PI-EDG: Physics-Informed Full-Space Electron Density Generation from Molecular Geometry

Jiahang Shen[1], **Zhixiang Cheng**[2,3,4], **Keke Chen**[2,3,4], **Yingzhuo Tu**[2,3,4], **Xiangxiang Zeng**[3,4], **Wenjie Du**[1,†], **Hongxin Xiang**[2,3,4,†]

[1]University of Science and Technology of China, Hefei, China
[2]College of Computer Science and Electronic Engineering, Hunan University, Changsha, China
[3]State Key Laboratory of Chemo and Biosensing, Institute of AI4Science, Hunan University, Changsha, China
[4]Yuelushan Laboratory, Changsha, China

Abstract

Full-space electron density generation supports the analysis of molecular quantum states, ground-state properties, and chemical interactions. High-fidelity density functional theory calculations require costly self-consistent iterations, which limit large-scale quantum-chemical modeling. We introduce PI-EDG, a physics-informed voxel-space framework that maps molecular geometries directly to full-space electron-density fields. PI-EDG combines quantum-mechanical prior injection, hybrid FNO-CNN modeling of local and non-local interactions, magnitude-aware DS-LMoE routing across extreme density ranges, and inference-time electron-number normalization. On EDBench, PI-EDG achieves an MAE of 3.405×10^{-3} under the original output protocol and an average inference time of 133.4 ms per molecule. Under a common electron-number normalization applied to all evaluated methods, PI-EDG retains the lowest MAE among the compared models. Its predicted densities also yield the lowest external-region molecular electrostatic potential (MEP) MAE of 5.300×10^{-2} Hartree/e. PI-EDG provides an efficient route for generating approximate full-space electron-density fields from molecular geometry. Our code can be found at `https://github.com/jiahangshen/PI-EDG`.

1 Introduction

Electron density (ED) is a fundamental physical quantity in quantum mechanics, describing the probability distribution of electrons in a three-dimensional space [1]. According to the Hohenberg–Kohn (HK) theorem, the properties of the ground state of a molecular system are uniquely determined by its ED [2]. Therefore, ED plays a central role in modeling molecular quantum states, chemical interactions, and ground-state properties [3–6]. Unfortunately, the traditional methods for obtaining electron density are highly dependent on density functional theory (DFT) calculations based on first principles [7, 8]. This computational iteration process is extremely time-consuming (Figure 1A). To break this computational bottleneck, using deep learning to predict electron density directly from molecular geometry has become an extremely valuable research task.

In recent years, existing ED prediction methods (Figure 1B) can be broadly grouped into three categories: coordinate-query methods, basis-based methods, and function-space mapping methods. Coordinate-query methods evaluate density at spatial points conditioned on molecular structures [9–11], but reconstructing high-resolution fields requires costly dense queries. EDBench reduces this cost through density thresholds and point-cloud representations [6], which can omit chemically relevant low-density information. Basis-based methods predict coefficients of atom-centered functions [12–15], but are constrained by predefined bases and often require symmetry-breaking heuristics, nodal

