Journal of Cheminformatics

RESEARCH Open Access

LAGNet: better electron density prediction for LCAO-based data and drug-like substances

Konstantin Ushenin[1,2*], Kuzma Khrabrov[1], Artem Tsypin[1], Anton Ber[1], Egor Rumiantsev[3] and Artur Kadurin[1,4]

Abstract

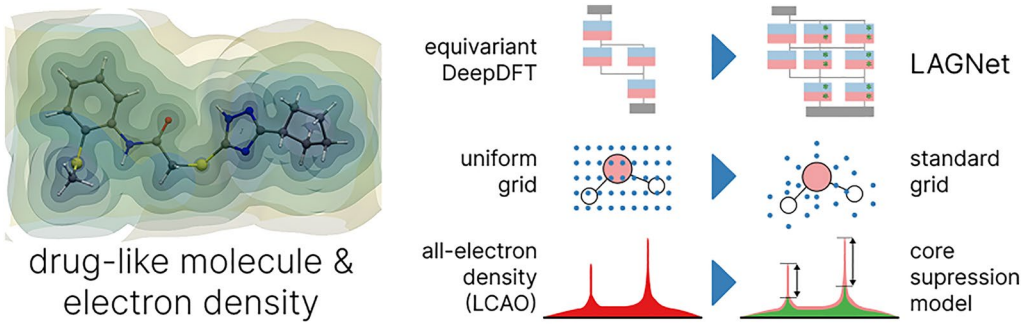
The electron density is an important object in quantum chemistry that is crucial for many downstream tasks in drug design. Recent deep learning approaches predict the electron density around a molecule from atom types and atom positions. Most of these methods use the plane wave (PW) numerical method as a source of ground-truth training data. However, the drug design field mostly uses the Linear Combination of Atomic Orbitals (LCAO) for computation of quantum properties. In this study, we focus on prediction of the electron density for drug-like substances and training neural networks with LCAO-based datasets. Our experiments show that proper handling of large amplitudes of core orbitals is crucial for training on LCAO-based data. We propose to store the electron density with the standard grids instead of the uniform grid. This allowed us to reduce the number of probing points per molecule by 43 times and reduce storage space requirements by 8 times. Finally, we propose a novel architecture based on the DeepDFT model that we name LAGNet. It is specifically designed and tuned for drug-like substances and ∇^2DFT dataset.

Scientific contribution We propose a core suppression model to correctly handle core orbitals and train neural network on LCAO-based data with atoms of the 3rd and 4th periods. We show that using the standard grid instead of the uniform grid drastically reduces the number of electron density probing points and data storage requirements. Finally, we propose the LAGNet model that allows to get better results on drug-like substances than the equivariant DeepDFT model.

Keywords Electron density, Deep learning, Quantum chemistry, Linear combination of atomic orbitals

*Correspondence:
Konstantin Ushenin
konstantin.ushenin@urfu.ru
Full list of author information is available at the end of the article

Graphical abstract



Introduction

The electron density is a fundamental quantity in quantum chemistry, as it uniquely determines the ground-state of a quantum system and directly enters the expression for total energy in density functional theory (DFT) [1]. Beyond its theoretical importance, electron density is used for many computational analysis in drug design.

The electron density enables visualization of molecules and computation of molecular descriptors. For example, molecular electrostatic potentials (MEPs) derived from electron density reveal charge distribution and identify key interaction sites for hydrogen and halogen bonding, chemical reactivity, binding affinity, and other biophysical properties [2–4]. Additionally, quantum theory of atoms in molecules (QTAIM) [5–7] allows researchers to characterize and quantify intermolecular interactions in a drug-target complex at the electron density level [7]. The magnitude of electron density at these intermolecular bond critical points correlates with interaction strength and overall binding affinity [8]. The descriptors derived are suitable for improving the robustness of quantitative structure-activity relationship (QSAR) models [5, 7]. The study [9] directly shows that partial atom charges (i.e. Bader's partial charges) can be computed from the electron density predicted with a neural network without relying on traditional numerical methods. Moreover, the electron density can serve as an input for other machine learning methods. Computed electron density has been used as input data for models that predict protein-ligand binding affinity [8], host-guest interactions [10], virtual screening results [11], and molecular classification [12].

Recent studies have applied machine learning to predict electron density directly from molecular conformations [13–30] using machine learning and deep learning approaches. However, most of these studies use the QM9 dataset, and the ground truth data from plane-wave (PW) calculations with pseudopotentials. Moreover, such studies typically utilize uniform grids to represent the electron density in a discrete form.

QM9 [31] is a popular benchmark for machine learning in computational chemistry. However, the molecules in QM9 dataset are relatively small and only contain up to 9 heavy atoms. Consequently, this dataset does not cover a major part of the drug-like chemical space. Larger and more diverse datasets, such as ∇^2DFT [32], QMugs [33], SPICE [34], Frag20 [35], and OrbNet Denali [36], have recently emerged.

Deep learning approaches to electron density prediction use outputs of quantum chemistry software as ground-truth data for training. There are two main branches of numerical methods for computation. The plane-wave (PW) method decomposes the quantum system into plane waves. In practical applications, the PW methods use pseudopotentials that replace core electron orbitals with their approximations. The following software systems are PW-based: VASP [37], Quantum Espresso [38], GPAW [39], and cp2k [40]. Most deep learning approaches use PW-based data as the ground truth: [17–27, 41, 42]. We describe possible reasons to prefer PW-based methods in previous studies in the related work section 4. The other branch of numerical methods uses a local combination of atomic orbitals (LCAO). LCAO decomposes the quantum system into combinations of Gaussians and spherical harmonics. Unlike the PW-based method, the LCAO method usually takes into account all electron orbitals. This method is implemented in psi4 [43], pyscf [44], Turbomole [45], Gaussian [46], and other software packages.

The electron density is usually represented as a set of points in the 3D space with electron density values. The most common representation utilizes a uniform grid over a rectangular volume, that encompasses the molecule. Using such representation is mandatory for approaches that use convolutional neural networks [21, 22] and Fourier neural operators [23, 24]. Moreover,

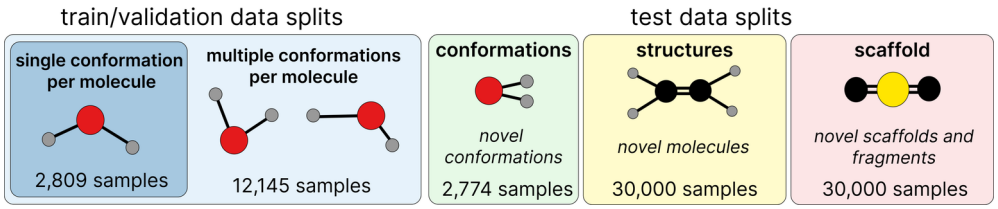


Fig. 1 This study employs a comprehensive partitioning of the ∇^2DFT dataset into training, validation, and test splits. Two training splits are defined: one containing a single conformation per molecule and another containing multiple conformations per molecule. Three progressively challenging test splits assess model generalization: the *conformation* split holds out novel conformations of molecules present in the training splits; the *structure* split holds out entirely new molecules not seen during training; and the *scaffold* split holds out molecules with unseen scaffolds

the PW-based method requires the uniform grid to efficiently compute electron density values [47]. However, the uniform grid is not an optimal discrete representation of function, if an integral over 3D space is required. LCAO-based methods use the standard grid which is a combination of the radial grid around each atom and a Lebedev grid.

In this work, we propose an approach to train neural networks on LCAO-based electron density data. The key difference between LCAO-based data and PW-based data is the presence of core electron orbitals, which create large variations in density values (from 10^{-20} to 10^{4}) and complicate training. To address this, we introduce a core suppression model. Also, we propose to store the electron density for training on the standard grid instead of the uniform grid. As a source of such molecules, we use the largest dataset of electron densities for commercially available drug-like substances ∇^2DFT [32]. Finally, we present the **L**ebedev-**A**ngular **G**rid Network (LAGNet), a new architecture based on DeepDFT and specifically designed to be trained on LCAO-based data, and to work with drug-like substances (such as those from the ∇^2DFT dataset). Our contributions are as follows:

1. We propose a core suppression model that reduces amplitude of core orbital. That is a data normalization approach critical for training on LCAO-based electron density data.
2. We propose using the standard grid to represent electron density data, reducing the number of points per molecule on average by 43 times, and cutting data storage requirements by 8 times.
3. We introduce LAGNet, a novel neural network architecture derived from the equivariant DeepDFT model. We extensively modify and tune it for ∇^2 DFT, achieving exceptional performance on drug-like substances.

This paper is organized as follows. Section 2 describes the data 2.1, grid sampling 2.2, the core suppression model 2.3, the invariant and equivariant DeepDFT 2.4, the LAGNet model 2.5, and performance metrics 2.6. Section 3 presents general observations on differences between electron density in QM9 and ∇^2DFT datasets 3.1, analyzes the effect of core orbital suppression 3.2, provides an ablation study for LAGNet 3.3, shows the advantages of SG-0 3.4,compares performance on various grids 3.5 before reporting final metrics 3.6, and shows metrics for cross-dataset evaluation 3.7. Section 4 reviews related works on electron density prediction and relevant datasets. Lastly, Sect. 5 discusses possible applications and limitations of this study and presents a conclusion.

Methods

Electron density dataset

We use the ∇^2 DFT dataset as a source of ground truth data for electron density. A sample with index i includes one molecule with a set of $\{z_j\}_{j\in[1,M_i]}$ atom numbers and their positions $\{\vec{\mathbf{r}}_j\}_{j\in[1,M(i)]}$. Dataset include molecules with various number of atoms ($M(i)$). For each molecule, the dataset includes the Hamiltonian (F, the Fock matrix, or the full Hamiltonian) and the overlap matrix (S). The solution of the generalized eigenvalue problem ($F\mathbf{c}_i = \lambda_i S$) provides a set of vectors ($\mathbf{c}_i$, $i \in [0, 1, \ldots, n]$). These vectors form the density matrix $D = CC^T$, $C = [\mathbf{c}_0, \mathbf{c}_1, \ldots, \mathbf{c}_d, \mathbf{0}_{d+1}, \ldots, \mathbf{0}_n]$, where $d = \frac{n}{2} = \frac{1}{2}\sum_{j\in[1,M]} z_j$ is a number of occupied orbitals. For a point $\vec{\mathbf{r}}$, atomic orbitals $\psi(\vec{\mathbf{r}})$ are defined by the geometry of the molecule and the basis-set (def2-SVP in the case of ∇^2DFT). The ground truth electron density is computed as follows: $\rho_{\text{lcao}}(\vec{\mathbf{r}}) = \psi(\vec{\mathbf{r}})^T D \psi(\vec{\mathbf{r}})$.

The ∇^2DFT dataset includes 1,936,931 unique molecules and 12,676,264 conformations. In order to simplify the usage, it is divided into several train and test splits with different sizes. For example, *tiny train split* contains 2,809 molecules and 12,145 conformations. These splits are accompanied by three test splits. A *tiny conformation*

test split includes novel conformations for molecules from the tiny train split. A *structure test split* includes novel molecules that were not present in the train split. A *tiny scaffold test split* includes novel molecules with scaffold that were not presented in the train split.

This study uses the tiny train split (2,809 samples, 12,145 conformations) to train neural network (see Fig. 1). The neural networks were trained with a single conformation per molecule (2,809 samples) or multiple conformations per molecule (12,145 samples, 4.32±3.83 conformations per molecule). Five percent of each training set was held out as a validation set to monitor performance and prevent overfitting. The testing of a neural network is performed with the conformation test split (2,774 samples, 2,747 molecules, 2,774 conformation, 1±0.12 conformations per molecule), the structure test split (30,000 samples) and the scaffold test split (30,000 samples).

In addition, we performed cross-dataset evaluation of the best checkpoint of LAGNet using part of QMugs [33] and the Hutchinson dataset [48]. This methods and results explained in detail in the Sect. 3.7.

Grid sampling

The key idea of [26, 27] is to train and validate neural networks with a small random subset of points from the uniform grid around a molecule. A subset of grid points is unique at each step. The authors of DeepDFT use a uniform grid and pick 1000 points of this grid for the training step and 5000 points for the validation step. This setup was designed for PW-based data with pseudopotentials. However, in the case of LCAO-based data (ie, ∇^2 DFT), a batch of training data may contain many points sampled from the core orbitals of Br, Cl, and S. Significant differences in statistical properties between batches can negatively affect training stability. The number of points in the uniform grid becomes notably high for molecules with more than 10 atoms. Therefore, storage of the uniform grid requires a significant amount of memory or disk space to store one sample. The numerical values and analysis for this issue are presented in the Result section 3.

LCAO-based solvers use a standard grid (SG) instead of the uniform grid. SG is a combination of the radial grid and Lebedev's grid. Each atom in a molecule is surrounded by a set of spheres (the radial grid) with equal step between the spheres. Each sphere contains a set of integration points that is angular grid. In particular, the

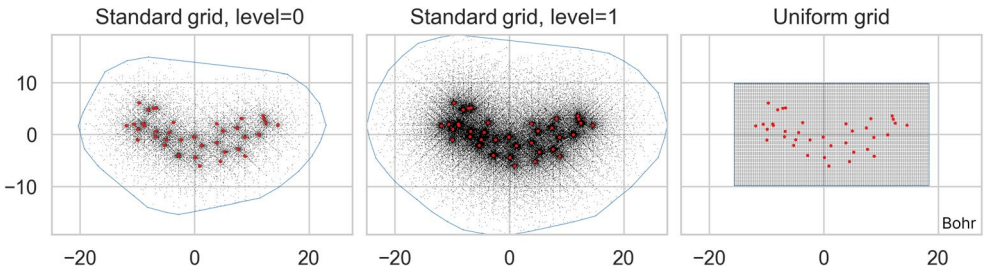


Fig. 2 Examples of different grid types for the same molecule projected onto the XY-plane. The left panel shows the standard grid at level 0, the middle panel shows the standard grid at level 1, and the right panel shows the uniform grid. Red points denote atomic positions, black dots denote probe points, and blue lines indicate the convex hull of each point set after projection. Distances are given in Bohr

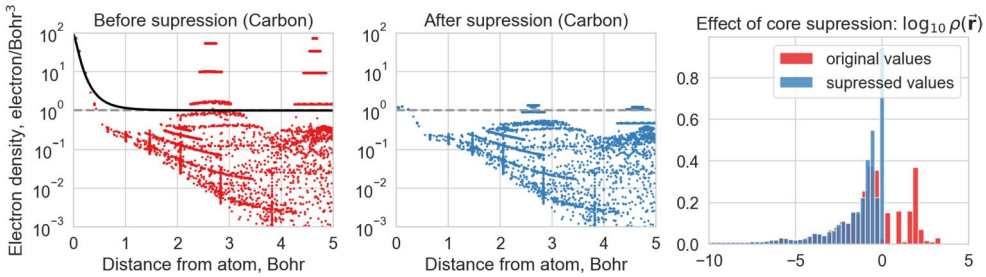


Fig. 3 Example of the core suppression method. The left panel shows the electron density around a single carbon atom in a sample molecule. The red line is the regression model for the core orbital and its local environment, $1 + \alpha(\vec{r})\beta(\vec{r})$. The center panel shows how the electron density changes after applying core suppression to all atoms. The right panel is a histogram comparing electron density before ($\rho_{\text{lcao}}(\vec{r})$, red) and after ($\rho(\vec{r})$, blue) core suppression

Lebedev grid is a particular type of angular grid. The LCAO solver uses the SG to compute integral over space with better precision than is possible with a uniform grid containing the same number of integration points. This makes the calculation of the exchange-correlation functional faster and reduces the total processing time for the LCAO solvers [47, 49, 50].

