zeta (SZ) basis. Black lines delimit blocks containing all interactions between an atom pair (ρ_{ij}). Red lines mark the boundaries for interactions between a pair of sets of orbitals.

GNN [23] and descriptor based models [24]. Very recently two works have also introduced the one electron reduced DM as a learning target using the potential matrix [25] and the atomic coordinates [26] as input for the models. The models presented in those works are not inherently equivariant, equivariance is treated as a data processing step. In practical terms this hinders the usability for large systems and new chemical spaces. Both works show only application to small molecules and use a new model for each molecule.

In our approach, we demonstrate that it is possible to use the knowledge of the matrix symmetries to build an equivariant model that predicts the density and thus accelerates DFT simulations over large chemical spaces and for big systems. To the best of our knowledge, this is the first demonstration of DFT acceleration through DM prediction over such a wide range of systems. Our models rely on established frameworks for atomic environment representation, and then convert those representations to a matrix. The interplay of our models with ML interatomic potentials and DFT is displayed in figure 1.

In summary, the main contributions of our work are:

- We define a method, which we call Graph2Mat, for converting an atom centered equivariant representation (based on e3nn [27]) to a sparse matrix of spherical harmonics interactions. The DM is a particular example of such matrices, but the framework is generic and can be applied to learn other matrices such as the Hamiltonian (H), the Overlap (S) or the energy DM (EDM).
- We implement the methods needed to efficiently deal with the sparsity of these matrices during training and inference.
- For the first time, we propose a linear scaling DM prediction method that will allow very large scale DFT simulations accelerated with ML
- All the functionality is packaged in an open source python package (graph2mat, which can be found at https://github.com/BIG-MAP/graph2mat) for ease of access and further development.

With this method and related implementations, we enable:

- Faster DFT calculations with fewer SCF iterations.
- Non self-consistent DFT with similar accuracy as self-consistent DFT.
- Data efficient training with electron densities.

2. Methods

Graph2Mat is a learnable SE(3) equivariant function that converts a graph to an equivariant matrix, as depicted in figure 2. Coupled with some atomic environment representation, we use it to approximate the computation of the electronic DM for a given subset of the chemical space. The decision to split the problem into two tasks (first describing the atomic environment and then creating the matrix out of it) is motivated by the possibility of utilizing the newest models from the molecular graph representation learning community, which is developing fast. In making Graph2Mat a standalone portion of code, we allow seamless switching to improved atomic environment representations developed in the future, as long as it is based on *e3nn* [27] or can be converted to it. It also allows for picking the representation that fits best for a particular problem. In this work, we use MACE [13] as the atomic environment representation, and therefore we refer to the model architecture as MACE+Graph2Mat. End-to-end learning is performed from atomic numbers and coordinates to the equivariant matrices through the whole MACE+Graph2Mat model. As the MACE representation is also graph-based, we use the same graph for MACE and Graph2Mat for practical reasons, but this is not a requirement.

2.1. Graph2Mat: generating an equivariant matrix

Before discussing the process of generating an equivariant matrix $M_{\mu\nu}$, it is important to understand its internal structure. In figure 1 the example output matrix is an equivariant matrix for the simple case of a water molecule with a single zeta (SZ) basis [28]. In a SZ basis, hydrogen atoms have only one s $(\ell = 0)$ orbital each, while oxygen has one s orbital and a set of p $(\ell = 1)$ orbitals. Delimited by black lines we find all the elements that belong to an atom–atom interaction. We will refer to them as M_{ij}, being i and j the indices of the atoms that interact. These blocks will be the target unit of our predictions. As discussed in section 1, this makes GNNs a natural strategy where node n_i should be trained to produce M_{ii} and the output of an edge $n_i \rightarrow n_j$ (e_{ij}) should be trained to be M_{ij}. Therefore, the goal is to find two functions f_n and f_e that generate the blocks from nodes and edges respectively. That is, $f_n(n_i) = M_{ii}$ and $f_e(n_i, n_j) = M_{ij}$. These functions must be equivariant. In the following paragraph we discuss the conditions that they must satisfy.

Each element of an equivariant matrix $M_{\mu\nu}$ corresponds to the interaction between a basis orbital μ and another basis orbital ν. However, by predicting each element in the matrix separately we would be ignoring the correlations that exist between them. Considering their collective behavior is a physically justified approach, and that makes the model more robust. In particular, a set of orbitals with the same ℓ number and radial function can be clearly identified as a group. Each group has $2l + 1$ elements. Data associated to these groups will be arrays of length 1 $(\ell = 0)$, 3 $(\ell = 1)$, 5 $(\ell = 2)$... In fact, $\ell = 0$ data is a scalar invariant quantity and $\ell = 1$ arrays are simply equivariant 3D vectors, and it is trivial to understand how they should behave under rotations of the system. Even for higher ℓ data, whose distribution in space is harder to grasp, the transformations are perfectly defined. The next natural step is then to ask what happens when two of these groups interact in a product. Take for example the product between two sets of p $(\ell = 1)$ orbitals (p_x, p_y, p_z). The outer product will create a 3×3 matrix containing all possible products. If we want to predict this matrix, we could treat each of the elements of the matrix individually, loosing all notion of what the equivariance of the data was. But within the e3nn framework [27], the matrix can be decomposed into a set of arrays with known equivariance. This is known as the irreducible representation (from now on, *irrep*). In particular, the irrep of the $p \times p$ matrix consists of three features: an $\ell = 0$ feature (the trace, which is the scalar product), an $\ell = 1$ feature (related to the symmetry of the matrix, the cross product) and an $\ell = 2$ feature (rest of the matrix). A given atom might contain more than one set of orbitals, but the irrep is simply the sum of the irreps of all possible products between them. For example, given two atoms with s and p orbitals:

$$\mathrm{irreps}\left(M_{ij}\right) = \mathrm{irreps}\left((s+p) \times (s+p)\right) = \mathrm{irreps}\left(s \times s\right) + \mathrm{irreps}\left(s \times p\right) + \mathrm{irreps}\left(p \times s\right) + \mathrm{irreps}\left(p \times p\right). \quad (2)$$

Therefore, each atom-pair block has a different irreducible representation. The trivial approach to this problem is to have a separate function for each block type t. We have taken this approach, as depicted in figure 2. An M^t_{ii} is a node block for atom of type t, while $M^t_{ij}, i \neq j$ is an edge block where t is the edge type, determined by the two atom types that it connects. We therefore need as many $f^t{}_n$ functions as atom types in the dataset, and as many $f^t{}_e$ functions as edge types in the dataset. Each of those functions must be an equivariant function that produces an output with the shape of irreps(M^t_{ij}). The irreducible representation is then expanded into the matrix block. Finally, all blocks are combined into the final full matrix of the system (in practice this is a flat array, not a 2D matrix, because the matrix is sparse). The f functions can be arbitrarily complex, but they can also be as simple as a tensor square for f_n and a tensor product for f_e.