[†]Correspondence: `duwenjie@mail.ustc.edu.cn`, `xianghx@hnu.edu.cn`

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

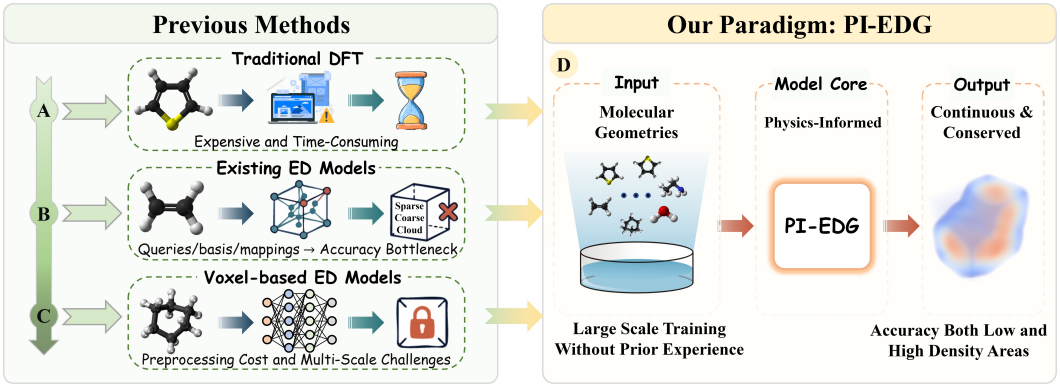


Figure 1: **Comparison of PI-EDG with Existing Methods.** (A) Traditional DFT relies on computationally expensive self-consistent calculations. (B) Existing ED prediction methods use coordinate queries, basis expansions, or function-space mappings, facing efficiency or accuracy limitations. (C) Voxel-based ED models often require SAD-grid preprocessing and struggle to capture the wide dynamic range of electron density. (D) PI-EDG directly generates full-space electron density from molecular geometry using physics-informed modeling, followed by electron-number normalization.

rules, or full-grid reconstruction. Function-space mappings predict continuous density fields [16], but purely operator-based models can oversmooth sharp near-nuclear peaks. These limitations make efficient and accurate prediction across the full density range challenging.

Inspired by advances in computer vision [17, 18], voxel-based ED predictors directly generate complete density grids [19, 20], avoiding coordinate-wise queries and explicit basis reconstruction (Figure 1C). However, these methods typically use spherical neutral atomic density (SAD) grids as intermediate inputs, adding preprocessing cost. Pure CNN architectures also face the challenge of modeling the wide density range, with sharp nuclear peaks and low-density tails coexisting within the same grid.

As illustrated in Figure 1D, we propose the Physics-Informed Electron Density Generator (PI-EDG), a voxel-space framework that generates full-space electron-density fields directly from molecular geometry. First, to reduce input-preparation cost, we construct explicit quantum-mechanical prior fields directly from molecular geometry, including the external potential v_{ext} and nuclear charge peak Z_{peak}. Second, to model sharp local density variations and long-range electrostatic interactions, we introduce a hybrid FNO-CNN bottleneck that combines local convolutions with global spectral modeling. Third, to address the numerical imbalance across density scales, we combine cube-root regression with a Dual-Space Learnable Mixture of Experts (DS-LMoE), whose experts specialize in different density regimes. Finally, the predicted grid is rescaled to the neutral electron count at inference.

In summary, our contributions can be summarized as follows:

- We propose PI-EDG, a physics-informed architecture for direct full-space electron-density generation. This architecture integrates explicit physical prior injection and a hybrid FNO-CNN module, achieving collaborative modeling of local features and global long-range interactions.
- PI-EDG combines a cube-root transformation with soft mixture-of-experts routing to mitigate the numerical imbalance across density scales. At inference, the predicted grid is rescaled to the known electron count of the neutral molecule, enforcing the corresponding global integral constraint.
- Experiments show that PI-EDG has the lowest density MAE both when each method uses its original output and when all methods receive the same electron-number normalization. It also gives the lowest external-region molecular electrostatic potential (MEP) error among the compared models and generates a complete 64^3 density grid with an average end-to-end inference time of 133.4 ms per molecule.