In this study, we combine the DeepDFT grid sampling and the SG. The usage of SG instead of a uniform grid allows us to train neural networks with the smaller size of the stored data and to increase the stability of the neural network training process. Examples of uniform and standard grids are shown in Fig. 2.

Usually, the SG is parametrized in software packages with the 'level of grid' property. Each level of the grid associates with a predefined number of spheres and the Lebedev grid order per atom type. The whole SG is a union of subgrids assigned to each atom in a molecule. In our study, we used grids from the pyscf/Turbomole software with default parameters and disabled pruning.

We preserve the SG-0 subgrid structure for each atom in our training inputs. During each iteration, our sampler randomly selects 5% of the points from every atom's SG-0 subgrid. Consequently, each batch contains 5% of the total SG-0 points, ensuring that sampling uniformly covers the local environment of all atoms. This strategy prevents batches from being dominated by points around a single atom or region of the molecule, resulting in more consistent statistical properties across iterations. For validation, we use the full SG-0 grid (100% of points).

For the test, this study uses SG-0, SG-1, and the uniform grid. The last has a margin of 2 Å and a step of 0.1 Å. The same uniform grid parameters were used in the datasets from [26, 27], and the following studies [21, 24].

Core suppression method

In contrast to computation with PW-basis and pseudopotentials, the LCAO computation directly simulates core orbitals and generates data with large values of the electron density near the heteroatoms. For the LCAO method, the magnitude of the electron density is approximately 0.7 el./Bohr^3 near the hydrogen atom and reaches 2500 el./Bohr^3 around a bromine atom. At the same time, the electron density at the point between two atoms is in the range of 0 to 1 el./Bohr^3. Thus, the distribution of the electron density values is notably skewed with extreme values on the right side of the distribution.

Our experiments show that training with the default implementation of the DeepDFT model is not stable for this type of data. To completely avoid issues with huge amplitudes of the electron density, we propose a core orbital suppression model that decreases the electron density around the atom positions and increases the neural networks training stability. The effect of the model is shown in Fig. 3.

To approximate the electron density around the atom, we propose a radial basis function that is general enough to represent the atom neighborhood. To build the function, we start from $\alpha(\vec{\mathbf{r}}) = 1/(1 + \varepsilon\|\vec{\mathbf{r}}\|_2)$, which is the classic inverse quadric function for the radial basis function interpolation, and ε is a hyperparameter. Then we generalize it to the function $\alpha(\vec{\mathbf{r}}) = \varepsilon/P_k(\|\vec{\mathbf{r}}\|_2)$, where ε is a trainable coefficient, and $P_k(x) = \varepsilon_0^2 + \varepsilon_1^2 x + \varepsilon_2^2 x^2 + \cdots + \varepsilon_k^2 x^k$ is a polynomial of the degree k with a set $\{\varepsilon\}_0^{i=k}$ of $k+1$ trainable coefficients. Since the coefficients are raised to the second power, the polynomial is strictly positive ($P_k(x) > 0$). If $\varepsilon > 0$, then $\alpha(\vec{\mathbf{r}}) > 0$.

The set of coefficients $(\varepsilon, \varepsilon_0, \varepsilon_1, \ldots, \varepsilon_k)$ is fitted for each atom type z_i separately. In this work, we use a degree 4 polynomial and employ the L-BFGS method from *scipy.optimize* [51] to train such a model on 333 randomly chosen molecules from the dataset. To verify the stability of the regression model, we performed 30 runs with various subsets of molecules, and the convergence was successful for all subsets in all runs.

In order to localize the effect of the model inside the covalent atomic radii ($r_{cov}(z_i)$), we additionally introduce a soft cutoff function that limits the effect of the first model: $\beta(\vec{\mathbf{r}}) = 1/(1 + \exp(r_{\text{cov}}(z_i) - \|r_i\|_2)$. To avoid disruptions of the target function, we use the multiplicative form for data modification instead of subtraction of the regressed values. Multiplication by $\mu(\vec{\mathbf{r}}) = 1/\left(1 + \alpha_i(\vec{\mathbf{r}})\beta_i(\vec{\mathbf{r}})\right)$ suppresses the core orbitals and the density in the neighborhood of the atoms. The core orbital suppression model, direct and inverse data transformation finally look like this:

$$\alpha_i(\vec{\mathbf{r}}) = w/P_k(\|\vec{\mathbf{r}}_i\|_2),\ \beta_i(\vec{\mathbf{r}}) = \frac{1}{1 + \exp(r_{\text{cov}}(z_i) - \|\vec{\mathbf{r}}_i\|_2)}$$

$$\rho(\vec{\mathbf{r}}) = \mu(\vec{\mathbf{r}})\rho_{\text{lcao}}(\vec{\mathbf{r}}),\ \mu(\vec{\mathbf{r}}) = \prod_{i \in M} \frac{1}{1 + \alpha_i(\vec{\mathbf{r}})\beta_i(\vec{\mathbf{r}})}$$

$$\rho_{\text{lcao}}(\vec{\mathbf{r}}) = \mu^{-1}(\vec{\mathbf{r}})\rho(\vec{\mathbf{r}}),\ \mu^{-1}(\vec{\mathbf{r}}) = \prod_{i \in M} 1 + \alpha_i(\vec{\mathbf{r}})\beta_i(\vec{\mathbf{r}})$$

In the last expression, ρ_{lcao} is an original density function of the LCAO method, ρ is a target value for the training of neural networks. The proposed method is similar to an isotropic exponential function method, which was proposed in [42]. The main advantages of the proposed method are a more universal type of function that can be extended to more complex datasets with heavier heteroatoms and a straightforward approach to fit the model coefficients.

Viewed from a machine-learning perspective, the core-suppression method functions as a data-normalization technique tailored to LCAO-based electron densities. During training, the core-suppression model is applied to the raw data to improve convergence speed and stability of the neural network. At inference time, the inverse transformation is applied to the network's output, ensuring that all evaluation metrics compare predictions against the original target values (with full core orbital amplitudes).

Invariant and equivariant DeepDFT

Denote $\mathbf{h}, \mathbf{d}$ as abstract tuples of tensor values and $\mathfrak{g}$ as a group action. The function f is invariant if the group action applied to the input does not affect the output: $\mathbf{h} = f(\mathfrak{g}.\mathbf{d})$. The function f is equivariant if the group action applied to the input and output keeps the following expression: $\mathfrak{g}.\mathbf{h} = f(\mathfrak{g}.\mathbf{d})$. Denote $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ as a geometric graph with vertices $v \in \mathcal{V}$ and edges $e_{uv} \in \mathcal{E}$; $u, v \in \mathcal{V}$. Each vertex v has a position in the space $\vec{\mathbf{r}}$, a tuple of scalar features $\mathbf{s}$, and a tuple of vector features $\mathbf{v}$: $\mathbf{d}_v \equiv (\vec{\mathbf{r}}, \mathbf{s}, \mathbf{v})_v$, $v \in \mathcal{V}$, $\mathbf{s} \in \mathbb{R}^{1\times F}$, $\mathbf{v} \in \mathbb{R}^{3\times F}$. Denote an edge as $\vec{\mathbf{e}}_{uv} \equiv \vec{\mathbf{r}}_u - \vec{\mathbf{r}}_v$. Operation $\mathcal{N}(v)$ returns a set of vertices with the distance to vertex v smaller than some specific finite cutoff.

A geometric graph neural network (G-GNN) processes information in sequential layers with index l: $l \in [1, D]$.

$$\mathbf{d}_u^{(l+1)} \Leftarrow \underline{\text{Update}}^{(l)}\left(\mathbf{d}_u^{(l)}, \sum_{v\in N_u} \underline{\text{Message}}^{(l)}(\mathbf{d}_u^{(l)}, \mathbf{d}_v^{(l)}, \mathbf{e}_{uv}^{(l)})\right),$$

where $\underline{\text{Update}}^{(l)}$ and $\underline{\text{Message}}^{(l)}$ are functions with trainable parameters. This section employs the notation from [52] and marks trainable functions and trainable parameters with an underline.

DeepDFT is an architecture that works on a graph with two partitions: atoms $\mathcal{V}_a$ and probe points $\mathcal{V}_p$.

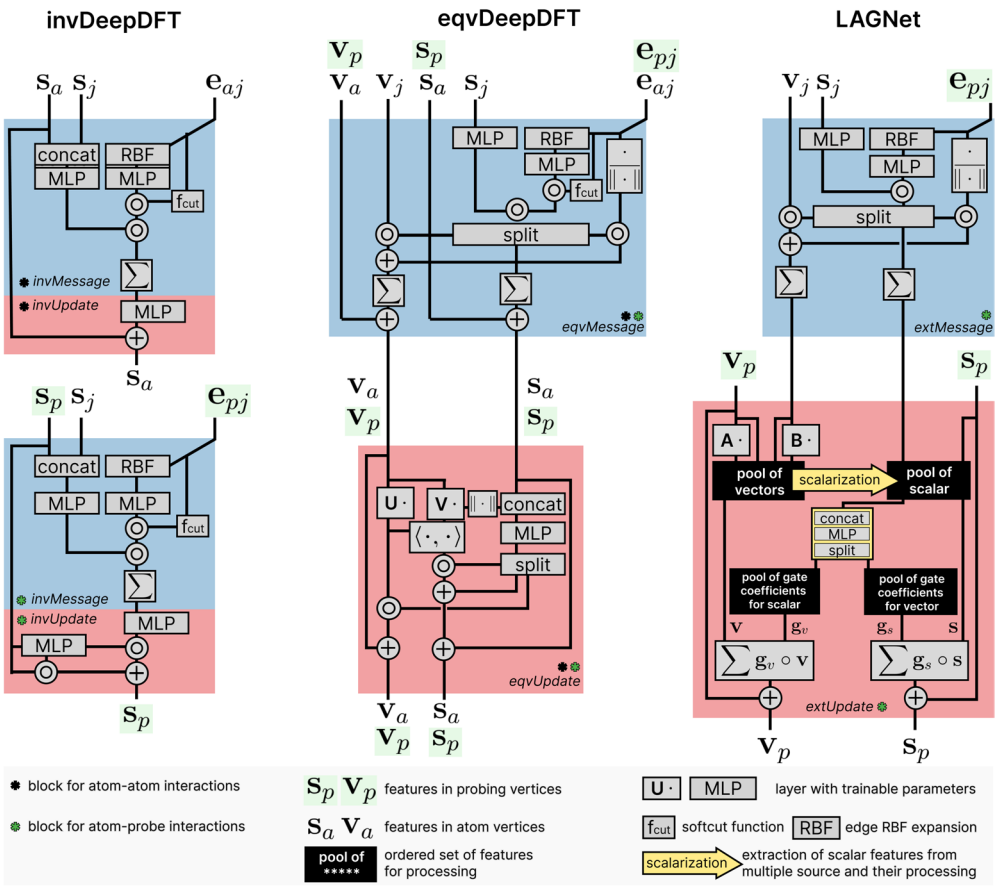


Fig. 4 Diagram showing the main modules in each neural network architecture. Blue blocks represent the message-passing part of the GNN, and red blocks represent the update part. invDeepDFT uses different blocks for atom-to-atom and atom-to-probe interactions. eqvDeepDFT uses the same blocks for both types of interactions. LAGNet further adapts the eqvDeepDFT blocks to enable an advanced message processing from atoms to probe points. The invDeepDFT and eqvDeepDFT diagrams are based on the original DeepDFT paper and source code, while LAGNet is introduced in this work

These partitions do not intersect and include all vertices: $\mathcal{V}_a \cup \mathcal{V}_p = \mathcal{V}, \mathcal{V}_a \cap \mathcal{V}_p = \varnothing$. Atoms send messages to each other, and then information from atoms is propagated to probe points. Messages between atoms propagate in both directions, but messages from atoms propagate to probe points in one direction. This framework also uses the embedding layer, and readout multilayer perceptron (MLP) to compute the electron density. Operations $\mathcal{N}_a(v)$ and $\mathcal{N}_p(v)$ provide all atoms around a specific atom or probe points, respectively.

$$\mathbf{d}_a = (\vec{\mathbf{r}}, \mathbf{s}, \mathbf{v})_a, \quad \mathbf{s}_a^{(0)} = \underline{\text{Embedding}}(z_i), \quad \mathbf{v}_a^{(0)} = \mathbf{0}, \qquad a \in \mathcal{V}_a,$$

$$\mathbf{d}_p = (\vec{\mathbf{r}}, \mathbf{s}, \mathbf{v})_p, \quad \mathbf{s}_p^{(0)} = 0, \quad \mathbf{v}_p^{(0)} = \mathbf{0}, \qquad p \in \mathcal{V}_p,$$

$$\mathbf{d}_a^{(l+1)} \Leftarrow \underline{\text{Update}}_A^{(l)}\left(\mathbf{d}_a^{(l)}, \sum_{v \in N_a} \underline{\text{Message}}_A^{(l)}(\mathbf{d}_a^{(l)}, \mathbf{d}_v^{(l)}, \mathbf{e}_{av}^{(l)})\right),$$

$$\mathbf{d}_p^{(l+1)} \Leftarrow \underline{\text{Update}}_P^{(l)}\left(\mathbf{d}_p^{(l)}, \sum_{v \in N_p} \underline{\text{Message}}_P^{(l)}(\mathbf{d}_p^{(l)}, \mathbf{d}_v^{(l)}, \mathbf{e}_{pv}^{(l)})\right),$$

$$\rho(\vec{\mathbf{r}}) \Leftarrow \underline{\text{Readout}}(\mathbf{d}_p^{(D)})$$

In order to reduce over-complicated notation, we omit the upper index of the layers ($\square^{(0)}$, $\square^{(l)}$, $\square^{(D)}$) and keep only $\Leftarrow$ between the left and right parts of the equations.