The overall architecture of Graph2Mat is depicted in figure 2. Graph2Mat has two inputs. The first input is the structural information, represented as a graph that contains the atomic types as well as the connections between atoms. As commented in section 1, we are only interested in matrix elements of overlapping pairs of orbitals. Therefore, in this graph two atoms should be connected only if any of their orbitals overlap. The second input is the atomic environment representations (in our work, MACE node embeddings), which need to be equivariant with respect to the atomic coordinates. Before going into the block generating

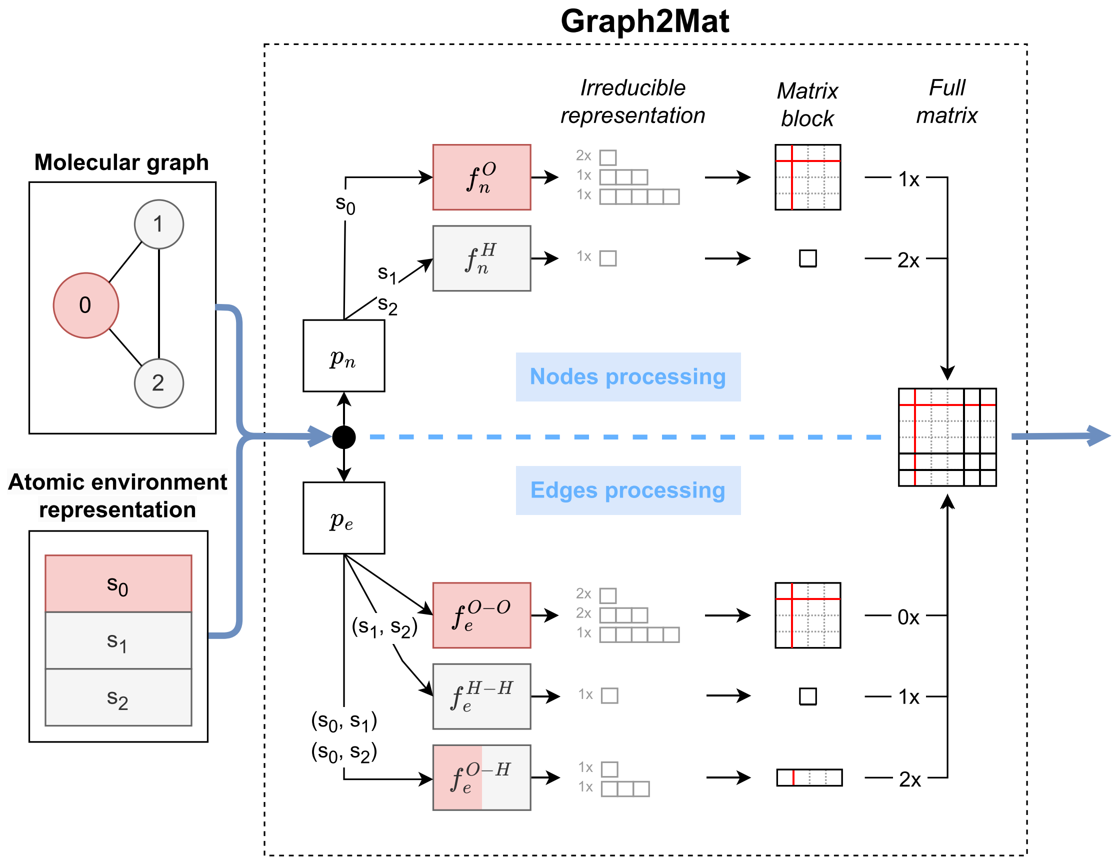


Figure 2. Architecture of the Graph2Mat function for a dataset consisting of structures with O and H atoms using a SZ basis. The figure also illustrates the passing of a water molecule through it. The two inputs are the molecular graph and the atomic environment representation for each node in the graph. Then, the processing splits in two: the computation of node blocks (M_{ii}) and the computation of edge blocks ($M_{ij}, i \neq j$). Preprocessing functions p compute the final states that will be passed to the block generating functions f. Each function f produces an output with the shape of an irreducible representation, and that representation is then converted to a matrix block M_{ij}. Finally, all blocks are merged to generate the full matrix.

functions f we allow for a pre-processing function p that modifies the graph. In general, there can be two separate pre-processing functions, one for the nodes (p_n) and one for the edges (p_e). The function p can be for example a message passing layer.

The Graph2Mat function is not specific to the DM and therefore can be used to generate any other matrix that is composed of tensor products in a basis of spherical harmonics. Other orbital equivariant matrices which originate as the representation of an operator in a basis set that involves spherical-harmonics are for example the Hamiltonian (H), the Overlap (S) or the EDM. However, Graph2Mat is not specific for atomic systems and therefore can potentially be used to compute any equivariant matrix for a 3D graph with arbitrary point types.

2.2. MACE + Graph2Mat: hyperparameters

2.2.1. MACE hyperparameters

In section 3 we study the influence of MACE hyperparameters on the DM predictions. These hyperparameters have been thoroughly explored in the MACE paper [13] for energy and force targets. Some of them are: hidden representation size (which in this case is the size of the representation inputted into Graph2Mat), correlation number (controls the body order of the interactions, e.g. 2-body, 3-body, etc), number of message passing interactions, and max ℓ (maximum order of spherical harmonics used during the interactions).

2.2.2. Graph2Mat hyperparameters

For the results presented in this work, we use a MACE-like message passing layer for the pre-processing functions p. For the f functions, we use simple learnable operations provided directly by the e3nn python package [27]. We use a tensor square operation for f_n and a tensor product operation for f_e, filtering the output to get only the required irreps. The size and order of MACE's representation is maintained by the pre-processing functions p.

3. Results

In this section we present the results obtained with the MACE+Graph2Mat model. We focus on (a) effect of hyperparameters in accurately predicting the DM with the corresponding density error (b) the learning curves based on dataset size (c) utility in integration of the model within DFT and (d) a unique and novel way of estimating uncertainty for such models.

3.1. Datasets

The datasets used for this work are computed with SIESTA method [18] using latest version [29]. We use pseudodojo's [30] psml pseudopotentials and all of SIESTA's default settings, including the default split-valence DZP basis set, unless stated otherwise. In the dataset, along with the coordinates and atomic numbers of each structure, we store the target matrix (or matrices). In particular, the datasets we use contain the DM, overlap, Hamiltonian, and EDM. However, we only present the results on predicting the DM, which is the focus of this paper.

3.1.1. MD17 aspirin

MD17 [31] is a molecular dynamics dataset with trajectories of benzene, uracil, naphthalene, aspirin, salicylic acid, malonaldehyde, ethanol, and toluene. The dataset is often used as a small benchmark for ML force fields. We only use the aspirin trajectory.

3.1.2. QM9

QM9 [32] is a dataset of 134k small organic molecules with up to nine heavy atoms (CNOF) and is widely used for benchmarking ML models for molecular property predictions.

3.1.3. Ethylene carbonate

The ethylene carbonate (EC) dataset is a molecular dynamics simulation of a liquid ethylene carbonate electrolyte consisting of 8 ethylene carbonate molecules in a unit cell. The dataset [33] was developed for benchmarking the equivariant DeepDFT model [8]. Although EC is not a crystal, it is a periodic system and it therefore tests whether the models treat periodicity correctly.