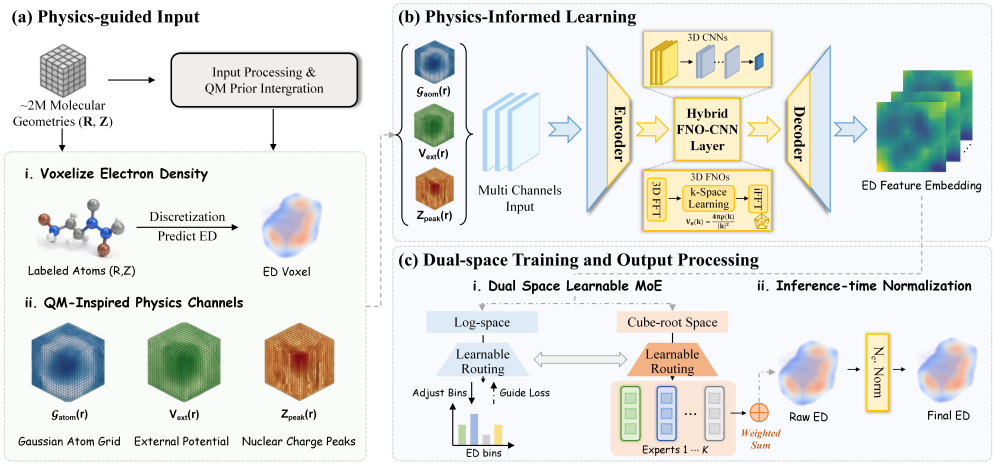


Figure 2: **The overall architecture of PI-EDG.** (a) Discrete atomic configurations are transformed into physical prior fields grounded in quantum mechanics. (b) A multi-scale backbone equipped with a hybrid FNO-CNN bottleneck jointly captures local high-frequency topological variations and global long-range electrostatic interactions. (c) The DS-LMoE module routes features across logarithmic and cube-root spaces, and the generated grid is normalized to the neutral electron count at inference.

2 Methodology

We introduce the PI-EDG, which builds upon the 3D U-Net [17] architecture to generate full-space electron-density fields. As illustrated in Figure 2, we enhance the standard 3D U-Net with three integrated components: (a) Multi-scale physics-injected backbone to reinforce spatial priors (see 2.2); (b) Hybrid FNO-CNN bottleneck to model long-range interactions (see 2.3); (c) Dual-Space Learnable Mixture of Experts module to manage extreme scale variance (see 2.4).

2.1 Problem Formulation

Given a molecular geometry with M atoms defined by their coordinates $\mathbf{R}$ and nuclear charges $\mathbf{Z}$, we learn a mapping f_θ parameterized by model parameters θ:

$$f_\theta : \{\mathbf{R}, \mathbf{Z}\} \to \rho(\mathbf{r}), \quad where \quad \mathbf{R} = \{\mathbf{R}_A\}_{A=1}^{M}, \quad \mathbf{Z} = \{Z_A\}_{A=1}^{M}. \tag{1}$$

Here, $\mathbf{r} \in \mathbb{R}^3$ is a spatial coordinate, $\mathbf{R}_A$ denotes the three-dimensional coordinate of atom A, and Z_A denotes its nuclear charge. The goal is to predict the continuous ground-state ED field $\rho(\mathbf{r})$. We discretize the continuous space into a regular voxel grid with depth D, height H, and width W.

2.2 Quantum Mechanics-Guided Representation

According to the Hohenberg-Kohn theorem [2], the ground-state ED $\rho(\mathbf{r})$ is a unique functional of the external potential $v_{ext}(\mathbf{r})$. Since $v_{\text{ext}}(\mathbf{r})$ encodes the Coulomb attraction between electrons and nuclei, it directly contributes to the electron–nucleus interaction energy:

$$E_{ne} = \int \rho(\mathbf{r}) \cdot v_{ext}(\mathbf{r}) d\mathbf{r}. \tag{2}$$

To incorporate this physics into the network, we transform the discrete molecular inputs $(\mathbf{R}, \mathbf{Z})$ into a continuous multi-channel 3D grid evaluated at each voxel coordinate $\mathbf{r}$. This grid contains three physics-guided fields that respectively encode atomic occupancy, electron–nucleus attraction, and exact nuclear charge peak. First, a Gaussian-smeared atom grid $\mathcal{G}_{atom}(\mathbf{r})$ represents each atom on the grid using radial basis functions:

$$\mathcal{G}_{atom}(\mathbf{r}) = \sum_{A=1}^{M} Z_A \cdot \exp(-\frac{\|\mathbf{r} - \mathbf{R}_A\|^2}{2\sigma_G^2}), \tag{3}$$

where σ_G controls the Gaussian smearing width. Second, the external Coulomb potential field evaluates the electrostatic interaction across the space. To avoid singularities, we use a small positive constant ϵ_c:

$$v_{\text{ext}}(\mathbf{r}) = \sum_{A=1}^{M} -\frac{Z_A}{\|\mathbf{r} - \mathbf{R}_A\| + \epsilon_c}. \tag{4}$$

This regularization prevents division by zero near nuclear centers. Third, to encode nuclear positions at voxel resolution within the discrete grid, we construct a charge peak field:

$$Z_{peak}(\mathbf{r}) = \sum_{A=1}^{M} Z_A \cdot \mathbb{I}(\mathbf{r} = \mathbf{r}_A^{grid}), \tag{5}$$

where $\mathbf{r}_A^{grid}$ denotes the center coordinate of the voxel containing atom A, and $\mathbb{I}(\cdot)$ is the indicator function that equals 1 if the condition is satisfied and 0 otherwise. These three computed fields serve as the initial input.