Invariant DeepDFT (invDeepDFT) is based on the SchNet [53] neural network (see 4 for visual explanation). We directly obtain the following equations from the original DeepDFT implementation. The Message and Update functions of invDeepDFT are combined as follows:

$$\mathbf{s}_a \Leftarrow \mathbf{s}_a + \underline{\text{MLP}}\Big(\sum_{j \in \mathcal{N}(a) \setminus a} \mathbf{g}_a \circ \underline{\text{MLP}}\big([\mathbf{s}_a, \mathbf{s}_j]\big)\Big)$$

$$\mathbf{g}_a = \underline{\text{MLP}}(\text{RBF}(\mathbf{e}_{aj})) \circ \text{f}_{\text{cut}}(\mathbf{e}_{aj})$$

where $[\cdot,\cdot]$ is concatenation and $\circ$ is a pointwise product. The same functions for atom-probes iteration are slightly changed:

$$\mathbf{s}_p \Leftarrow \mathbf{g}_u \circ \mathbf{s}_p + (1 - \mathbf{g}_u) \circ \underline{\text{MLP}}\Big(\sum_{j \in \mathcal{N}(p)} \mathbf{g}_p \circ \underline{\text{MLP}}\big([\mathbf{s}_p, \mathbf{s}_j]\big)\Big)$$

$$\mathbf{g}_u = \underline{\text{MLP}}(\mathbf{s}_p)$$

$$\mathbf{g}_p = \underline{\text{MLP}}(\text{RBF}(\mathbf{e}_{pj})) \circ \text{f}_{\text{cut}}(\mathbf{e}_{pj})$$

The invDeepDFT model is based on SchNet but includes notable modifications. The message function concatenates the sender and receiver scalar values. The update function includes the gating expressions that differ for the atom-atom interaction and the atom-probe interaction. invDeepDFT does not use vector features and edge directions.

Equivariant DeepDFT (eqvDeepDFT) uses more complex message and update functions (see Fig. 4 for a visual explanation). The eqvMessage function takes information from the edge direction and utilizes scalar and vector features. The message function of eqvDeepDFT for atom-atom interaction is defined as follows:

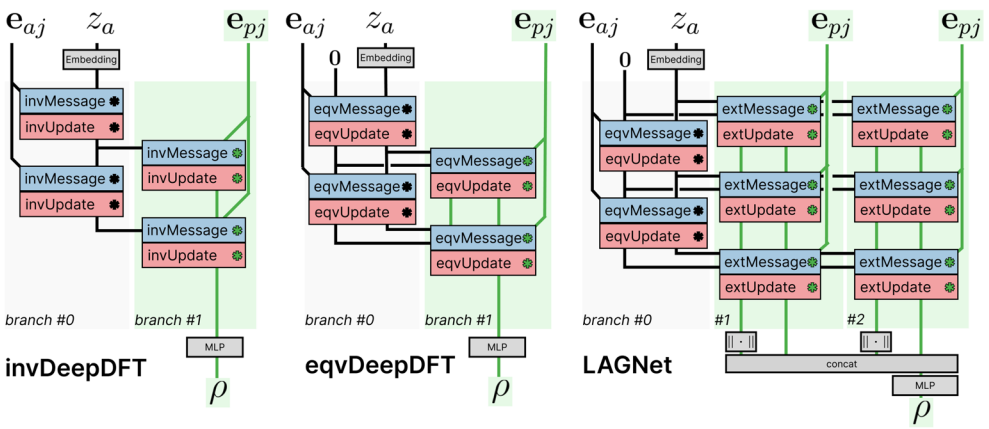


Fig. 5 Overall block layout for each architecture. Branch #0 exchanges messages between atoms and updates atomic states (gray blocks and lines). Branches #1 and #2 send messages from atoms to probe points and update probe states (green blocks and lines). The connections show that invDeepDFT passes only scalar features, while eqvDeepDFT and LAGNet pass both scalar and vector features. LAGNet uses two separate branches (#1 and #2) specifically for atom-to-probe message passing

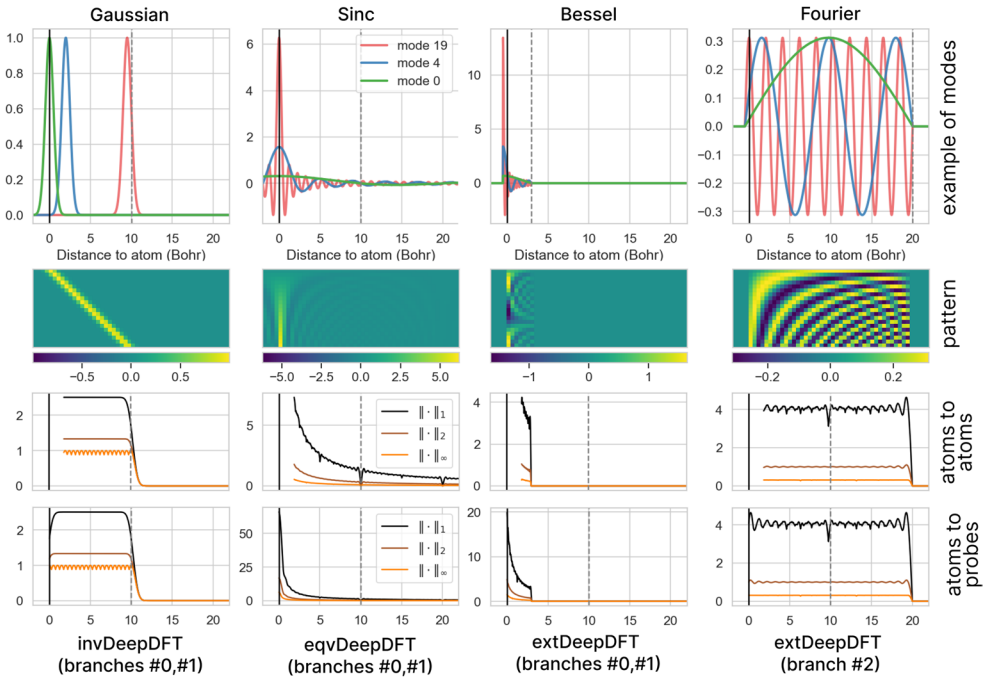


Fig. 6 Radial basis function (RBF) expansions used in each model. Each column corresponds to one RBF function. From top to bottom, the rows show: the function name; examples of three expansion modes; a pattern image displaying all 20 modes; the l_1, l_2, and l_∞ norms of the RBF expansions computed on real molecules for atom-to-atom and atom-to-probe messages (vertical lines mark the zero and cutoff distances from the atom); and the model name with the branch where the expansion is applied

$$[\mathbf{g}_s, \mathbf{g}_{v1}, \mathbf{g}_{v2}] = \underline{\mathrm{MLP}}(\mathbf{s}_a) \circ \underline{\mathrm{MLP}}(\mathrm{RBF}(\mathbf{e}_{aj})) \circ \mathrm{f_{cut}}(\mathbf{e}_{aj})$$

$$\mathbf{s}_a \Leftarrow \mathbf{s}_a + \sum_{j \in \mathcal{N}(a) \setminus a} \mathbf{g}_s$$

$$\mathbf{v}_a \Leftarrow \mathbf{v}_a + \sum_{j \in \mathcal{N}(a) \setminus a} \left(\mathbf{g}_{v1} \circ \mathbf{v}_a + \mathbf{g}_{v2} \circ \frac{\vec{\mathbf{e}}_{aj}}{\|\vec{\mathbf{e}}_{aj}\|} \right)$$

The expressions for the atom-atom and atom-probe interactions are the same in eqvDeepDFT. The only difference is the function $\mathcal{N}(p)$, which returns in the neighborhood of an probe point instead of atom neighborhood ($\mathcal{N}(a) \setminus a$).

$$[\mathbf{g}_s, \mathbf{g}_{v1}, \mathbf{g}_{v2}] = \underline{\mathrm{MLP}}(\mathbf{s}_p) \circ \underline{\mathrm{MLP}}(\mathrm{RBF}(\mathbf{e}_{pj})) \circ \mathrm{f_{cut}}(\mathbf{e}_{pj})$$

$$\mathbf{s}_p \Leftarrow \mathbf{s}_p + \sum_{j \in \mathcal{N}(p)} \mathbf{g}_s$$

$$\mathbf{v}_p \Leftarrow \mathbf{v}_p + \sum_{j \in \mathcal{N}(p)} \left(\mathbf{g}_{v1} \circ \mathbf{v}_p + \mathbf{g}_{v2} \circ \frac{\vec{\mathbf{e}}_{pj}}{\|\vec{\mathbf{e}}_{pj}\|} \right)$$

The update layers are also the same for atoms and probes. The matrix multiplications $\underline{U}\mathbf{v}$ and $\underline{V}\mathbf{v}$ are applied to a list of vectors $\vec{\mathbf{v}} \in \mathbb{R}^{3 \times F}$ along the feature dimension, where $\underline{U}$, $\underline{V}$ are training weights. These equivariant operations are named *channel-mixing* operations as in [52].

The update expressions for eqvDeepDFT (indexes a and p are omitted for simplicity):

$$[\mathbf{g}_{vv}, \mathbf{g}_{sv}, \mathbf{g}_{ss}] = \underline{\mathrm{MLP}}(\left[\mathbf{s}, \|\underline{V}\mathbf{v}\|_2\right])$$

$$\mathbf{s} \Leftarrow \mathbf{s} + \mathbf{g}_{ss} + \mathbf{g}_{sv} \circ \langle \underline{U}\mathbf{v}, \underline{V}\mathbf{v} \rangle$$

$$\mathbf{v} \Leftarrow \mathbf{v} + \mathbf{g}_{vv} \circ \underline{U}\mathbf{v}$$

Atom-atom interactions and atom-probe interactions form two branches of the model, which we denote as branch #0 and branch #1 (see Fig. 5). invDeepDFT and eqvDeepDFT use an embedding layer to encode atoms. The MLP readout combines the outputs of the branches and computes the electron density at the probe point.

LAGNet

After the analysis of the performance of eqvDeepDFT on the ∇^2DFT dataset, we introduce several modifications that improve the model. We name our model LAGNet.

The first modification is related to the expansion of the RBF in the message block. According to [52], an RBF expansion enriches the edge representation and maps $\vec{\mathbf{e}} \in \mathbb{R}^3$ to a multidimensional cube ($\mathrm{RBF}(\vec{\mathbf{e}}) \in [0, k]^N$). Figure 6 shows the expansions used in the models. invDeepDFT uses the Gaussian expansion: $\mathrm{RBF}(\vec{\mathbf{e}}_{ij}) = \underset{n \in [1, N+1]}{\mathrm{concat}} \exp(-((\|\vec{\mathbf{e}}_{ij}\|_2 - \mu)^2)/(2\sigma^2))$.

$\mu \in [0, \sigma]$, $\sigma = nC/N_{\text{exp}}$, where $N_{\text{exp}} = 20$ is the size of the expansion of the RBF, C is the cutoff value, and 'concat' denotes the concatenation of features together. The eqvDeepDFT model uses the Sinc expansion: $\text{RBF}(\vec{\mathbf{e}}_{ij}) = \underset{n\in[1,N+1]}{\text{concat}} \sin(\frac{\pi n}{C}\|\vec{\mathbf{e}}_{ij}\|_2)/\|\vec{\mathbf{e}}_{ij}\|_2$.

The distribution of the edge distances differs significantly for atom-atom and atom-probe interactions. The minimal distance between a pair of atoms in ∇^2 DFT is 1.7 Bohr (0.9 Å). Consequently, for the atom-atom interaction, the model does not encounter the distance between 0 and 0.9 Å in the cases of Sinc or Gaussian expansions. Unlike atoms, probe points may lie at arbitrary distances from the atomic center. Figure 6 shows the norms for the expansion vectors for a random subsample of the dataset. In this subsample, the maximum value of the l_1 norm of the Sinc expansion for atom-atom interactions is 7.31. However, the maximum value for atom-probe interactions is 66. This discrepancy may cause instabilities with neural network training.

In aim to solve the last problem we propose to use different RBF expansions. The available literature [54], frameworks [55, 56], and implementations propose numerous variants of RBF expansions with several modifications and hyperparameters in each. We performed a series of preliminary experiments and tested the functions of the scipy [51], the e3nn [56] and SchNetPack [57] frameworks with a shallow neural network (D=2, F=32). These libraries include the most common RBF expansions that are used in deep learning models and other applications. The Bessel function (similar to Sinc) causes better prediction for probe points which are close to atoms, and the Fourier expansion provides better results for points with distances larger than 8 Bohr. We use these two functions for LAGNet (see Fig. 6). In addition, to fix the issue of large amplitudes of RBF expansions in atom-probe interactions, we changed the distance value by 0.5 Bohr ($\|\vec{\mathbf{e}}_{ij}\|_2 + 0.5$). The final RBF functions are defined as follows:

$$\text{RBF}(\vec{\mathbf{e}}_{ij}) = \underset{n\in[1,N+1]}{\text{concat}} \frac{1}{\sqrt{c}x} \sin(n\frac{x}{c}),$$
$$x = \|\mathbf{e}_{ij}\|_2 + 0.5,\ c = C + 0.5$$
$$\text{RBF}(\vec{\mathbf{e}}_{ij}) = \underset{n\in[1,N+1]}{\text{concat}} \frac{\sin(\pi n x)}{\sqrt{0.25 + N/2}},$$
$$x = \frac{\|\vec{\mathbf{e}}_{ij}\|_2 + 0.5}{C + 0.5}$$

The second modification targets softcut functions ($f_{\text{cut}}(\cdot)$) in the Mesage block. InvDeepDFT uses $\text{f}_{\text{cut}}(\vec{\mathbf{e}}_{ij}) = 1 - \text{sigmoid}(5(\|\vec{\mathbf{e}}_{ij}\|_2 - (C - 1.5)))$, and eqvDeepDFT uses $\text{f}_{\text{cut}}(\vec{\mathbf{e}}_{ij}) = \frac{1}{2}(\cos(\pi\|\vec{\mathbf{e}}_{ij}\|_2/C) + 1)$, where C is a cut-off distance in Angstrom. Both invDeepDFT and eqvDeepDFT build upon the SchNet and PaiNN architectures, respectively. Softcut functions are introduced to stabilize the gradients of the models with respect to atom positions. This is necessary for predicting force fields and incorporating trained models into molecular dynamic simulations. However, the benefits of these functions in the electron density prediction tasks are not obvious. In order to reduce unnecessary computation, we do not use f_{cut} in LAGNet in the message blocks.