The three datasets are available at [34].

3.2. Training

We train end-to-end from coordinates and atomic numbers to the DM. To avoid the overhead of computing the electronic density in real space or even constructing the matrix with the appropriate shape, we compute the loss function on flat arrays containing the matrix values as returned by Graph2Mat. Element-wise MAE is used as the loss function, ignoring errors that are lower than the SCF convergence threshold in SIESTA. Our tests have shown that using MSE instead of MAE leads to decreased performance. An Adam optimizer with a learning rate of 0.005 is used to optimize the parameters of the model. The training presented here has been performed on a single NVIDIA GeForce RTX 3090 card. However, our code is ready for multi-GPU distributed training, and we have tested for it.

3.3. Sensitivity to hyperparameters

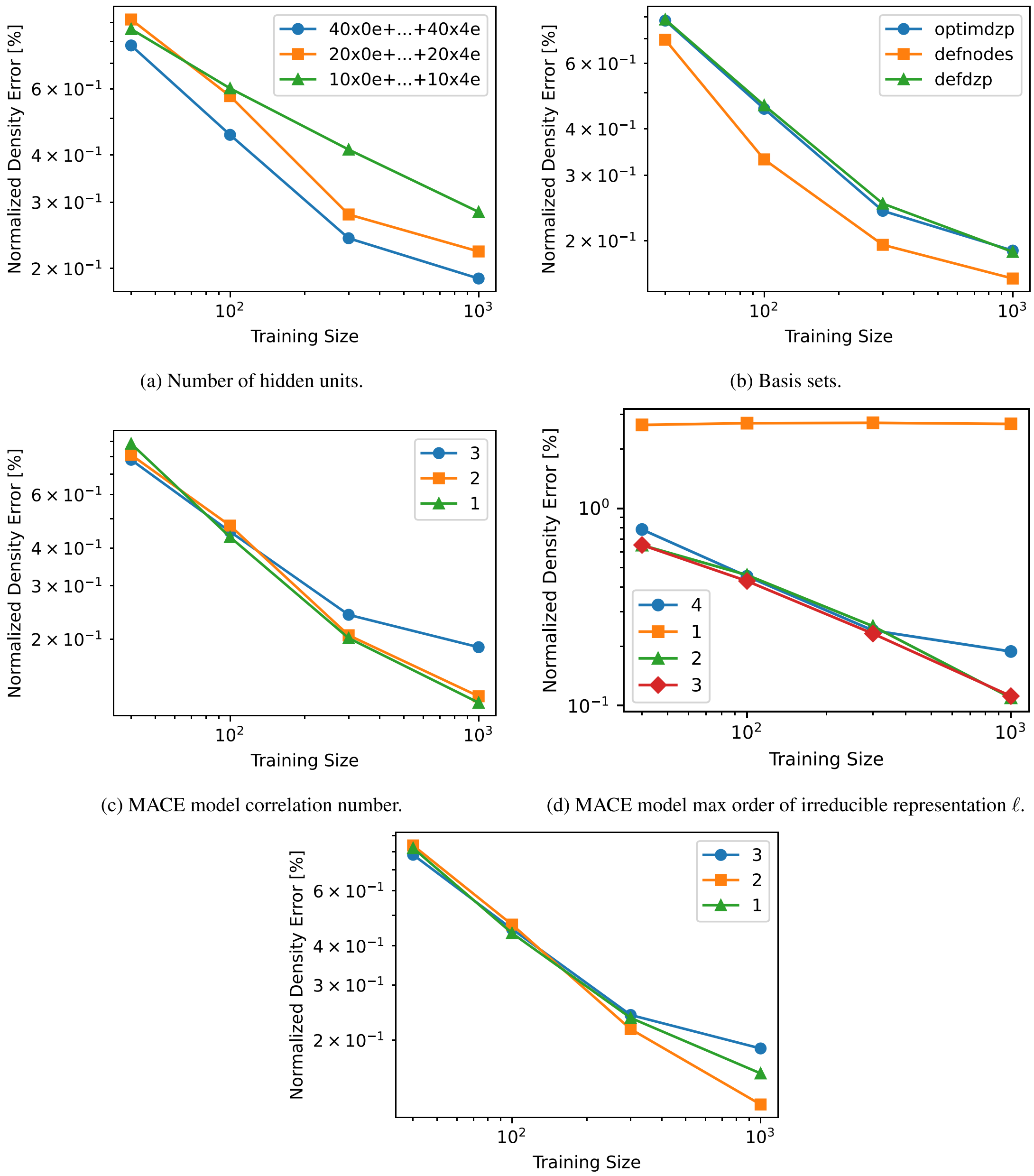
When designing the overall model architecture there are a number of hyperparameters that one needs to choose. For example, in the MACE article [13] the effect of the body order (ν, also referred to as MACE model correlation number) and the order of equivariance of the internal representation (ℓ) were shown to be important for the accuracy of the ML potential. It was found that a larger body order ($\nu = 3$) resulted in a steeper slope of the learning curve (energy prediction MAE vs number of training examples) than using $\nu = 1$ as in previous equivariant graph networks and the slope was not sensitive to the choice of maximum ℓ.

We conduct a number of similar experiments to investigate whether these conclusions carry over to DM prediction as well. Due to the big number of runs required, we only assess the effects on the smallest dataset, i.e. MD17 aspirin. The results of these experiments are shown in figure 3. Our error metric is the normalized density error (NDE), which is computed as:

$$\mathrm{NDE} = \frac{\int |\rho(\vec{r}) - \rho_{\mathrm{pred}}(\vec{r})| \mathrm{d}\vec{r}}{\int \rho(\vec{r}) \mathrm{d}\vec{r}} \qquad (3)$$

where ρ_{pred} is the predicted electron density.

The model is not very sensitive to architecture hyperparameters for the MD17 aspirin dataset, and higher accuracy can only be reached with higher number of hidden units.



(a) Number of hidden units.

(b) Basis sets.

(c) MACE model correlation number.

(d) MACE model max order of irreducible representation ℓ.

(e) Number of message passing interaction steps.

Figure 3. Learning curves for the model trained on the MD17 aspirin dataset with different model hyperparameters.

3.3.1. Number of hidden units

In the first experiment (figure 3(a)) we vary the number of hidden units, (effectively the number of parameters in the network). We observe that a larger number of parameters yields a more accurate model. We do not observe learning saturation even with the largest model. Therefore, further increase of the number of parameters is expected to give even better accuracy. However, in this experiment, we have reached the RAM limitations of the RTX 3090 (24GB). Further exploration would require a card with more memory or distributing the training across multiple GPUs.

3.3.2. Basis set

In the second experiment we keep the model hyperparameters the same, but we change the basis set used to compute the dataset. *defdzp* refers to the default split valence DZP basis, *optimdzp* is a split valence DZP optimized for aspirin (optimized cut-off radii), and *defnodes* is the default nodes DZP basis, which makes basis functions of the same atom orthogonal [28, 35]. As seen in figure 3(b), optimizing the basis set does not make the data easier to learn for the model. However, the model performs better with the nodes basis. This might be because of the increased orthogonality of the basis set or simply because the basis set is more similar to the radial functions used by the MACE model (bessel functions).