To prevent the loss of these physical priors during the downsampling operations of the U-Net backbone, we apply a Multi-scale Physics Injection mechanism. At each encoder and decoder spatial resolution, the scaled physical fields v_{ext}^{scaled} and Z_{peak}^{scaled} are concatenated with the intermediate feature maps. This ensures the model retains spatial physical references across all receptive fields.

2.3 Hybrid FNO-CNN Bottleneck

Predicting ED requires capturing both sharp local variations near nuclei and non-local long-range interactions such as the Hartree electron-electron repulsion energy E_H. This energy represents the classical electrostatic self-interaction of the electron cloud and is defined as:

$$E_H[\rho] = \frac{1}{2} \int \rho(\mathbf{r}) V_H(\mathbf{r}) d\mathbf{r}. \tag{6}$$

Directly evaluating the Hartree potential $V_H(\mathbf{r})$ through double integration is computationally expensive because it is a non-local term where the potential at any point depends on the density across the entire space. To accelerate this, V_H is typically obtained by solving the Poisson equation in real space:

$$\nabla^2 V_H(\mathbf{r}) = -4\pi\rho(\mathbf{r}). \tag{7}$$

Let $\boldsymbol{\xi}$ denote a reciprocal-space wavevector, and let $V_H(\boldsymbol{\xi})$ and $\rho(\boldsymbol{\xi})$ denote the Fourier transforms of the corresponding real-space fields. Applying a Fast Fourier Transform (FFT) gives

$$V_H(\boldsymbol{\xi}) = \frac{4\pi\rho(\boldsymbol{\xi})}{\|\boldsymbol{\xi}\|^2}, \qquad \boldsymbol{\xi} \neq \mathbf{0}. \tag{8}$$

Motivated by the Fourier-domain Green's function of the Poisson operator, whose kernel is proportional to $1/\|\boldsymbol{\xi}\|^2$, we replace the standard U-Net bottleneck with a hybrid FNO-CNN block. Both branches process the same 3D feature tensor. The FNO branch applies learnable complex weights to the retained low-frequency Fourier modes and transforms the result back via an inverse FFT, enabling global information exchange. The CNN branch uses local convolutions to preserve sharp variations near nuclei and chemical bonds. Appendix B.1 provides the Fourier-domain derivation.

2.4 Dual-Space Learnable Mixture of Experts

On a full molecular grid, ED ranges from approximately 10^2 near nuclei to 10^{-34} in vacuum regions. Most voxels have very low density, while large values are confined to small near-nuclear regions, making it difficult for a single regression head to fit all density scales. PI-EDG retains the complete grid and combines expert outputs using learned voxel-wise soft weights, with each expert assigned to a different density range.

DS-LMoE uses log-density guidance for routing and cube-root density for regression. The guide predicts $g(\mathbf{r}) \approx \log_{10}(\rho(\mathbf{r}) + \epsilon)$, where $\epsilon = 10^{-34}$ sets the numerical floor near zero density. Log-space intervals represent different density orders of magnitude for expert assignment. The guide output is detached before routing, while the experts regress $\rho^{1/3}$ to reduce the numerical range while retaining local spatial variation.