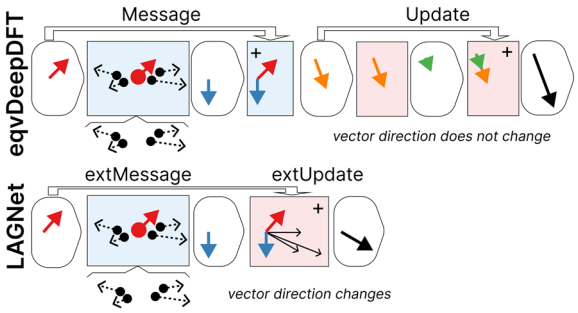


Fig. 7 This explains how we modified eqvDeepDFT's Message and Update blocks to create LAGNet. In eqvDeepDFT, the Update block only changes a vector's length, not its direction. LAGNet adds an extUpdate block that gathers vectors both before and after the Message block and combines them into a single output vector with any direction. We found that adding extUpdate blocks to branches #1 and #2 improves model performance. These branches send messages from atoms to probe points

The third modification targets the message-update pair of blocks. We introduce a novel, generalized update block that extends the PaiNN/DeepDFT framework by unifying scalar and vector operations. Rather than operating on a single list of input vectors, our update block accepts a *pool of vector* gathered from multiple sources. Each vector in the pool is first converted into multiple scalar features via *scalarization operations*. These pooled scalar features feed into a shared MLP, which produces two sets of gate coefficients: one for updating scalar features and one for updating vectors. The vector gates are applied element-wise to the original input vectors, yielding equivariant output vectors. This three-stage architecture-pooling, scalarization, and gated update-generalizes the standard PaiNN update block by allowing flexible aggregation of information from diverse vector inputs while preserving equivariance. This section provides intuition for the proposed extensions and presents all conversions in a formal manner.

eqvDeepDFT uses skip connections with the Update block: $\mathbf{s} \Leftarrow \mathbf{s} + f_s(\mathbf{s})$, $\vec{\mathbf{v}} \Leftarrow \vec{\mathbf{v}} + f_v(\vec{\mathbf{v}})$. Mixing of the channels $\underline{U}\mathbf{v}$ does not change the orientation of the input vector, and multiplication on the gate coefficient also does not change the direction. Thus, the entire update block

does not change the direction of the input vector (see Fig. 7 for visual explanation). This limits the expressiveness of an update block and is the target for an extension.

According to recent reviews, DeepDFT is a neural network that belongs to the class of Cartesian tensor G-GNN [52], and to the class of equivariant values network [58]. The major advantage of this network is its simplicity [52], and its performance [58]. Experiments with the prediction of energy and force fields show that Cartesian tensors G-GNN handle both tasks with good precision and even surpass some spherical tensor G-GNN with a greater number of trainable parameters [32]. For these reasons, we aim to extend the Update block with elements from a spherical tensor G-GNN, but keep a general class of networks as a Cartesian tensor G-GNN.

The major benefit of the spherical tensor G-GNN (irreducible representation neural network in terms of [58]) is its expressiveness and ability to represent the neighborhood of complex atoms as a unique state of a tensor. A crucial part of the spherical tensor G-GNN is a trainable tensor product operation. The tensor product is a function equivariant to the group action on two arguments: $\mathfrak{g}.\mathbf{h} = f(\mathfrak{g}.\mathbf{d}_1, \mathfrak{g}.\mathbf{d}_2)$. We design the extUpdate block to have the same properties without usage spherical harmonics of higher orders.

The extUpdate block is shown in Fig. 4. This block receives two inputs: $(\mathbf{s}_1, \mathbf{v}_1)$ from the previous update block and $(\mathbf{s}_2, \mathbf{v}_2)$ from the message block. The original update block (eqvUpdate) from eqvDeepDFT contains the channel mixing operation ($\underline{U}\mathbf{v}$). In extUpdate, channel mixing is applied to each input vector: $\underline{A}\mathbf{v}_1$, $\underline{B}\mathbf{v}_2$. Four vectors form a vector feature pool $\mathcal{U} = [\mathbf{v}_1, \mathbf{v}_2, \underline{A}\mathbf{v}_1, \underline{B}\mathbf{v}_2]$.

An MLP cannot work with vector features directly. A scalarization operation transforms pool of vector features into scalar values suitable as input to an MLP. The eqvUpdate block in eqvDeepDFT uses two scalarization operations: the l_2-norm $\|\cdot\|_2$ and the dot product $\langle\cdot,\cdot\rangle$. In the extUpdate, we extend this list of operations by the cross-product. Define a scalarization($\mathcal{U}$) operation that operates on a pool of vector features $\mathcal{U}$. This operation maps a set of vectors and vectors pairs to a set of scalar features: $\text{scalarization}(\mathcal{U}) = \left[\underset{\mathbf{a}\in\mathcal{U}}{\text{concat}}\|\mathbf{a}\|_2, \underset{\mathbf{a},\mathbf{b}\in\mathcal{U}}{\text{concat}}\langle\mathbf{a},\mathbf{b}\rangle, \underset{\mathbf{a},\mathbf{b}\in\mathcal{U}}{\text{concat}}\|\mathbf{a}\times\mathbf{b}\|_2\right]$.

The input scalar features and the extracted scalar features are then concatenated and passed to an MLP layer. The output layer produces a pool of gating coefficients for the vector features ($\mathcal{G}_s$) and for the scalar ones ($\mathcal{G}_v$). Each gate

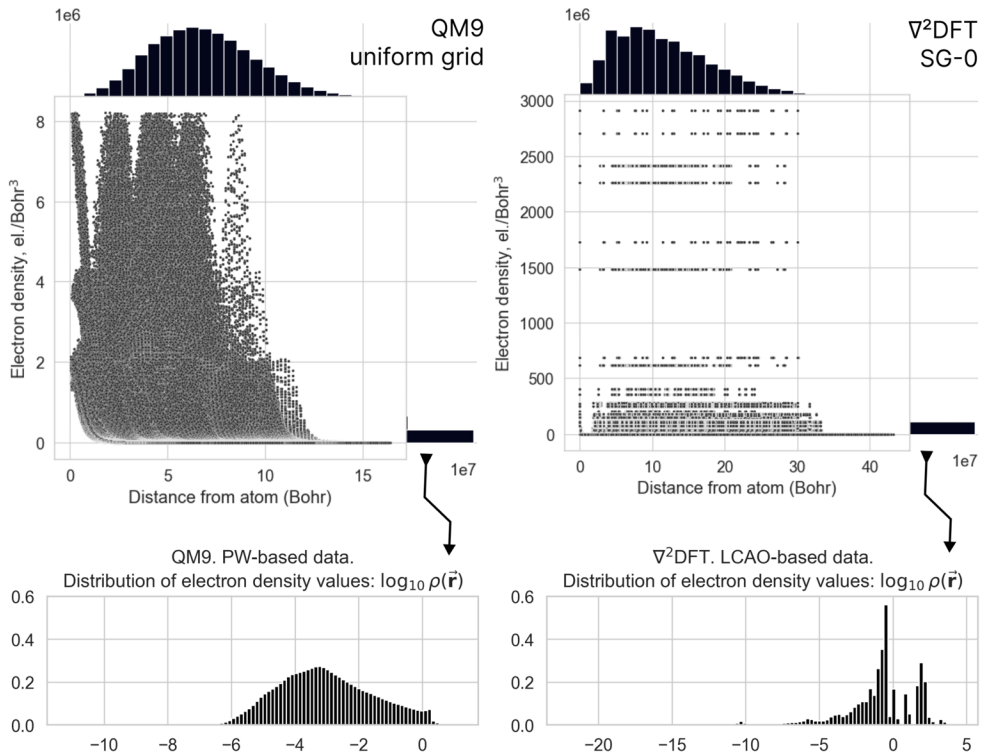


Fig. 8 These panels show the main differences between PW-based electron density on a uniform grid and LCAO-based electron density on the standard grid. The two grid layouts produce different distributions of distances between atoms and probe points. In addition, PW data use pseudopotentials, while LCAO data include full core orbitals with all electrons. This difference leads to distinct distributions of electron density amplitudes

coefficient is multiplied by a corresponding vector from the pool of vector features and a scalar from the pool of scalar features. These linear combinations form the output scalar and vector features. All together:

$$\begin{aligned}
\mathcal{U} &= [\mathbf{v}_1,\ \mathbf{v}_2,\ \underline{A}\mathbf{v}_1,\ \underline{B}\mathbf{v}_2] \\
&\text{scalarization}(\mathcal{U}) \\
&= \Big[\operatorname*{concat}_{\mathbf{a}\in\mathcal{U}}\|\mathbf{a}\|_2,\ \operatorname*{concat}_{\mathbf{a},\mathbf{b}\in\mathcal{U}}\langle\mathbf{a},\mathbf{b}\rangle,\ \operatorname*{concat}_{\mathbf{a},\mathbf{b}\in\mathcal{U}}\|\mathbf{a}\times\mathbf{b}\|_2\Big] \\
\mathcal{S} &= \big[\mathbf{s}_1,\ \mathbf{s}_2,\ \text{scalarization}(\mathcal{U})\big] \\
&[\mathcal{G}_s, \mathcal{G}_v] = \underline{\text{MLP}}\Big(\mathcal{S}\Big) \\
\mathbf{s}_1 &\Leftarrow \mathbf{s}_1 + \sum_{i\in[1,\ldots,|\mathcal{G}_s|]} \mathcal{G}_s[i]\circ\mathcal{S}_s[i] \\
\mathbf{v}_1 &\Leftarrow \mathbf{v}_1 + \sum_{i\in[1,\ldots,|\mathcal{G}_s|]} \mathcal{G}_v[i]\circ\mathcal{U}_v[i]
\end{aligned}$$

By $\mathcal{A}[i]$, we denote the i-th element of the list $\mathcal{A}$. The described approach is the generalized form of the gates that are used in invDeepDFT and eqvDeepDFT (see Fig. 4). The general definitions given above can be extended for other scalarization operations and additional input vectors. However, our specific implementation of scalarization does not use repeated results from associative binary operations and does not use two vector products: $\|A\mathbf{v}_1 \times \mathbf{v}_1\|, \|B\mathbf{v}_2 \times \mathbf{v}_2\|$. Thus, the input features in concatenation and MLP do not repeat.

The fourth modification targets the global structure of the neural network. Atom-atom interactions are calculated inside the list of sequential blocks that we denote as branch #0. This branch is based on the SchNet or PaiNN models. For invDeepDFT and eqvDeepDFT, the atom-probe interactions are calculated within the second list of blocks, which we denote as a branch #1. To use two different RBF expansions, we add an additional branch #2 to LAGNet (see Fig. 5).

Both invDeepDFT and eqvDeepDFT do not send the initial atom embeddings to the probe point, so information is exchanged between the atoms at least one time before passing to the nodes. However, in earlier studies [19], the electron density is directly predicted by atom embeddings. In order to use the initial atom embedding, we add an additional message-passing layer. Thus, the initialization of the scalar and vector features 2 was replaced by the additional message and update layers. We also add additional scalarization of the vector features and pass the norm of the vector values into the readout function (see Fig. 5).

Neural network parametrization and implementation. In our study, all models (invDeepDFT, eqvDeepDFT, LAGNet) are parametrized by the pair of hyperparameters (F, D): F is the number of scalar and vector features, and D is the depth of G-GNN. We implement all three models in the independent codebase. In G-GNNs, a graph is usually represented by a list of its edges. This approach is beneficial for large sparse graphs, but not in the case of electron density prediction. A large distance between the atom and the farthest probe point enforces setting of a big cutoff value. Therefore, most atom-atom and atom-probe pairs are included in the graph. That makes geometric graph similar to a dense graph with a small percentage of omitted edges. For this reason, we implement models with an adjacency matrix and dense tensors (usual PyTorch tensors). Edges outside the cutoff distance are implemented via zero mask in the message aggregation inside the message block. Part of the source code is available in the supplementary material. The full project code will be available soon as an open source solution under the MIT open license.

Here are examples of real model performance during training and inference. LAGNet (D=4, F=128) fully converges in 26 h when trained on 2,809 conformations using a single NVIDIA A100 GPU, and in 170 h when trained on 12,145 conformations. For inference, evaluating the integral with SG-0 requires 140 ms for a molecule with 104 atoms (382 electrons). Generating a cube file on a dense uniform grid (margin = 2 Å, spacing = 0.1 Å) takes 6.5 s. This is significantly faster than a classical DFT calculation. Figure A3 in the Appendix includes additional measurements of computational performance.

Metrics

Denote $\mathcal{M} = \{(\vec{\mathbf{r}}_i, w_i) \mid i \in [0\ldots|\mathcal{M}|]\}$ as an integration grid with $|\mathcal{M}|$ pairs of point coordinates $\vec{\mathbf{r}}$ and point weights w. The key metric to measure the model performance is the normalized mean absolute error (NMAE). We define exact NMAE via integrals and approximate this integral with the standard grid using weight coefficients. Following the approach of [26, 27], we define the average and maximal NMAE for the dataset $\mathcal{D}$. We report the standard deviation, median, and minimal value of NMAE as well. In this study, the mean squared error (MSE), the mean absolute error (MAE), and the mean absolute percentage error (MAPE) are defined in the standard way, without integration over the grid.

$$\begin{aligned}
\text{NMAE}_{\mathcal{D}}(\rho,\hat{\rho}) &= \frac{\int|\rho(\vec{\mathbf{r}})-\hat{\rho}(\vec{\mathbf{r}})|d\vec{\mathbf{r}}}{\int|\rho(\vec{\mathbf{r}})|d\vec{\mathbf{r}}} \\
&\approx \frac{\sum_{i\in\mathcal{G}} w_i|\rho(\vec{\mathbf{r}}_i)-\hat{\rho}(\vec{\mathbf{r}}_i)|}{\sum_{i\in\mathcal{G}} w_i|\rho(\vec{\mathbf{r}})_i|}, \\
\text{avg-NMAE}_{\mathcal{D}}(\rho,\hat{\rho}) &= \frac{1}{|\mathcal{B}|}\sum_{\rho\in\mathcal{D}}\text{NMAE}_{\mathcal{D}}(\rho,\hat{\rho}), \\
\text{max-NMAE}_{\mathcal{D}}(\rho,\hat{\rho}) &= \max_{\rho\in\mathcal{D}}\text{NMAE}_{\mathcal{D}}(\rho,\hat{\rho}),
\end{aligned}$$

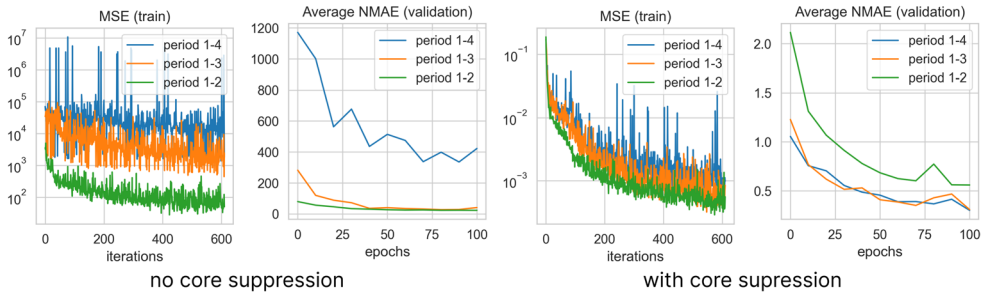


Fig. 9 This panel shows why core suppression is needed to successfully train neural networks on LCAO-based data containing third- and fourth-period atoms. The plots display the MSE loss on the training set and the average NMAE on the validation set. Including core orbitals in the data produces very large target values, which slows down and destabilizes neural network convergence. The training metrics are shown for eqvDeepDFT. Results for invDeepDFT and LAGNet are similar, but omitted for brevity.