3.3.3. Correlation number

In the third experiment we vary the MACE model correlation number while keeping other parameters fixed. This effectively changes the body order of the messages. As seen on figure 3(c) we do not see any improvement with increased body order. This is in contrast to the MACE article results, where an increasing body order makes the learning curve steeper and learning more effective. It is possible that more complex systems can benefit from increasing the correlation number to learn the DM.

3.3.4. Representation order (ℓ)

In the fourth experiment we sweep the maximum order of the irreducible representation ℓ, while keeping the total number of learnable parameters constant (adjusting number of hidden units). Since the DZP basis set for aspirin contains d ($\ell = 2$) orbitals, it is expected that $\ell = 1$ is not enough to learn the DM. This is clear from figure 3(d). Increasing ℓ further does not improve expressivity or learning when the number of learnable parameters are kept constant.

3.3.5. Number of message passing steps

In the fifth experiment we change the number of message passing steps. This variable, along with the neighbor cutoff radius, determines how far interactions we can model. As shown in figure 3(e), 2 interaction steps is slightly better than just 1. We see a small decrease in accuracy for 3 interactions. The aspirin molecule is small enough that there are no neighbors beyond the distance spanned by two interactions. A larger number of interactions makes the network deeper and the optimization might therefore be more difficult. The lack of improvement in more message passing steps can also be due to message squashing [36].

3.4. Learning curves on larger scale benchmarks

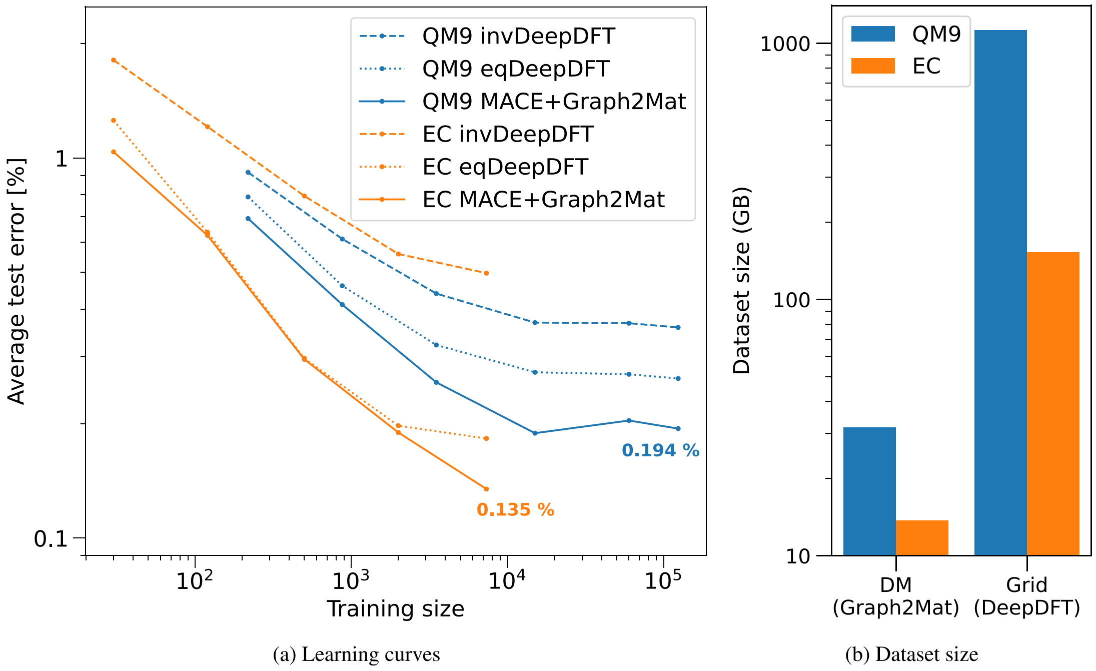
To evaluate the performance of MACE+Graph2Mat at predicting densities, we compute learning curves for our two larger scale datasets: QM9 and ethylene carbonate. We compare the NDE of our models (equation (3)) to that of DeepDFT, which is grid-based. As displayed in figure 4(a), errors are very similar for the same number of training examples, although MACE+Graph2Mat seems to give slightly better performance. However, the matrix-based approach is much more data efficient and inference can be much faster for large systems. Figure 4(b) shows how the DM approach can be trained with a much smaller dataset (being both data efficient and a lightweight representation) to produce very similar results. The QM9 DM dataset is more than 1 order of magnitude smaller than the grid dataset, while EC shows a 1 order of magnitude decrease. The difference in the benefits for both systems is due to QM9 molecules being surrounded by vacuum, which makes the sparse representation even more efficient.

3.5. Integration with DFT: computing properties from density

We use the best hyperparameters found in section 3.3 to test the performance of the model on QM9 as a function of training size, and with a fixed test set of 10k structures. The results are shown in figure 5. Following, we discuss each assessed property individually.

3.5.1. Number of SCF iterations

Although the time scaling of a DFT code depends mainly on the algorithms within an SCF iteration, the number of iterations introduces a prefactor which is not negligible, especially for big systems. Figure 5 shows how using the predicted DMs as an initial guess for SIESTA reduces the number of iterations needed to reach convergence. The models trained with the bigger training sets show an average decrease in the number of iterations of around 40% with respect to initializing the DM from atomic densities. With a training set of just 218 molecules, the model already decreases by 27% on average the amount of iterations needed. This is a remarkable outcome of this model as the amount of internal computation in DFT gets reduced while to a user the results obtained are identical to a regular DFT calculation, without risk of spurious unreliable results even for out-of-distribution systems. We foresee that a baseline foundation model (including all elements) can be distributed with DFT codes to construct initial density guess, enabling significant reduction in HPC resources used for DFT. An important remark is that the QM9 molecules tested here are so small that the reduction in SCF steps does not translate into drastically faster simulations. Running on 1 CPU, the ML prediction of the density introduces a total overhead of around 1–2 s (I/O time + inference time), while each SCF step in SIESTA takes around 0.2 s (running on 10 CPUs), with an average of 12 SCF iterations needed to converge from scratch. However, this is not a big concern because (1) the ML model could be run on GPU



(a) Learning curves

(b) Dataset size

Figure 4. Comparison with DeepDFT (a grid based approach) on two large scale benchmarks: QM9 in blue and EC in orange. We compare to both invariant DeepDFT (invDeepDFT) and equivariant DeepDFT (eqDeepDFT). (a) Shows the learning curves with respect to the number of structures used for training. The performance is very similar, although MACE+Graph2Mat performs a bit better. (b) Shows the comparison of dataset size when using the DM vs when using grids to store the density. The compressed size is displayed for grids, since DeepDFT workflows uncompress them on the fly.

for faster inference, (2) the ML model could be optimized beyond the proof of concept implementation reported here and (3) in bigger systems the linear scaling of the ML model will beat the cubic scaling of the DFT algorithm used in these tests.