The router divides the log-density range among K experts using ordered learnable thresholds. We define finite expert bounds $\{t_i\}_{i=0}^{K}$ with $t_0 = -34$ and parameterize the remaining bounds as

$$t_i = t_0 + \sum_{j=1}^{i} \exp(\alpha_j), \qquad i = 1, \ldots, K. \tag{9}$$

Here, α_j are unconstrained learnable increments; the exponential keeps the bounds ordered. We initialize $\alpha_j = 0$, so $t_i = t_0 + i$ and each expert initially covers one decade. These finite bounds are optimized with the density predictor. For routing only, we introduce sentinel boundaries

$$s_0 = -\infty, \qquad s_K = +\infty, \qquad s_k = t_k \text{ for } 1 \le k \le K-1. \tag{10}$$

The routing weight of expert k is

$$\pi_k(\mathbf{r}) = \mathrm{sigmoid}(\tau(g(\mathbf{r}) - s_{k-1})) - \mathrm{sigmoid}(\tau(g(\mathbf{r}) - s_k)), \tag{11}$$

where τ controls routing sharpness. The sentinel boundaries give $\sum_k \pi_k(\mathbf{r}) = 1$. During training, the density loss updates the finite bounds through the routing weights and expert outputs. Detaching $g(\mathbf{r})$ keeps this gradient out of the guide branch, which is supervised by the guide loss in Section 2.5. Appendix B gives the full derivation.

For expert k, the finite log-density bounds define a cube-root output interval:

$$\begin{aligned} a_k &= 10^{t_{k-1}/3}, & b_k &= 10^{t_k/3}, \\ y_k(\mathbf{r}) &= \mathrm{sigmoid}(\mathrm{Expert}_k(\mathbf{h}(\mathbf{r}))) \in [0,1], & E_k(\mathbf{r}) &= a_k + y_k(\mathbf{r})(b_k - a_k). \end{aligned} \tag{12}$$

Here, a_k and b_k are the lower and upper bounds of the assigned interval, and Expert_k is the k-th expert head. The expert converts the shared feature $\mathbf{h}(\mathbf{r})$ to $y_k(\mathbf{r}) \in [0,1]$ and then maps it into $[a_k, b_k]$. This keeps each expert output within its current density range while the thresholds remain learnable. The weighted expert output gives the cube-root prediction and the corresponding density:

$$\tilde{q}(\mathbf{r}) = \sum_k \pi_k(\mathbf{r}) E_k(\mathbf{r}), \qquad \tilde{\rho}_{fused}(\mathbf{r}) = \tilde{q}(\mathbf{r})^3. \tag{13}$$

2.5 Training Objective and Inference-time Electron-number Normalization

The network is trained end-to-end with two supervision terms. Let ρ_{ref} denote the reference electron density. The global loss $\mathcal{L}_{global}$ supervises the fused density in cube-root space, and the guide loss $\mathcal{L}_{guide}$ trains the log-density field used by the router. With loss weights λ_{global} and λ_{guide}, both terms use the squared $\mathcal{L}_2$ norm:

$$\mathcal{L}_{total} = \lambda_{global} \|\tilde{q} - \rho_{ref}^{1/3}\|_2^2 + \lambda_{guide} \|g(\mathbf{r}) - \log_{10}(\rho_{ref} + \epsilon)\|_2^2. \tag{14}$$

After the forward pass at inference, let $\tilde{\rho}_i := \tilde{\rho}_{fused}(\mathbf{r}_i)$ be the raw prediction at voxel i. It is clipped at zero and rescaled to the neutral electron count:

$$\rho_i^{+} = \max(\tilde{\rho}_i, 0)\,, \tag{15}$$

$$\rho_i^{\mathrm{final}} = N_e \frac{\rho_i^{+}}{\sum_j \rho_j^{+} \Delta V + \epsilon_{\mathrm{num}}}, \tag{16}$$

Here, i and j denote voxel indices, $N_e = \sum_A Z_A$ is the electron count of a neutral molecule, ΔV is the voxel volume, and ϵ_{num} prevents division by zero. We apply this normalization once after inference. Appendix B.5 gives the implementation details, and Appendix C.6 applies the same procedure to all compared methods.

3 Experiments

In this section, we conduct comprehensive experiments to evaluate PI-EDG using global and density-resolved errors, controlled electron-number normalization, and external-region MEP. We further examine runtime and resolution scaling, molecule-level and subgroup variation, and the contribution of each model component.