Results

General observations

In Fig. 8, we show the major differences between the QM9 dataset from [26, 27] and the ∇^2DFT dataset. The QM9 dataset includes only atoms from the first and second periods (H, C, N, O, F), while ∇^2DFT also includes atoms from the third (S, Cl) and fourth (Br) periods. For the original DeepDFT study, the dataset was generated with the VASP software, which utilizes the PW basis and pseudopotentials. The electron density was stored on the uniform grid with a margin of 2 Å from the boundary atoms, and the grid step was close to 0.1 Å. In this study, we train the neural networks on the electron density, stored on the SG of the first level (pyscf/Turbomole implementation, no pruning, default parameters).

The crucial difference of ∇^2DFT from QM9(VASP) is the presence of the core orbitals and the significant impact of the symmetric atom-centered electron density on the data. To highlight the difference in the order of values, we use the logarithm with base 10. As shown in Fig. 8, the logarithm of the electron density for the QM9(VASP) dataset is close to the normal distribution. At the same time, the logarithm of electron density values for ∇^2DFT remains strongly skewed.

Figure 8 also shows the difference in maximal electron density values for PW-based and LCAO-based data. To measure this difference, we use polynomial regression. We find the closest probe point to each atom and build a regression model between the atom number and the electron density amplitude at the probe point. For ∇^2DFT data, the best model is $\rho_{\text{near\_atom}}(z_i) = 0.6344z^3$, $p < 1e^{-6}$, $R^2 = 0.998$. For QM9(VASP) data, the best model is $\rho_{\text{near\_atom}}(z_i) = 0.248z$, $p < 1e^{-6}$, $R^2 = 0.44$. Thus, for LCAO-based data, the electron density near the atom is proportional to the cube of atom number. For PW-based data, the electron density near each atom exhibits large local variation but never exceeds an absolute value of 8. This does not imply a fundamental physical law, but is related to a specific subset of chemical space in the data and computation methods.

The type of grid affects the distribution of the atom-probe distances. The uniform grid arrangements depend on the size of the molecule and the minimal bounding box around it. The SG-0 arrangement depends on the atom positions. The SG always includes probe points at a specific distance from each atom. These features cause specific bars in the histogram related to core orbitals (see Fig. 8, bottom row), and the linear arrangements of points in scatter plots (see Fig. 8, top row). That allows us to conclude that the LCAO-based ground truth data differ significantly from the PW-based one.

The core orbitals

The presence of core orbitals is a prominent feature of the LCAO. It introduces a notable impact on the amplitude of the symmetric part of the atom-centered electron density. In this study, we introduce an approach to reduce the core orbital amplitudes and suppress the atom-centered symmetric part of the electron density.

To estimate the impact of high amplitudes on models, we trained eqvDeepDFT (D=5, F=128) models with and without the core supression model. Figure 9 shows a major problem with the target data. If core suppression is not applied, the initial loss value is very high and training does not converge to MSE=0.1 and NMAE(%)=200% for more than 10,000 epochs. However, these values of the metrics are reached at the 2 epoch for the same models after the application of the core suppression model.

In addition, we studied the influence of the period of elements on the training. For this, we created three splits of the dataset: with elements of 1–2 period (H, C, N, O, F), 1–3 period (the same and S, Cl) and 1–4 period (the same and S, Cl, Br). Figure 9 shows the training dynamics for these splits. The loss value is reduced, when the models are trained only on the elements of 1–2 period. However, without core suppression, the presence of Cl

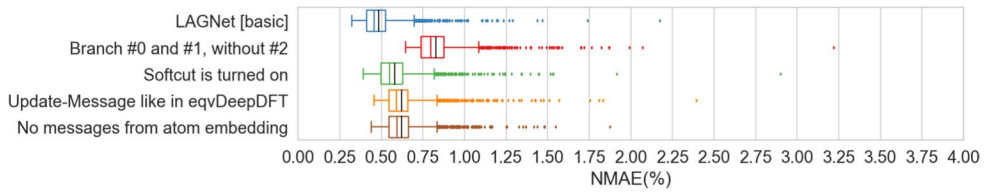


Fig. 10 Ablation study for LAGNet. LAGNet (D=2, F=32) was trained on single conformation per molecule. NMAE(%) was computed on the conformation test split

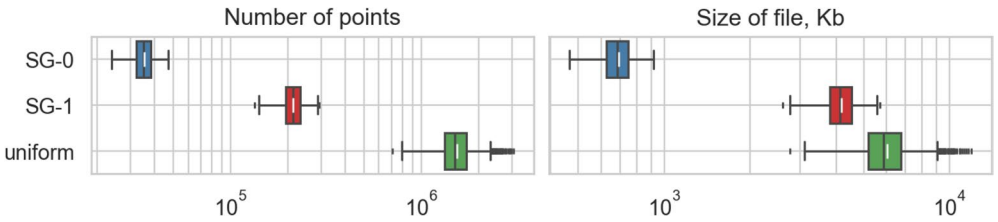


Fig. 11 Number of points and size of data for SG-0, SG-1, and the uniform grid. Values are computed for the scaffold test split. The vertical white marks show means. X-axes are logarithmic.

and S in the dataset negatively affects the convergence of neural network training. The presence of Br leads to a plateau with enormously high values of the loss function on the validation split. The application of the core suppression model allows us to effectively avoid any obstacle related to the element period, the element number, and the amplitude of the core orbitals (see the right of the Fig. 9).

Ablation study of LAGNet

To achieve a better NMAE in ∇^2 DFT, we introduce the LAGNet model. The model is derived from eqvDeepDFT with four modifications. To verify the role of the proposed changes, we used the partial ablation of the LAGNet model (D = 2, F = 32). The ablated models were trained on a single conformation per molecule. The neural network tests were performed on the scaffold test split. These experiments show the major differences in the NMAE metric.

The ablation includes several changes which simplify LAGNet. Each simplification disables a single modification from the Methods section 2.4. The first ablation uses only the branch #0 and #1. The branch #1 uses the Bessel RBF expansion with a 20 Bohr cutoff. The second ablation experiment is to use of f_{cut} function in the Update block like in the original eqvDeepDFT model. This function is used in the message block as in the original eqvDeepDFT. The third ablation uses the same message and update operations in the #1 and #2 branches of LAGNet as in the eqvDeepDFT. In the fourth ablation, the embedding message is not sent to the #1 and #2 branches. In the last case, the messages are exchanged at least once between atoms before being sent to the probe points. The results of the ablations are presented in Fig. 10 and Table A2 (in the Appendix). The absence of each modification from the proposed model reduces performance.

Advantages of SG-0 for training

The main reason to use SG in LCAO is to reduce the computation time while maintaining good precision for the results (see 2.2). Our experiments show that SG also improves the solution of the electron density prediction task. The representation of the electron density with a standard grid requires a sufficiently smaller number of points than in the case of a uniform grid. For a moderate-size drug-like molecule from ∇^2DFT, a uniform grid with spacing 0.1 Å and margin 2 Å from the boundary atoms includes about 43 times more points than the level zero standard grid (see Fig. 11). For a molecule from the scaffold test split, the number of probe points is on average 1,543,929±308,857 for the uniform grid, and 35,346±4,093 for the SG-0.

The benefits for memory usage are slightly smaller, but also significant. The standard grid requires five floating-point values to store one data point: 3 coordinates ($\vec{\mathbf{r}} = (x, y, z)$), weight ($w$), and the electron density value ($\rho(\vec{\mathbf{r}})$). Any points in the orthogonal uniform grid are defined by its index in the array ($i, j, k \in \mathbb{N}$), the position of the origin point of the grid ($(x_o, y_o, z_o) \in \mathbb{R}^3$), and the size of the bounding box ($(S_x, S_y, S_z) \in \mathbb{R}^3$). The positions of the grid points are recalculated from the index and the size of the bounding box as $(x, y, z) = (\frac{S_x}{N_x} i, \frac{S_y}{N_y} j, \frac{S_z}{N_z} k)$. Therefore, the representation of the electron density on

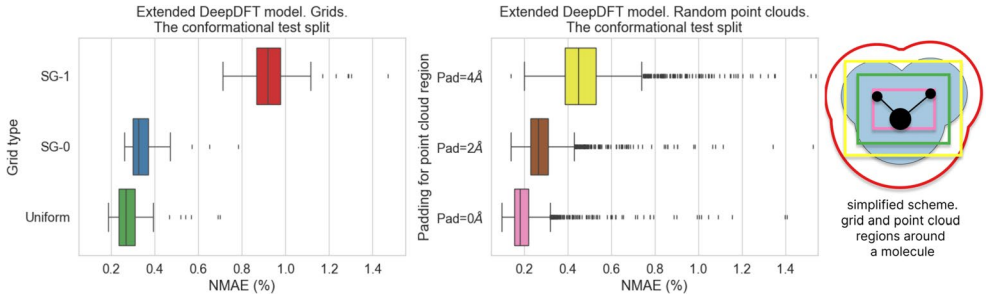


Fig. 12 This figure presents the performance of the best-trained model (LAGNet, D=4, F=128), which was trained on the SG-1 grid using multiple conformations per molecule. The left panel compares accuracy (NMAE%) on three grid types: SG-0, SG-1, and a uniform grid. The center panel shows performance on randomly sampled point clouds inside each molecule's bounding box, with increasing padding distances from the outermost atoms. The right panel is a schematic illustrating the sizes of these padded regions and their overlap. Boxplot colors match the corresponding regions in the schematic. Model error increases as the evaluation region grows (larger padding). A LAGNet model trained on SG-0 performs well on the uniform grid but shows reduced accuracy when evaluated on SG-1

the uniform grid requires one floating-point value for each point and six additional values that define the bounding-box geometry. We measure the required storage capacity for molecules from the scaffold test split of the ∇^2DFT dataset with the *float32* datatype for the electron density representation. The representation with SG-0 requires about 8 times less space (691±79 Kb, ≈0.67 Mb) than the representation with the uniform grid (6031±1206 Kb, ≈5.8 Mb). Figure 11 shows the distribution of sample sizes in the scaffold test split.

Model evaluation with other grids

SG-0 allows numerical methods to compute integrals over space with good precision. However, many downstream tasks require a uniform grid. For example, calculations of the Bader partial charges and visualization of the molecular electrostatic field. In this study, we check the performance of LAGNet (D = 4, F = 128) on the test dataset. We also expect that NMAE is sensitive to the integration region. Regions with a larger padding from the molecule may provide a larger (worse) NMAE. To verify this hypothesis and avoid unexpected effects of the grid generation, we sampled 20,000 random points from the various bounding boxes with padding of 0 Bohr (0 Å), 3.78 Bohr (2 Å), 7.55 Bohr (4 Å) from boundary atoms.

Our experiments confirm a good performance of the electron density prediction on the uniform grid if the model was trained on the SG-0. The maximum distance between the atom and the probe point in the uniform grid with a margin of 2 Å is 3.4641 Å. However, in the case of SG-0 this value is 5 Å. Therefore, the region of space covered by the uniform grid is contained in the region of space covered by SG-0. As shown in Fig. 12, NMAE(%) for the prediction of electron density in the uniform grid is higher than in SG-0. However, the error increases for prediction on the SG-1 and higher grid levels (see Fig. 12).

We calculated NMAE at random points inside the bounding box around the molecules. Figure 12 shows that the best NMAE is obtained for the zero padding bounding box. The increase in padding reduces the NMAE and the performance of the evaluated model. This result explains the behavior of neural network prediction on different grids. If the test grid is contained inside the training grid, then the NMAE metric is low. NMAE metric is high for test cases where evaluated region is larger than region during training. This also means that the neural network does not generalize for points outside the region of the training grid.

NMAE is an integral-based metric with normalization of values. It is sensitive to the noise contained in the outputs of the neural network and accumulates errors for a large region of integration. The electron density values for the probe points, which are far from the molecule's atoms, are small but nonzero. The region of integration between 5Å and 15Å has a large volume, causing a negative impact on the values of NMAE. Thus, NMAE for the region with a large padding of a molecule is higher compared to the results for regions with tighter restrictions.

Generalization to new conformations and molecules

The most important property of any neural network in quantum chemistry is its ability to generalize to novel conformations and molecules that the neural network did not observe during training. To test the generalization abilities, we evaluate invDeepDFT (D=6, F=128), eqvDeepDFT (D=6, F=128), and LAGNet (D=4, F=128) models on the test splits. The models were trained ether with single or multiple conformations per molecule on the tiny train split of ∇^2DFT. The ∇^2DFT dataset includes an advanced test split system to analyze the ability of neural networks to generalize. It contains conformation, structure, and scaffold test splits. The first split consists

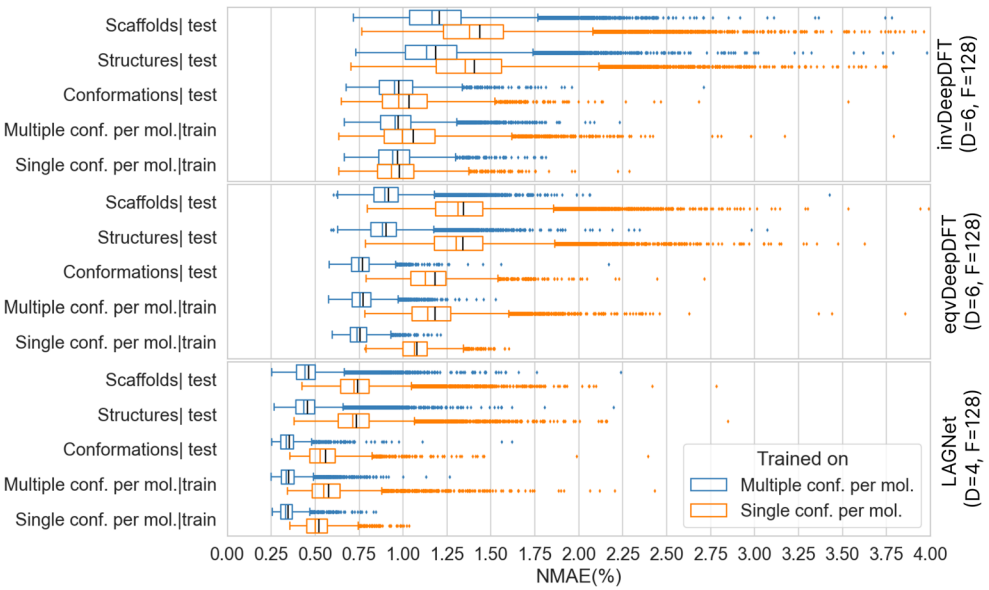


Fig. 13 Generalization performance on new conformations and unseen molecules. Results are shown for invDeepDFT (D=6, F=128), eqvDeepDFT (D=6, F=128), and LAGNet (D=4, F=128). Model accuracy is measured by NMAE(%). Two training splits were used: one with a single conformation per molecule and one with multiple conformations per molecule. Generalization is evaluated on three progressively harder test splits: the conformation split holds out new conformations of molecules seen during training; the structure split holds out entirely new molecules; and the scaffold split holds out molecules with unseen scaffolds

of the novel conformations, the second contains novel molecules with observed structural elements, and the last contains novel scaffolds.