3.5.2. Single SCF step energy

We can also utilize the predicted DMs as the true density in DFT to compute other properties, such as the total energy. Figure 5 shows the mean energy error going from 2.2 eV for the 218 molecules training set to 0.019 eV for the largest training sets. The error of the largest training sets is comparable to state-of-the-art ML interatomic potentials, which have reported mean absolute errors of around 5 meV [37–39]. An interesting difference from ML potentials is that the energy errors reported in this work have all the same sign. The variational principle ensures that an unconverged (but idempotent and correctly scaled) DM will always give higher energy than the converged solution.

3.5.3. Dipole moment

Dipole moments for molecules can be computed directly without going through the DFT workflow by evaluating $Tr[DM * P]$, where Tr represents the trace, $*$ means matrix multiplication and P is the expected value of the position operator $P_{ij} = \langle \phi_i | \vec{r} | \phi_j \rangle$. This operation is linear scaling since the full matrix multiplication is not needed, just its diagonal elements. P_{ij} has the same sparsity as the overlap, therefore only the sparse part of the DM (what we predict) is needed. With this approach, figure 5 shows a MAE of 0.13 Debye for the larger training sets, which is one order of magnitude higher than published end-to-end approaches [38, 39], but still makes for a useful indicator of a molecule's dipole.

3.5.4. Mulliken charges

Mulliken charges [40] can also be computed directly from the DM using the overlap. We compute them with the sisl python package [41]. With this approach, the smallest training set in figure 5 shows a MAE of 0.015 e^-, while the MAE converges at around 0.003 e^- with training size.

For all properties, there are some large outliers, especially when using the smallest training set. These outliers are common across properties and correspond to structures that are ill represented in the training set. For example, the training set of 218 molecules does not contain structures with F and the obvious outliers (energy errors above 100 eV) correspond to molecules with 3 F atoms. Such outliers distort the mean error, and it might be better to look at the median of the errors to understand how the models perform. In

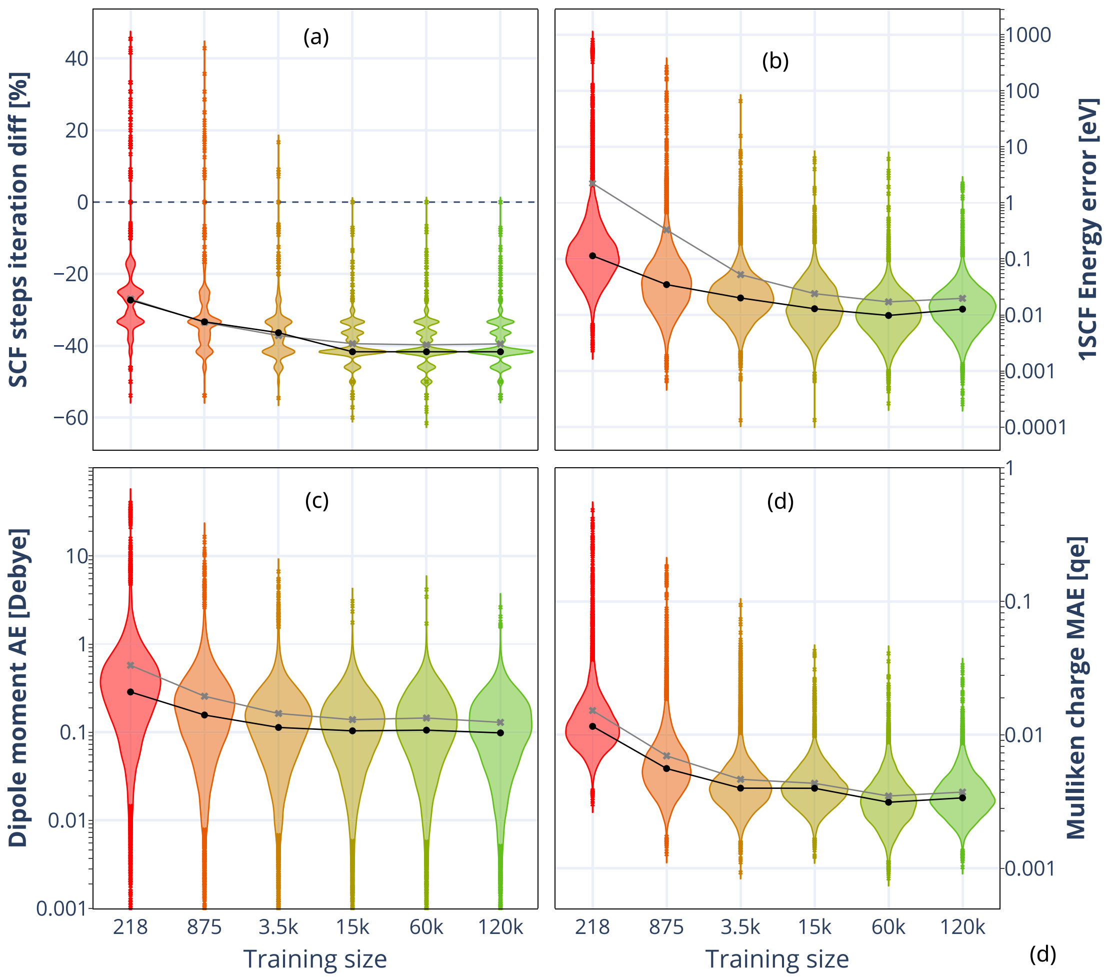


Figure 5. Performance of QM9 models in various downstream tasks in the test set as a function of training set size. Violin plots show the distribution while the gray/black lines shows the mean/median. (a) SCF steps taken to converge the SIESTA run starting from the predicted DM (vs starting from atomic densities). (b) Error made on the total energy if we just do 1 SCF step in SIESTA from the predicted DM. (c) Absolute error on the magnitude of the dipole moment. (d) Mean absolute error of the Mulliken charges in a molecule. The top two plots are from integration of the model within DFT, while the bottom ones are directly computed from the DM.

section 3.6 we discuss how to identify these outliers. It is also notable in figure 5, in conjunction with figure 4, that increasing the training set from 60k to 120k molecules does not seem to significantly increase the performance of the model.

3.6. Inherent uncertainty

Using the MACE+Graph2Mat models as a source of initial electron densities for DFT is completely safe, since DFT will converge and therefore end up correcting any error of the model. However, when using the models to directly predict properties (electron density, partial charges, energy, forces…) there is a need to assess the probability of the prediction being wrong. Predictions are often wrong for chemical space regions where the model is uncertain. Therefore, it is crucial to have methods to estimate the uncertainty of ML models. Ensemble based methods are very popular because they work across model architectures and application fields, but they are expensive to evaluate as the resources (time and memory) needed to train and run inference are multiplied by the size of the ensemble. In this section we show how the uncertainty of the MACE+Graph2Mat models when predicting density matrices can be estimated without the need for model ensembles. In particular, the error of the model on the total number of electrons is a computationally inexpensive uncertainty metric, while self-consistency metrics can be used in cases where running 1 SCF step is reasonable or necessary for a particular downstream task. In figure 6 we correlate these metrics with the error made on the energy and forces computed directly from the predicted density.