Figure 13 shows the performance of invDeepDFT, eqvDeepDFT, and LAGNet. The distributions of the values of the NMAE metric are similar for training splits and the conformation test split. The distributions of the NMAE metric for the scaffold test split and the structure test split are also close to each other. This means that the neural network generalizes to unobserved molecular conformations and experiences issues with generalization to unobserved molecules. However, unobserved scaffolds do not cause additional degradation of the performance metric. These results highlight the difference between the generalization of neural networks that train on electron density prediction tasks and the other tasks. A recent study with the ∇^2 DFT dataset [32] reports that almost any model for energy and force field prediction performs worse for the scaffold test split than for the structure test split.

Figure 13 also presents a comparison of models that were trained in a single conformation per molecule, and multiple conformations per molecule. The performance of the model on the scaffold and the structures test splits differs notably. Any model that was trained in the multiple conformations scenario performs better in terms of average, median, and maximal NMAE (see Table A1). The benefits of training in multiple conformations are especially notable for max-NMAE. Across test splits, these metrics were reduced by 2–10 times for invDeepDFT, 2–20 times for eqvDeepDFT, and 1.5 times for LAGNet, compared to the single-conformation scenario. In general, training with multiple conformations per molecule reduces the heavy right tail in NMAE distributions.

In Fig. 13, there is a gap between the lowest NMAE value and zero (0.25%). The same phenomenon can be observed in the original DeepDFT paper [27], where the gap of and 0.1 % is shown. The study [9] reports a lower bound of point-wise prediction quality 10^{-4}el./ Å^3. In our experiments, the NMAE gap is lowered with the increase of the number of model parameters and with the usage of multiple conformations per molecule.

Cross-dataset model evaluation

To assess LAGNet ability to generalize beyond its training distribution, we evaluated the best-performing checkpoint (LAGNet, F=128, D=4), which was trained exclusively on the nabla2DFT dataset, on two external benchmarks: Hutchinson's dataset [48] and QMugs [33]. Trained models are not applicable to full datasets, because Hutchinson's dataset includes charged molecules, and QMugs includes P and I. We filter such molecules and evaluate LAGNet on obtained subsets aiming to check LAGNet generalization on bigger molecules, molecules from various sources, and other approaches to the conformation space generation.

Hutchinson dataset comprises three subcollections: Astex Diverse Set, OmegaPDB, and OmegaCSD. The first one includes experimental crystallography data. The second and third subcollections are derived from the Protein Data Bank. Molecular geometry for OmegaPDB and OmegaCSD are generated from SMILES using OpenBabel and subsequent DFT optimization. In addition, we add some molecules with bigger number of heavy atoms and more electrons from the QMugs dataset. The QMugs is derived from the ChEMBL database. We randomly selected 100 large, drug-like molecules (48–52 heavy atoms each) from QMugs with three conformations per molecule (300 total conformations).

From each subcollection, we retained only neutral, P- and I-free molecules. Reference electron densities for all selected conformations were computed at the wB97X-D/def2-SVP level of theory without any modification of molecule geometry presented in datasets. Table A3 in the Appendix contains basic statistics of evaluation subsets.

Figure 14 reports normalized mean absolute error (NMAE(%)) for each molecule. Performance of the model is slightly worse on Hutchison dataset and QMugs, than on test subsets of nablaDFT. That is related to the dataset shift, which is a widespread issue in generalization of deep learning models and observed in many general tasks in cross-dataset model evaluation [59]. However, for 92% of molecules, NMAE(%) is below 1%. We suppose that it shows a good generalization ability of our model in more diverse chemical and conformation spaces. We found no systematic correlation between NMAE and molecular size (either number of electrons or number of heavy atoms).

We perform additional experiments to detect possible correlation between properties of molecular system and NMAE(%). The results (Appendix, Figs. A1, A2) confirm that LAGNet's inference error is independent of common molecular descriptors: total atom count, heavy-atom count, heteroatom count, ring count, rotatable bond count, hydrogen bond donors/acceptors, and number of chiral centers.

Related works

Electron density is a function that maps a point from a 3D space to the estimated number of electrons at this point: $\rho(\vec{\mathbf{r}}) : \mathbb{R}^3 \to \mathbb{R}$. The total number of electrons in the space is calculated with an integral: $\int \rho(\vec{\mathbf{r}})\, d\vec{\mathbf{r}} = \frac{1}{2} \sum_{j \in [1, M(i)]} z_j = N_e$. Abstract electron density is not a differentiable function. It includes a cusp in atom positions and requires an infinite number of basis functions for decomposition. That makes the functional space of the abstract electron density not properly defined, dependable from theory, and generally intractable in computational machines with finite memory in finite time. All machine learning methods that predict the electron density are based on their definition from specific numerical methods. The linear combination of atomic orbitals uses the sum of symmetric Gaussians and spherical harmonics to form molecular orbitals $\psi_i(\vec{\mathbf{r}})$ from atomic orbitals $\phi_i(\vec{\mathbf{r}})$: $\psi(\vec{\mathbf{r}}) = C\phi(\vec{\mathbf{r}})$. The matrix C forms the electron density matrix $D = C^T C$ and defines the electron density at a point $\rho(\vec{\mathbf{r}}) = \psi(\vec{\mathbf{r}})^T D \psi(\vec{\mathbf{r}})$.

There are several major branches of studies for density prediction with machine learning techniques. The first branch employs Gaussian processes and Gaussian approximation. This branch of study is developed in the following works: [13–16, 16]. The second branch employs neural networks. Descriptors-based approaches are historically the first methods to predict electron density. This

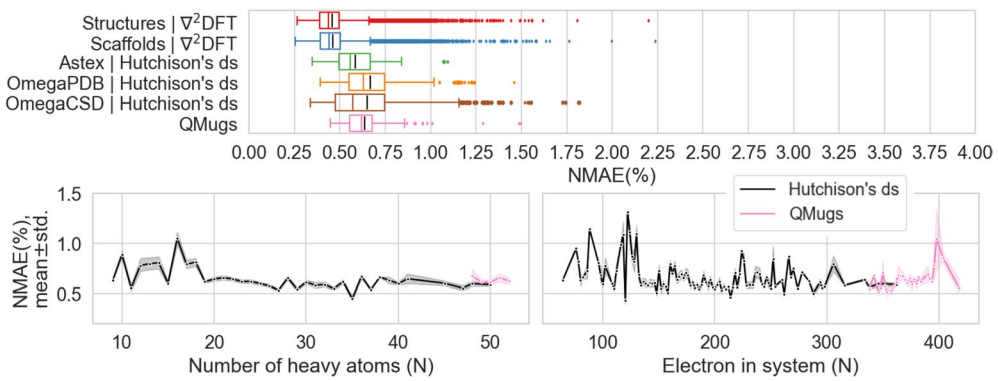


Fig. 14 Cross-dataset evaluation of LAGNet's electron-density predictions. Top row: Distribution of normalized mean absolute error (NMAE(%)) for three test sets - nabla2DFT (structures and scaffolds test splits)), the Hutchinson dataset (combined Astex, OmegaPDB, and OmegaCSD subsets), and a QMugs subset of 100 large molecules (three conformers each). Boxplots show median, interquartile range, and outliers. Black line marks mean of the distribution. Bottom row: Plot of NMAE(%) mean and standart deviation versus heavy-atom count and total electron count across all evaluated molecules. Error remains consistently low (≤1% for 92% of samples) and shows no systematic increase with molecular size

approach calculates molecular fingerprints and descriptors around probe points, then unifies features, and makes predictions of density value pointwise. These approaches were used in [17–20]. Despite the simplicity of these methods, they can reach 1–5% of NMAE for molecules that include H, C, N, and O. The second approach is to use CNN and U-Net architectures [21, 22]. The third approach uses the Fourier neural operator [23, 24]. The fourth approach employs the construction of basis functions around atoms and represents density as the sum of these bases [25] (AM2D), [42] (AM3D), [21] (InfGCN), [24] (SCDP), etc. Other approaches implicitly build such bases, using a one-directional message passing from atoms to probe points [26, 27]. The state-of-the-art approach usually combines two methods together. For example, [24] unifies the Fourier neural operator and explicit basis around atoms. An additional approach to electron density prediction is related to the full Hamiltonian (Fock matrix in self-consistency field iteration) and the prediction of overlap matrices [28, 29]. Using these two matrices and the basis-set, it is possible to calculate the electron density. Usually, Hamiltonian-focused studies do not report metrics on the electron density, but some do this: [29, 30].

Across all presented studies, we see significant biases in recent studies: datasets with small and simple molecules, PW-based computation of density with pseudopotentials, and use of the uniform grid. The most popular dataset for density prediction is QM9. This dataset includes only four heavy atoms (C, N, O, and F). There are more complex datasets for studies: ∇^2DFT [32], QMugs [33], SPICE [34], Frag20 [35], OrbNet Denali [36] datasets. These recent datasets include molecules with up to 100 heavy atoms and elements of 3–4 period (P, S, Cl, Se, Br, I, Ca, Mg). As we show in the results, atoms from the 3rd and 4th periods may cause issues during the training. An exhaustive review of the datasets is presented in [32]. ∇^2DFT, QH9, QMugs include information that allows a user to compute the electron density without recomputing the whole data. We chose ∇^2DFT for the study because it includes a rich split system and allows us to analyze the behavior of the models on out-of-distribution data in detail.

In modern quantum chemistry, there are two main branches of numerical methods and numerical software for the simulation of quantum chemical systems. One branch uses a Gaussian plane-wave basis (PW). This approach is implemented in VASP [37], Quantum Espresso [38], GPAW [39], and cp2k [40]. The other branch uses a local combination of atomic orbitals (LCAO). There are psi4 [43], pyscf [44], Turbomole [45], Gaussian [46], and other software packages based on LCAO. Most studies focus on PW-based data [21, 23, 24, 26, 27], only several studies use LCAO as ground truth data [25, 42]. The PW method requires a pseudopotential basis. Such an approach avoids issues with huge electron density values in core orbitals. The LCAO computation assumes core orbitals with full amplitudes. Due to the difference in numerical methods, model training for LCAO-based ground truth data is a more challenging task than training on PW-based data.

Uniform meshes are necessary for PW-basis computation and are mandatory for VASP software. VASP allows researchers to compute materials and molecules, making this software favorable in a wide range of possible downstream applications. Uniform grids are needed in studies that employ CNN (U-Net) architectures [21, 22] and Fourier neural operators [21, 23, 24]. Visualization of density isosurface is performed with algorithm of marching cubes in most packages. This method also requires a uniform grid. All of these together shift research attention to the uniform grid. This study is focused on LCAO-based data. LCAO computation employs the standard grid to compute spatial integrals with a smaller number of points than is necessary with the uniform grid to the same precision. For this reason, we analyze the benefits and drawbacks of using the standard grid with neural networks.

Discussion

This study has several practical implications for the prediction of the electron density with deep learning. The most recently proposed approaches for the prediction of electron density have been evaluated on small molecules such as those from the QM9 database. These molecules are limited to nine heavy atoms and do not cover the space of drug-like substances. The diversity of atoms in QM9 is limited by the second period. There are several datasets with bigger molecules (∇^2DFT, QMugs, SPICE, Frag20, OrbNet Denali). The biggest challenge to train on larger molecules and drug-like substances is the size of the molecules. Big molecules require significantly larger number of probing points for representation of electron density. In this study, we propose using the standard grid instead of the uniform grid for a discrete representation of the electron density. This allows the user to reduce the storage space of the training data and to reduce data transfer overhead.

Most studies on electron density prediction use data generated by the PW method as ground truth data. PW do not consider all electrons and replace the core electron with pseudopotentials. However, about half of all available open-source and proprietary software uses the LCAO method (Gaussian, ORCA, Turbomole, psi4, pyscf, etc.). LCAO method usually uses an all-electron computation setup that leads to a huge amplitude of core orbitals in the generated electron density. In this study, we show that normalization of core orbitals is necessary for effective training of neural network. We propose a

core orbital suppression model that is a specific approach for data normalization.

Current G-GNN for 3D molecular data processing can be roughly divided into Cartesian tensor G-GNNs and spherical tensor G-GNNs [52]. Cartesian tensor G-GNNs require fewer operations and are simpler to implement but may underperform compared to spherical tensor G-GNNs. We introduce a novel architecture, LAGNet, based on the Cartesian tensor G-GNN DeepDFT. The key modification is in the Message and Update blocks. The original DeepDFT update block can only scale vector features and cannot change their orientation. To improve the expressive power of the network, we remove these restrictions by rearranging the connections between the Message and Update block. We draw inspiration from the spherical tensor G-GNN's tensor product with trainable weights and augment the DeepDFT architecture with a set of scalarization operations, pools of vectors, scalars, and gating coefficients (see Fig. 4). Additionally, we use different RBF expansions for different probing points and discard the softcut function from the message block.

There are two main approaches for electronic structure analysis. The first approach uses information about each molecular orbital and typically requires prediction of the Hamiltonian. The second approach operates directly on the full electron density without requiring individual orbitals. Any method in this second category can be applied to electron density predicted by invDeepDFT, eqvDeepDFT, or LAGNet.

In this study, we do not evaluate LAGNet's performance on downstream tasks, but we outline several potential applications. Study [9] shows that partial atomic charges (Bader's charges from superposed atomic densities) can be obtained from neural-network-predicted electron density rather than traditional numerical methods. LAGNet can likewise compute these partial charges directly.

Using LAGNet's predicted electron density on a uniform grid, one can generate high-quality visualizations with chosen isosurfaces, as illustrated in the graphical abstract. Molecular electrostatic potentials (MEP) can also be calculated from the predicted density using SG-0 and the uniform grid. Other common analyses such as quantum theory of atoms in molecules (QTAIM) and non-covalent interaction analysis (NCI) may also be feasible with LAGNet outputs, but with limitations. Some analysis related to QTAIM requires knowledge of electron density matrix and all orbitals.

Computed electron density has proven useful as input for machine learning models in drug discovery, including predictions of protein-ligand binding affinity [8], host-guest interactions [10], virtual screening outcomes [11], and molecular classification [12]. We suppose some of these models can be combined with LAGNet, but these require further studies.

Limitations

The performance of DeepDFT, LAGNet, and similar graph models is inherently tied to the quality and scope of the training data. These models cannot generalize on data with atoms that are not presented during the training. ∇^2DFT datasets does not contain elements such as boron (B), phosphorus (P), selenium (Se) and iodine (I). In addition, ∇^2DFT does not cover radicals or charged molecules. However, some biological research requires anions, cations, and organic molecules with B, P, Se, I. LAGNet checkpoints trained in this study cannot handle these data.