Charge neutrality (or any other constraint on the number of electrons) is not enforced on the models. The first impression might be that building a total charge constraint into the model is clearly beneficial. However, the fact that the model is unconstrained in terms of the total charge provides us with a measure of

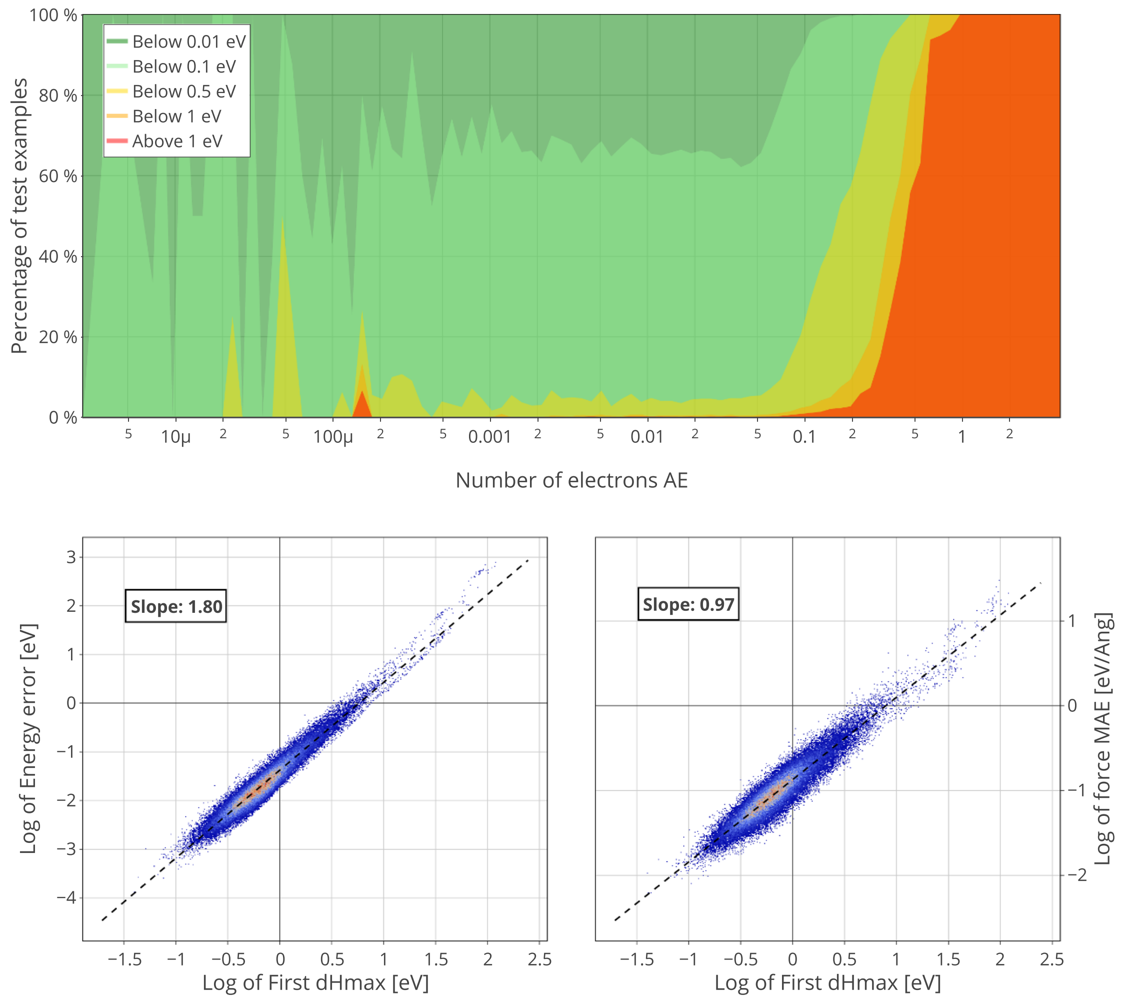


Figure 6. Two different (top and bottom) uncertainty proxies from the QM9 test set. Data from all training sizes is used to generate the figures. (Top) For a given error on the number of electrons, percentage of test examples that are below a certain energy error threshold. The number of electrons of a DM can be computed without any SCF step. DMs are rescaled to the correct number of electrons before being fed into the DFT machinery to compute energies. (Bottom) Correlation of the maximum self consistency error of the Hamiltonian at the first SCF step with the error of 1-SCF energies and forces. The self consistency error is only available after 1 SCF step. Since the axes are logarithmic, the slope of the linear fit indicates the degree of the polynomial that correlates the X and Y variables.

uncertainty. Figure 6 shows how the error that the model makes on the number of electrons is clearly correlated to the energy error. However, errors of less than approximately 0.05 electrons do not seem to cause any significant variance on the energy errors. Only after that threshold the number of electrons actually becomes a good proxy for estimating the accuracy of the predicted DM. This uncertainty metric based on simple charge counting imposes no training requirements and does not increase inference time.

While the number of electrons is a '0-SCF' proxy for the uncertainty of the model, there are other useful proxies that come up if performing 1 SCF step within the DFT framework is reasonable or even part of the workflow. These are particularly useful when the model is deployed for molecular dynamics, where 1 SCF iteration is required to compute the forces because of the non-orthogonality of the basis set (appendix A in [18]). The best metric that we have found for this job is the maximum error on the self-consistency of the Hamiltonian matrix (dHmax). Figure 6 shows its correlation to energy and forces errors. The energy error changes quadratically with the self-consistency error, while the forces error changes linearly, as expected [42].

4. Discussion

We show that learning the DM directly is a good approach to predict electron densities with no performance drawbacks against the popular grid-based prediction models. When training density data are generated with codes based on atom-centered basis sets, where the DM is readily available and contains the exact density, learning the DM is more practical due to its superior data efficiency of at least one order of magnitude. The main drawback of the DM as a learning target is that it is basis set dependent. However, for a particular system, a basis set is typically optimized once and then kept fixed for the rest of the calculations. In molecular dynamics, for example, the basis set dependence is of no concern. On the other hand, it can be a limitation for high-throughput calculations over large chemical spaces that require basis set flexibility. This is a general drawback of atom-centered basis sets compared to plane-wave basis sets. Therefore, we do not expect this problem to discourage the use of our methodology by scientists who already use codes based on atomic orbitals.

Regarding the usefulness of predictions, DM predictions have mostly the same applications as direct density predictions. The primary cause driving the exploration of learning the electron density instead of particular properties is the idea that once we have the electron density we can compute all ground-state properties from it. This is certainly possible, as has been shown for example in [15]. Our own results support that DM prediction can be integrated within the DFT framework leading to computation of all observables. We observe good accuracy on the computation of properties like energy, mulliken charges or dipole moments with an accuracy similar but somewhat lower than end-to-end ML models that are optimized to predict specific properties. However, computing them through the density is a more robust approach, as for out-of-distribution cases the approach does not give spurious results like black box surrogates.

One source of robustness is the fact that properties are computed based on physics from the density and therefore are consistent with each other. If there is an error in the density, the same error will show on the rest of properties since they are correlated by physics equations, not by a learnable blackbox. This leaves us with many possible measures of uncertainty. The most straightforward but novel and unique uncertainty measure in this approach is the error on the total charge of the system, as the model is unaware of what are the constraints of physics with respect to that. We have shown that the error on the total charge correlates very well with the error on the total energy. We expect it to correlate with all other errors as well (e.g. the NDE). This is a powerful finding, because it removes the need for model ensembles or other more complex attempts at measuring uncertainty. When running molecular dynamics, we need to perform at least one SCF iteration to compute the forces due to the non-orthogonality of the basis set. In that scenario, we can access an even more powerful uncertainty metric: the Hamiltonian's self-consistent error. The combination of both metrics makes the model very promising to use for active learning applications, as it is aware of its own limitations (or in words of reference [43]: 'it knows what it does not know' and can avoid 'unknown unknowns').

5. Conclusion

The most general contribution from this work is the development of the SE(3) equivariant Graph2Mat framework that converts an atom-centered equivariant representation to a sparse matrix of spherical harmonics interactions, applicable to various electronic structure matrices such as the Hamiltonian (H), Overlap (S), and EDM. Going beyond electronic structure observables, it is versatile to be used to compute any equivariant matrix for a 3D point cloud with different point types, not limited to atomic systems. Graph2Mat can be coupled with a variety of graph representation learning models to create ML models for predicting physical observables. As a specific utility of Graph2Mat in learning electronic structure matrices, we have successfully used the DM as an alternative learning target for ML to predict electron density. We have coupled Graph2Mat with state-of-the-art molecular graph representation learning model MACE that converts atomic environments into node embeddings.

As opposed to density fitting approaches, the DM contains the exact density generated by a code that uses atom-centered basis functions. Compared to grid-based methods, learning the DM is much more data efficient. In the case of SIESTA, basis functions are of the form of a radial function times a spherical harmonic, a feature that introduces symmetry considerations which further enhance efficiency and robustness. QM9 and ethylene carbonate benchmarks show that the method can achieve performance similar to state-of-the-art grid-based approaches. The method presented here is universal and the fact that it works for the periodic ethylene carbonate system shows that our models treat periodicity correctly. Therefore, there is no foreseeable problem with using it for all kind of systems like molecular/liquid/solid/solid–liquid interfaces. Multiple properties (directly accessible or obtained via one DFT SCF step) can be reliably calculated with the predicted DM. When the model is seamlessly integrated into a DFT code it significantly reduces the number of iterations needed to fully converge the SCF loop. Further work in the electronic structure domain can focus on developing foundational models for DM prediction to be used as initial DM estimators for SIESTA to make all future DFT simulations significantly more efficient. On a broader perspective, our representation+Graph2Mat ML approach can be extended to other graph-based physics simulation surrogates as well in the future.

Data availability statement

The data that support the findings of this study are openly available at the following URL/DOI: https://doi.org/10.11583/DTU.c.7310005.

Acknowledgments

PF has received funding from the Project PID2022-139776NB-C62 funded by MCIN/AEI https://doi.org/10.13039/501100011033 ERDF, EU and the Grant PRE2019-089784 funded by MCIN/AEI https://doi.org/10.13039/501100011033 and by ESF Investing in your future. This work has been carried in the framework of the Materials Science PhD Program at Universitat Autònoma de Barcelona (UAB). AB was supported by Independent Research Fund Denmark with project DELIGHT (Grant No. 0217-00326B) TeraBatt project (Grant No. 2035-00232B) and ADANA project (Grant No. 3164-00297B). AG and MP were supported by Grant PID2022-139776NB-C61 of the Spanish MCIN/AEI https://doi.org/10.13039/501100011033, by the ERDF, A way of making Europe, and by Generalitat de Catalunya (Grant 2021SGR01519). AG acknowledges AEI through the Severo Ochoa MaTrans42 (CEX2023-0001263-S) Excellence Centre distinction, and the European High Performance Computing Joint Undertaking under grant Agreement no. 101093374 (Materials design at the exascale), co-funded by AEI PCI2022-134978-2. This project has received funding from the European Union's project Battery Interface Genome—Materials Acceleration Platform: EU-H2020 Grant Agreement ID: 957189. PF, MP, AG and PO acknowledge support from EU MaX CoE (Grant 101093374) and Grants PCI2022-134972-2, PCI2022-134978-2, PID2022-139776NB-C61 and PID2022-139776NB-C62 funded by the Spanish MCIN/AEI https://doi.org/10.13039/501100011033 and by the ERDF, A way of making Europe, and Grant 2021SGR01519 from Generalitat de Catalunya. The ICN2 is supported by the Severo Ochoa Centres of Excellence programme, Grant No. CEX2021-001214-S, funded by MCIN/AEI https://doi.org/10.13039/501100011033.

ORCID iDs

Pol Febrer https://orcid.org/0000-0003-0904-2234
Peter Bjørn Jørgensen https://orcid.org/0000-0003-4404-7276
Miguel Pruneda https://orcid.org/0000-0002-3621-6095
Alberto García https://orcid.org/0000-0001-5138-9579
Pablo Ordejón https://orcid.org/0000-0002-2353-2793
Arghya Bhowmik https://orcid.org/0000-0003-3198-5116

References

[1] Huang B, von Rudorff G F and von Lilienfeld O A 2023 The central role of density functional theory in the AI age *Science* **381** 170–5

[2] Marzari N, Ferretti A and Wolverton C 2021 Electronic-structure methods for materials design *Nat. Mater.* **20** 736–49

[3] Unke O T, Chmiela S, Sauceda H E, Gastegger M, Poltavsky I, Schuc̈ctt K T, Tkatchenko A and Muc̈cller K R 2021 Machine learning force fields *Chem. Rev.* **121** 10142–86

[4] Friederich P, Häse F, Proppe J and Aspuru-Guzik A 2021 Machine-learned potentials for next-generation matter simulations *Nat. Mater.* **20** 750–61

[5] Vita J A and Schwalbe-Koda D 2023 Data efficiency and extrapolation trends in neural network interatomic potentials *Mach. Learn.: Sci. Technol.* **4** 035031

[6] Yang X *et al* 2024 Curator: building robust machine learning potentials for atomistic simulations autonomously with batch active learning *ChemRxiv* (https://doi.org/10.26434/chemrxiv-2024-p5t3l)

[7] Busk J, Jørgensen P B, Bhowmik A, Schmidt M N, Winther O and Vegge T 2021 Calibrated uncertainty for molecular property prediction using ensembles of message passing neural networks *Mach. Learn.: Sci. Technol.* **3** 015012

[8] Jørgensen P B and Bhowmik A 2022 Equivariant graph neural networks for fast electron density estimation of molecules, liquids and solids *npj Comput. Mater.* **8** 1–10

[9] Bogojeski M, Vogt-Maranto L, Tuckerman M E, Müller K R and Burke K 2020 Quantum chemical accuracy from density functional approximations via machine learning *Nat. Commun.* **11** 5223

[10] Rackers J A, Tecot L, Geiger M and Smidt T E 2023 A recipe for cracking the quantum scaling limit with machine learned electron densities *Mach. Learn.: Sci. Technol.* **4** 015027