In this study, LAGNet was trained on only a small subset of the entire Nabla2DFT dataset. We suppose that expanding the training dataset and significantly increasing the training time (up to 10,000 GPU hours) could lead to substantial improvements in model performance.

Future Work. We plan to release a series of enhanced LAGNet models trained on larger datasets, including the full ∇^2DFT dataset, which comprises about 15,000,000 conformations and 2,000,000 molecules. Future iterations may also include models specifically trained on anions and cations to better capture a wider range of chemical properties and systems.

Appendix A Extended Data

See Tables 1, 2, 3

See Figs. 15, 16, 17, 18, 19

Table 1 Ability of model to generalize on conformations and unseen molecules

model	single/multiple conformation in train data	split	mean	std	min	50%	max
test data splits (conformations, structures, scaffolds)							
invDeepDFT	multiple	conformations	0.98	0.16	0.67	0.95	2.71
invDeepDFT	multiple	structures	1.18	0.26	0.73	1.13	8.31
invDeepDFT	multiple	scaffolds	1.20	0.23	0.72	1.17	3.78
invDeepDFT	single	conformations	1.03	0.23	0.65	0.98	3.53
invDeepDFT	single	structures	1.40	0.31	0.70	1.35	10.09
invDeepDFT	single	scaffolds	1.44	0.32	0.77	1.38	13.34
eqvDeepDFT	multiple	conformations	0.77	0.09	0.58	0.75	2.17
eqvDeepDFT	multiple	structures	0.90	0.13	0.59	0.88	3.07
eqvDeepDFT	multiple	scaffolds	0.92	0.13	0.60	0.90	3.43
eqvDeepDFT	single	conformations	1.18	0.90	0.79	1.13	47.47
eqvDeepDFT	single	structures	1.34	0.25	0.79	1.30	10.33
eqvDeepDFT	single	scaffolds	1.34	0.23	0.80	1.31	4.72
LAGNet	multiple	conformations	0.35	0.08	0.25	0.33	1.62
LAGNet	multiple	structures	0.46	0.11	0.27	0.44	2.20
LAGNet	multiple	scaffolds	0.46	0.11	0.25	0.44	2.24
LAGNet	single	conformations	0.56	0.14	0.36	0.53	2.39
LAGNet	single	structures	0.73	0.15	0.38	0.71	2.85
LAGNet	single	scaffolds	0.74	0.15	0.43	0.72	2.78
train data splits (single/multiple conformations per molecule)							
invDeepDFT	multiple	multiple	0.97	0.15	0.67	0.95	2.23
invDeepDFT	multiple	single	0.97	0.15	0.67	0.94	1.81
invDeepDFT	single	multiple	1.06	0.23	0.63	1.00	3.79
invDeepDFT	single	single	0.98	0.18	0.63	0.94	2.29
eqvDeepDFT	multiple	multiple	0.77	0.08	0.58	0.76	1.53
eqvDeepDFT	multiple	single	0.76	0.08	0.60	0.74	1.21
eqvDeepDFT	single	multiple	1.18	0.21	0.78	1.14	7.59
eqvDeepDFT	single	single	1.08	0.12	0.78	1.06	1.60
LAGNet	multiple	multiple	0.35	0.06	0.25	0.34	1.26
LAGNet	multiple	single	0.35	0.07	0.26	0.33	0.84
LAGNet	single	multiple	0.58	0.14	0.34	0.55	2.43
LAGNet	single	single	0.52	0.09	0.36	0.50	1.03

invDeepDFT (D=6, F=128), eqvDeepDFT (D=6, F=128), LAGNet (D=4, F=128). NMAE(%) metric

Table 2 Ablation study of the extended DeepDFT

Ablation target	mean	std	min	50%	max
LAGNet (basic)	0.48	0.12	0.32	0.46	2.17
Branch #0 and #1, without #2	0.83	0.14	0.64	0.80	3.22
Softcut is turned on	0.58	0.13	0.39	0.55	2.90
Update-Message like in eqvDeepDFT	0.62	0.13	0.46	0.59	2.39
No messages from atom embedding	0.62	0.15	0.44	0.59	5.25

LAGNet (D=2, F=32) were trained on single conformation per molecule. NMAE(%) was computed on the conformation test split. Any modification reduces model performance

Table 3 Basic statistics for data in the cross-dataset experiment

		full dataset		evaluation subset		
Dataset	Subset	molecules	conformers	molecules	conformers	comment
Hutchison dataset	Astex	79	759	49	463	
	OmagePDB	468	4,663	412	1,248	
	OmegaCSD	161	4,378	142	4,238	
QMugs	QMugs	665,911	1,992,984	100	300	48-52 heavy atoms

We exclude charged molecules and molecules with P,I from Hutchison dataset. We add 100 molecules from Qmugs that bigger in number of heavy atoms than molecules from Hutchison dataset and nabla2DFT

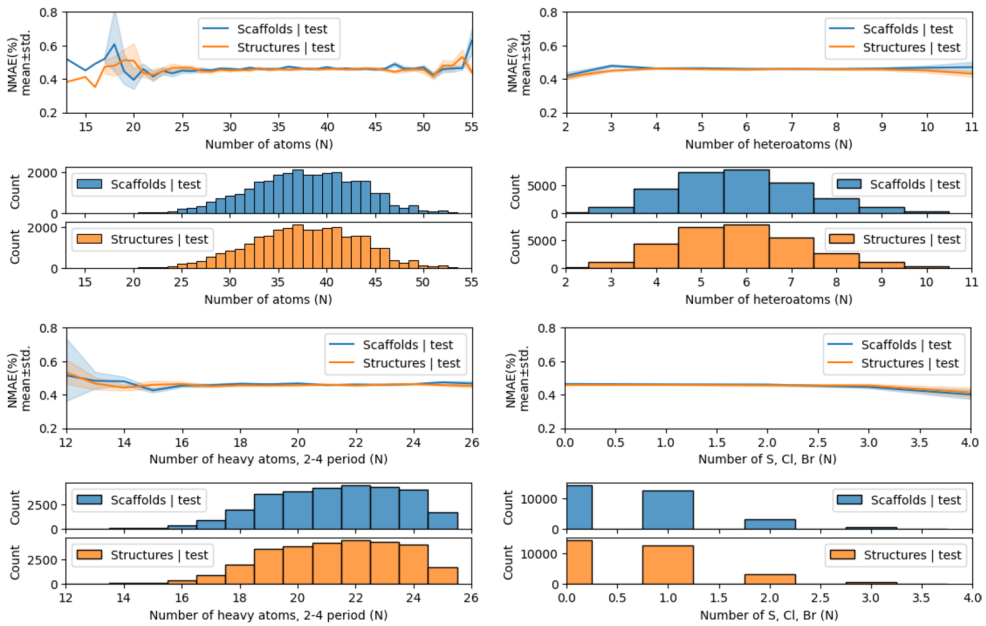


Fig. 15 LAGNet (D=4, F=128) performance on inference for structure and scaffold test sets, part 1. There are no statistically significant effect of molecule structure on average values of the target metric

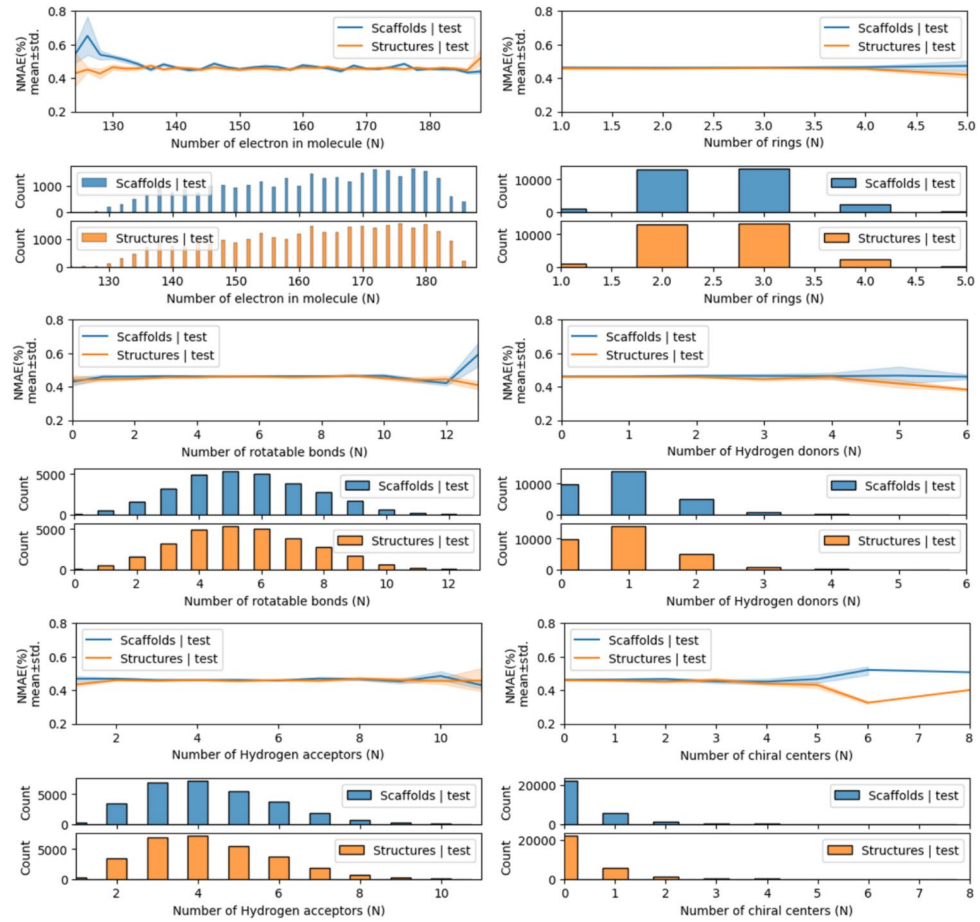


Fig. 16 LAGNet (D=4, F=128) performance on inference for structure and scaffold test sets, part 2. There are no statistically significant effect of molecule structure on average values of the target metric

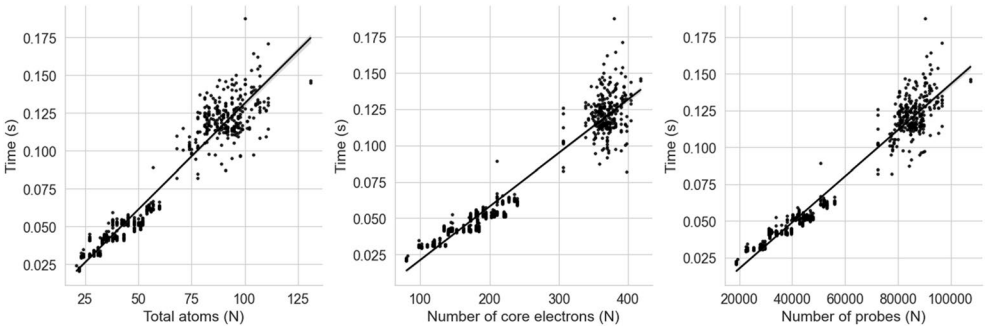


Fig. 17 LAGNet (D=4, F=128) performance on inference for Hutchison dataset (Astex subset) and Qmugs subsets. Performance are measured with SG-0

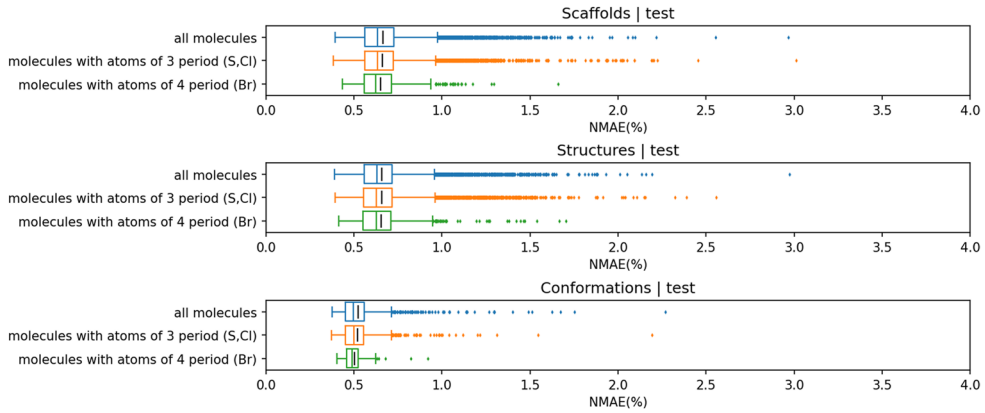


Fig. 18 LAGNet (D=4, F=128) error on test subsets of nablaDFT. Error is presented for molecules with atoms of 1–2nd,3rd,4yh period

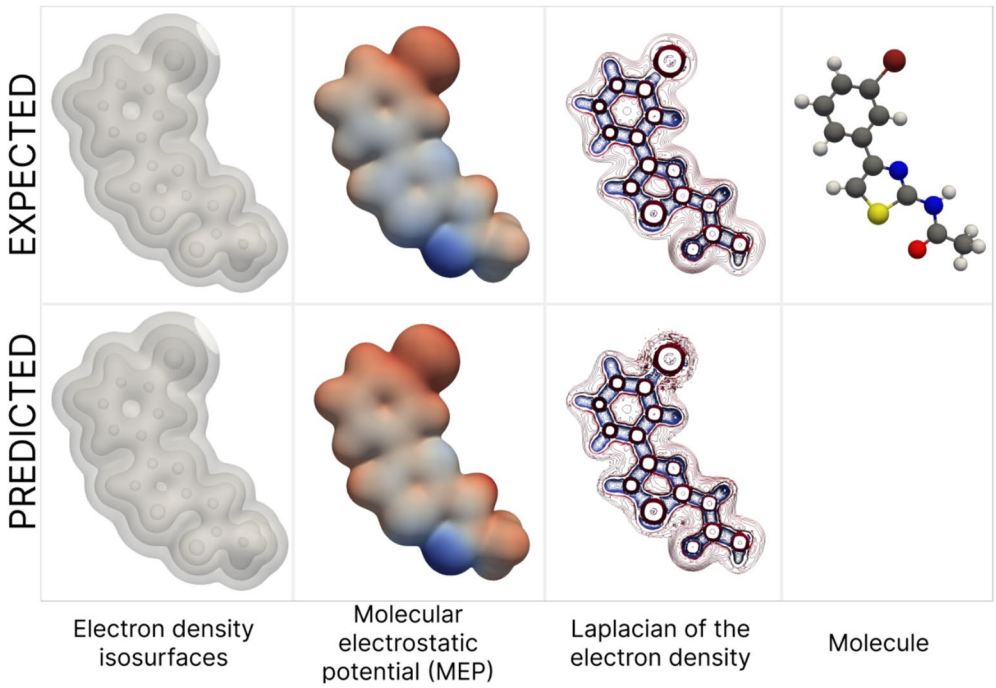


Fig. 19 LAGNet (D=4, F=128) using with random molecule from ∇^2DFT. Examples of downstream tasks with target and predicted electron density

Funding
The funding and computing resources for the study were provided by affiliated organizations.