[11] Li C, Sharir O, Yuan S and Chan G K 2024 Image super-resolution inspired electron density prediction (arXiv:2402.12335)

[12] Karniadakis G E, Kevrekidis I G, Lu L, Perdikaris P, Wang S and Yang L 2021 Physics-informed machine learning *Nat. Rev. Phys.* **3** 422–40

[13] Batatia I, Kovacs D P, Simm G, Ortner C and Csányi G 2022 Mace: Higher order equivariant message passing neural networks for fast and accurate force fields *Advances in Neural Information Processing Systems* vol 35 pp 11423–36 (available at: https://proceedings.neurips.cc/paper_files/paper/2022/hash/4a36c3c51af11ed9f34615b81edb5bbc-Abstract-Conference.html)

[14] Chandrasekaran A, Kamal D, Batra R, Kim C, Chen L and Ramprasad R 2019 Solving the electronic structure problem with machine learning *npj Comput. Mater.* **5** 22

[15] Koker T, Quigley K, Taw E, Tibbetts K and Li L 2024 Higher-order equivariant neural networks for charge density prediction in materials *npj Comput. Mater.* **10** 161

[16] Grisafi A, Lewis A M, Rossi M and Ceriotti M 2022 Electronic-structure properties from atom-centered predictions of the electron density *J. Chem. Theory Comput.* **19** 4451–60

[17] Cuevas-Zuviría B and Pacios L F 2021 Machine learning of analytical electron density in large molecules through message-passing *J. Chem. Inform. Modeling* **61** 2658–66

[18] Soler J M, Artacho E, Gale J D, García A, Junquera J, Ordejón P and Sánchez-Portal D 2002 The siesta method for ab initio order-n materials simulation *J. Phys.: Condens. Matter* **14** 2745

[19] Blum V, Gehrke R, Hanke F, Havu P, Havu V, Ren X, Reuter K and Scheffler M 2009 Ab initio molecular simulations with numeric atom-centered orbitals *Comput. Phys. Commun.* **180** 2175–96

[20] Batzner S, Musaelian A, Sun L, Geiger M, Mailoa J P, Kornbluth M, Molinari N, Smidt T E and Kozinsky B E 2022 (3)-equivariant graph neural networks for data-efficient and accurate interatomic potentials *Nat. Commun.* **13** 2453

[21] Unke O, Bogojeski M, Gastegger M, Geiger M, Smidt T and Müller K R 2021 Se (3)-equivariant prediction of molecular wavefunctions and electronic densities *Advances in Neural Information Processing Systems* vol 34 pp 14434–47 (available at: https://proceedings.neurips.cc/paper/2021/hash/78f1893678afbeaa90b1fa01b9cfb860-Abstract.html)

[22] Zhang L, Onat B, Dusson G, McSloy A, Anand G, Maurer R J, Ortner C and Kermode J R 2022 Equivariant analytical mapping of first principles Hamiltonians to accurate and transferable materials models *npj Comput. Mater.* **8** 158

[23] Zhong Y, Yu H, Su M, Gong X and Xiang H 2023 Transferable equivariant graph neural networks for the Hamiltonians of molecules and solids *npj Comput. Mater.* **9** 182

[24] Nigam J, Willatt M J and Ceriotti M 2022 Equivariant representations for molecular Hamiltonians and n-center atomic-scale properties *J. Chem. Phys.* **156** 014115

[25] Shao X, Paetow L, Tuckerman M E and Pavanello M 2023 Machine learning electronic structure methods based on the one-electron reduced density matrix *Nat. Commun.* **14** 6281

[26] Hazra S, Patil U and Sanvito S 2024 Predicting the one-particle density matrix with machine learning *J. Chem. Theory Comput.* **20** 4569–78

[27] Geiger M and Smidt T 2022 e3nn: Euclidean neural networks (arXiv:2207.09453)

[28] Artacho E, Sánchez-Portal D, Ordejón P, Garcia A and Soler J M 1999 Linear-scaling ab-initio calculations for large and complex systems *Phys. Status Solidi* b **215** 809–17

[29] García A *et al* 2020 Siesta: Recent developments and applications *J. Chem. Phys.* **152** 204108

[30] Van Setten M J, Giantomassi M, Bousquet E, Verstraete M J, Hamann D R, Gonze X and Rignanese G M 2018 The pseudodojo: Training and grading a 85 element optimized norm-conserving pseudopotential table *Comput. Phys. Commun.* **226** 39–54

[31] Chmiela S, Tkatchenko A, Sauceda H E, Poltavsky I, Kristof T T and Klaus-Robert M 2017 Machine learning of accurate energy-conserving molecular force fields *Sci. Adv.* **3** e1603015

[32] Ramakrishnan R, Dral P O, Rupp M and von Lilienfeld O A 2014 Quantum chemistry structures and properties of 134 kilo molecules *Sci. Data* **1** 140022

[33] Jørgensen P B and Bhowmik A 2022 *Ethylene Carbonate Molecular Dynamics Dataset* (available at: https://doi.org/10.11583/DTU.16691825)

[34] Bhowmik A and Jørgensen P B 2024 Density matrix, Hamiltonian matrix, overlap matrix and energy density matrix of qm9 and ethylene carbonate dataset (available at: https://doi.org/10.26434/chemrxiv-2024-j4g21)

[35] Junquera J, Paz O, Sánchez-Portal D and Artacho E 2001 Numerical atomic orbitals for linear-scaling calculations *Phys. Rev.* B **64** 235111

[36] Di Giovanni F, Giusti L, Barbero F, Luise G, Lio P and Bronstein M M 2023 On over-squashing in message passing neural networks: the impact of width, depth and topology *Int. Conf. on Machine Learning* (PMLR) pp 7865–85 (available at: https://proceedings.mlr.press/v202/di-giovanni23a.html)

[37] Musaelian A, Batzner S, Johansson A, Sun L, Owen C J, Kornbluth M and Kozinsky B 2023 Learning local equivariant representations for large-scale atomistic dynamics *Nat. Commun.* **14** 579

[38] Schütt K, Unke O and Gastegger M 2021 Equivariant message passing for the prediction of tensorial properties and molecular spectra *Int. Conf. on Machine Learning* (PMLR) pp 9377–88 (available at: https://proceedings.mlr.press/v139/schutt21a.html)

[39] Zhang S, Liu Y and Xie L 2023 A universal framework for accurate and efficient geometric deep learning of molecular systems *Sci. Rep.* **13** 19171

[40] Mulliken R S 1955 Electronic population analysis on LCAO–MO molecular wave functions. I *J. Chem. Phys.* **23** 1833–40

[41] Papior N 2024 sisl: v0.14.3 (https://doi.org/10.5281/zenodo.597181)

[42] Chan C T, Bohnen K P and Ho K 1993 Accelerating the convergence of force calculations in electronic-structure computations *Phys. Rev.* B **47** 4771

[43] Logan D C 2009 Known knowns, known unknowns, unknown unknowns and the propagation of scientific enquiry *J. Exp. Bot.* **60** 712–4