Data availability
Data and source code The dataset is available at https://github.com/AIRI-Institute/nablaDFT. The source code of LAGNet is available at https://github.com/AIRI-Institute/lagnet-dft.

Declarations

Competing interests
No competing interests.

Author details
[1]AIRI, Kutuzovskiy Prospect, Moscow 121170, Russian Federation. [2]Ural Federal University, Mira st., Yekaterinburg 620002, Russian Federation. [3]École Polytechnique Fédérale de Lausanne (EPFL), Station Z, 1015 Lausanne, Switzerland. [4]Institute of System Programming, Alexander Solzhenitsyn st., Moscow 109004, Russian Federation.

Received: 28 December 2024 Accepted: 6 April 2025
Published online: 29 April 2025

References

1. Kohn W (1999) Nobel lecture: electronic structure of matter-wave functions and density functionals. Rev Modern Phys 71(5):1253

2. Matta CF (2014) Modeling biophysical and biological properties from the characteristics of the molecular electron density, electron localization and delocalization matrices, and the electrostatic potential. J Comput Chem 35(16):1165–1198
3. Mendez L, Henriquez G, Sirimulla S, Narayan M (2017) Looking back, looking forward at halogen bonding in drug discovery. Molecules 22(9):1397
4. Suresh CH, Remya GS, Anjalikrishna PK (2022) Molecular electrostatic potential analysis: a powerful tool to interpret and predict chemical reactivity. Wiley Interdiscip Rev: Comput Mol Sci 12(5):1601
5. Matta CF, Arabi AA (2011) Electron-density descriptors as predictors in quantitative structure-activity/property relationships and drug design. Future Med Chem 3(8):969–994
6. Cortés-Guzmán F, Rodríguez JI, Anderson JS (2023) Introduction to QTAIM and beyond. In: Fernando C-G, Anderson James S.M., Rodriguez Juan I (eds) Advances in quantum chemical topology beyond QTAIM. Elsevier, Amsterdam, pp 1–19
7. Saravanan K, Sugarthi S, Suganya S, Kumaradhas P (2022) Probing the intermolecular interactions, binding affinity, charge density distribution and dynamics of silibinin in dual targets AChE and BACE1: qtaim and molecular dynamics perspective. J Biomol Struct Dyn 40(23):12880–12894
8. Isert C, Atz K, Riniker S, Schneider G (2024) Exploring protein-ligand binding affinity prediction with electron density-based geometric deep learning. RSC Adv 14(7):4492–4502
9. Sunshine EM, Shuaibi M, Ulissi ZW, Kitchin JR (2023) Chemical properties from graph neural network-predicted electron densities. J Phys Chem C 127(48):23459–23466
10. Parrilla-Gutiérrez JM, Granda JM, Ayme J-F, Bajczyk MD, Wilbraham L, Cronin L (2024) Electron density-based GPT for optimization and suggestion of host-guest binders. Nat Comput Sci 4(3):200–209
11. Ma W, Zhang W, Le Y, Shi X, Xu Q, Xiao Y, Dou Y, Wang X, Zhou W, Peng W et al (2023) Using macromolecular electron densities to improve the enrichment of active compounds in virtual screening. Commun Chem 6(1):173
12. Singh S, Zeh G, Freiherr J, Bauer T, Türkmen I, Grasskamp AT (2024) Classification of substances by health hazard using deep neural networks and molecular electron densities. J Cheminform 16(1):45
13. Grisafi A, Bussy A, Salanne M, Vuilleumier R (2023) Predicting the charge density response in metal electrodes. Phys Rev Mater 7(12):125403
14. Grisafi A, Lewis AM, Rossi M, Ceriotti M (2022) Electronic-structure properties from atom-centered predictions of the electron density. J Chem Theory Comput 19(14):4451–4460
15. Lewis AM, Grisafi A, Ceriotti M, Rossi M (2021) Learning electron densities in the condensed phase. J Chem Theory Comput 17(11):7203–7214
16. Grisafi A, Fabrizio A, Meyer B, Wilkins DM, Corminboeuf C, Ceriotti M (2018) Transferable machine-learning model of the electron density. ACS Central Sci 5(1):57–64
17. Lv T, Zhong Z, Liang Y, Li F, Huang J, Zheng R (2023) Deep charge: deep learning model of electron density from a one-shot density functional theory calculation. Phys Rev B 108(23):235159
18. Rio BG, Phan B, Ramprasad R (2023) A deep learning framework to emulate density functional theory. npj Comput Mater 9(1):158
19. Alred JM, Bets KV, Xie Y, Yakobson BI (2018) Machine learning electron density in sulfur crosslinked carbon nanotubes. Compos Sci Technol 166:3–9
20. Chandrasekaran A, Kamal D, Batra R, Kim C, Chen L, Ramprasad R (2019) Solving the electronic structure problem with machine learning. npj Comput Mater 5(1):22
21. Cheng C, Peng J (2024) Equivariant neural operator learning with graphon convolution. Adv Neural Inform Process Syst 36:61960
22. Sinitskiy AV, Pande VS (2018) Deep neural network computes electron densities and energies of a large set of organic molecules faster than density functional theory (DFT). arXiv preprint. https://doi.org/10.4855/arXiv.1809.02723
23. Kim S, Ahn S (2024) Gaussian plane-wave neural operator for electron density estimation. arXiv preprint. https://doi.org/10.48550/arXiv.2402.04278
24. Fu X, Rosen A, Bystrom K, Wang R, Musaelian A, Kozinsky B, Smidt T, Jaakkola T (2024) A recipe for charge density prediction. Adv Neural Inform Process Syst 37:9727–52
25. Cuevas-Zuviría B, Pacios LF (2020) Analytical model of electron density and its machine learning inference. J Chem Inform Model 60(8):3831–3842
26. Jørgensen PB, Bhowmik A (2022) Equivariant graph neural networks for fast electron density estimation of molecules, liquids, and solids. npj Comput Mater 8(1):183
27. Jørgensen PB, Bhowmik A (2020) DeepDFT: neural message passing network for accurate charge density prediction. arXiv preprint. https://doi.org/10.48550/arXiv.2011.03346
28. Yu H, Xu Z, Qian X, Qian X, Ji S (2023) Efficient and equivariant graph networks for predicting quantum Hamiltonian. In: International Conference on Machine Learning. PMLR, pp 40412–40424
29. Unke O, Bogojeski M, Gastegger M, Geiger M, Smidt T, Müller K-R (2021) SE(3)-equivariant prediction of molecular wavefunctions and electronic densities. Adv Neural Inform Process Syst 34:14434–14447
30. Qiao Z, Christensen AS, Welborn M, Manby FR, Anandkumar A, Miller TF III (2022) Informing geometric deep learning with electronic interactions to accelerate quantum chemistry. Proc Natl Acad Sci 119(31):2205221119
31. Ruddigkeit L, Van Deursen R, Blum LC, Reymond J-L (2012) Enumeration of 166 billion organic small molecules in the chemical universe database GDB-17. J Chemical Inform Model 52(11):2864–2875
32. Khrabrov K, Ber A, Tsypin A, Ushenin K, Rumiantsev E, Telepov A, Protasov D, Shenbin I, Alekseev A, Shirokikh M et al (2024) ∇^2DFT: a universal quantum chemistry dataset of drug-like molecules and a benchmark for neural network potentials. arXiv preprint. https://doi.org/10.48550/arXiv.2406.14347
33. Isert C, Atz K, Jiménez-Luna J, Schneider G (2022) QMugs, quantum mechanical properties of drug-like molecules. Sci Data 9(1):273
34. Eastman P, Behara PK, Dotson DL, Galvelis R, Herr JE, Horton JT, Mao Y, Chodera JD, Pritchard BP, Wang Y et al (2023) SPICE, a dataset of drug-like molecules and peptides for training machine learning potentials. Sci Data 10(1):11
35. Lu J, Xia S, Lu J, Zhang Y (2021) Dataset construction to explore chemical space with 3D geometry and deep learning. J Chem Inform Model 61(3):1095–1104
36. Christensen AS, Sirumalla SK, Qiao Z, O'Connor MB, Smith DG, Ding F, Bygrave PJ, Anandkumar A, Welborn M, Manby FR et al (2021) OrbNet Denali: a machine learning potential for biological and organic chemistry with semi-empirical cost and dft accuracy. J Chem Phys 0061990. https://doi.org/10.1063/5.0061990
37. Kresse G, Furthmüller J (1996) Efficient iterative schemes for *ab initio* total-energy calculations using a plane-wave basis set. Phys Rev B 54(16):11169
38. Giannozzi P, Baroni S, Bonini N, Calandra M, Car R, Cavazzoni C, Ceresoli D, Chiarotti GL, Cococcioni M, Dabo I et al (2009) Quantum Espresso: a modular and open-source software project for quantum simulations of materials. J Phys: Condens Matter 21(39):395502
39. Enkovaara J, Rostgaard C, Mortensen JJ, Chen J, Dułak M, Ferrighi L, Gavnholt J, Glinsvad C, Haikola V, Hansen H et al (2010) Electronic structure calculations with gpaw: a real-space implementation of the projector augmented-wave method. J Phys: Condens Matter 22(25):253202
40. Kühne TD, Iannuzzi M, Del Ben M, Rybkin VV, Seewald P, Stein F, Laino T, Khaliullin RZ, Schütt O, Schiffmann F et al (2020) CP2K: an electronic structure and molecular dynamics software package-quickstep: efficient and accurate electronic structure calculations. J Chem Phys. https://doi.org/10.1063/5.0007045
41. Koker T, Quigley K, Taw E, Tibbetts K, Li L (2024) Higher-order equivariant neural networks for charge density prediction in materials. npj Comput Mater 10(1):161
42. Cuevas-Zuviría B, Pacios LF (2021) Machine learning of analytical electron density in large molecules through message-passing. J Chem Inform Model 61(6):2658–2666
43. Smith DG, Burns LA, Simmonett AC, Parrish RM, Schieber MC, Galvelis R, Kraus P, Kruse H, Di Remigio R, Alenaizan A et al (2020) Psi4 1.4: open-source software for high-throughput quantum chemistry. J Chem Phys. https://doi.org/10.1063/5.0006002
44. Sun Q, Zhang X, Banerjee S, Bao P, Barbry M, Blunt NS, Bogdanov NA, Booth GH, Chen J, Cui Z-H et al (2020) Recent developments in the pyscf program package. J Chem Phys. https://doi.org/10.1063/5.0006074
45. Furche F, Ahlrichs R, Hättig C, Klopper W, Sierka M, Weigend F (2014) Turbomole. Wiley Interdiscip Rev: Comput Mol Sci 4(2):91–100

46. ...Frisch MJ, Trucks GW, Schlegel HB, Scuseria GE, Robb MA, Cheeseman JR, Scalmani G, Barone V, Petersson GA, Nakatsuji H, Li X, Caricato M, Marenich AV, Bloino J, Janesko BG, Gomperts R, Mennucci B, Hratchian HP, Ortiz JV, Izmaylov AF, Sonnenberg JL, Williams-Young D, Ding F, Lipparini F, Egidi F, Goings J, Peng B, Petrone A, Henderson T, Ranasinghe D, Zakrzewski VG, Gao J, Rega N, Zheng G, Liang W, Hada M, Ehara M, Toyota K, Fukuda R, Hasegawa J, Ishida M, Nakajima T, Honda Y, Kitao O, Nakai H, Vreven T, Throssell K, Montgomery JA Jr, Peralta JE, Ogliaro F, Bearpark MJ, Heyd JJ, Brothers EN, Kudin KN, Staroverov VN, Keith TA, Kobayashi R, Normand J, Raghavachari K, Rendell AP, Burant JC, Iyengar SS, Tomasi J, Cossi M, Millam JM, Klene M, Adamo C, Cammi R, Ochterski JW, Martin RL, Morokuma K, Farkas O, Foresman JB, Fox DJ (2016) Gaussian 16 Revision C.01. Gaussian Inc., Wallingford
47. Giannozzi P (2013) Numerical methods in quantum mechanics. University of Udine, Udine
48. Folmsbee D, Hutchison G (2021) Assessing conformer energies using electronic structure and machine learning methods. Int J Quantum Chem 121(1):26381
49. Dasgupta S, Herbert JM (2017) Standard grids for high-precision integration of modern density functionals: SG-2 and SG-3. J Comput Chem 38(12):869–882
50. Koch W, Holthausen MC (2015) A chemist's guide to density functional theory. John Wiley & Sons, Hoboken
51. Virtanen P, Gommers R, Oliphant TE, Haberland M, Reddy T, Cournapeau D, Burovski E, Peterson P, Weckesser W, Bright J, van der Walt SJ, Brett M, Wilson J, Millman KJ, Mayorov N, Nelson ARJ, Jones E, Kern R, Larson E, Carey CJ, Polat İ, Feng Y, Moore EW, VanderPlas J, Laxalde D, Perktold J, Cimrman R, Henriksen I, Quintero EA, Harris CR, Archibald AM, Ribeiro AH, Pedregosa F, van Mulbregt P (2020) SciPy 1.0 contributors: SciPy 1.0: fundamental algorithms for scientific computing in Python. Nat Methods 17:261–272. https://doi.org/10.1038/s41592-019-0686-2
52. Duval A, Mathis SV, Joshi CK, Schmidt V, Miret S, Malliaros FD, Cohen T, Lio P, Bengio Y, Bronstein M (2023) A hitchhiker's guide to geometric GNNs for 3d atomic systems. arXiv preprint https://doi.org/10.48550/arXiv.2312.07511
53. Schütt K, Kindermans P-J, Sauceda Felix HE, Chmiela S, Tkatchenko A, Müller K-R (2017) SchNet: A continuous-filter convolutional neural network for modeling quantum interactions. Adv Neural Inf Proc Syst 30
54. Schaback R (2007) A practical guide to radial basis functions. Electron Resour 11:1–12
55. Schütt KT, Hessmann SS, Gebauer NW, Lederer J, Gastegger M (2023) SchNetPack 2.0: a neural network toolbox for atomistic machine learning. J Chem Phys. https://doi.org/10.1063/5.0138367
56. Geiger M, Smidt T (2022) e3nn: Euclidean neural networks. arXiv preprint https://doi.org/10.48550/arXiv.2207.09453
57. Schutt K, Kessel P, Gastegger M, Nicoli K, Tkatchenko A, Muller K-R (2018) SchNetPack: a deep learning toolbox for atomistic systems. J Chem Theory Comput 15(1):448–455
58. Fei J, Deng Z (2024) Rotation invariance and equivariance in 3D deep learning: a survey. Artif Intell Rev 57(7):168
59. Quiñonero-Candela J, Sugiyama M, Schwaighofer A, Lawrence ND (2022) Dataset shift in machine learning. Mit Press, Cambridge

Publisher's Note