Table 1: **Quantitative comparison on the EDBench dataset.** PI-EDG is compared with electron-density-specific models and general 3D voxel regression architectures under each method's original output protocol. PI-EDG's protocol includes inference-time neutral-electron normalization. All values are reported as mean ± standard deviation across three independent runs. For each metric, the best result is highlighted in **bold**, and the second-best result is underlined.

Method	Category	MAE ($\times 10^{-3}$) ↓	RMSE ($\times 10^{-1}$) ↓	Pearson ↑
General 3D Voxel Regression				
3D U-Net [17]	3D CNN	$3.874_{\pm 0.043}$	$1.816_{\pm 0.026}$	$0.679_{\pm 0.011}$
V-Net [22]	3D CNN	$8.203_{\pm 0.092}$	$\underline{1.801}_{\pm 0.031}$	$0.684_{\pm 0.014}$
3D ResNet [23, 24]	3D CNN	$6.212_{\pm 0.071}$	$2.224_{\pm 0.038}$	$0.437_{\pm 0.019}$
Swin3D [25]	3D Transformer	$5.615_{\pm 0.064}$	$1.889_{\pm 0.029}$	$\underline{0.689}_{\pm 0.012}$
UNETR [26]	3D Transformer	$6.208_{\pm 0.068}$	$1.808_{\pm 0.027}$	$0.683_{\pm 0.013}$
Electron-Density-Specific Models				
DeepDFT [10]	Graph-based	$8.137_{\pm 0.095}$	$2.176_{\pm 0.041}$	$0.489_{\pm 0.021}$
ChargE3Net [11]	Graph-based	$\underline{3.419}_{\pm 0.039}$	$1.936_{\pm 0.033}$	$0.673_{\pm 0.015}$
ELECTRA [14]	Basis-based	$3.771_{\pm 0.046}$	$2.042_{\pm 0.035}$	$0.574_{\pm 0.018}$
SCDP [15]	Basis-based	$3.692_{\pm 0.078}$	$2.146_{\pm 0.047}$	$0.556_{\pm 0.014}$
GPWNO [16]	Basis-based	$4.201_{\pm 0.052}$	$1.900_{\pm 0.032}$	$0.677_{\pm 0.014}$
3D-ResNet-DFT [20]	Voxel-based	$4.550_{\pm 0.057}$	$2.203_{\pm 0.039}$	$0.629_{\pm 0.017}$
PI-EDG (Ours)	**Voxel-based**	$\mathbf{3.405}_{\pm 0.006}$	$\mathbf{1.786}_{\pm 0.024}$	$\mathbf{0.691}_{\pm 0.012}$

3.1 Implementation and Experimental Configurations

Datasets. We evaluate our framework on the large-scale EDBench dataset, which provides high-fidelity density functional theory calculations for ~2M molecular geometries [6]. Following [21], the continuous real space is discretized into regular voxel grids ($64 \times 64 \times 64$) for all evaluation purposes. Dataset preprocessing details are provided in Appendix C.1.

Baselines. To comprehensively evaluate the performance of PI-EDG, we compare it with 11 baseline methods, which are grouped into two categories according to their methodological characteristics. The first category consists of models specifically designed for predicting electron density. This includes models based on graphs and coordinates (DeepDFT [10] and ChargE3Net [11]), analytical basis set predictors (ELECTRA [14], SCDP [15], GPWNO [16]), and three-dimensional voxel generators based on grids (3D-ResNet-DFT [20]). The second category is a general three-dimensional voxel regression model to verify whether a purely computer vision architecture can autonomously learn physical fields. This includes classic three-dimensional convolutional neural networks (3D U-Net [17], V-Net [22], 3D ResNet [23, 24]) and three-dimensional visual transformers (Swin3D [25], UNETR [26]). All baselines follow exactly their official code and parameters. Detailed baseline protocols and full-grid reconstruction rules are provided in Appendix C.3.

Evaluation Metrics. We evaluate local prediction accuracy using MAE, RMSE, and Pearson correlation. After inference, PI-EDG clips negative predictions and rescales the remaining values with one shared factor. The factor is chosen so that the grid integral equals the neutral electron count. For the controlled comparison, we apply the same procedure to the raw full-grid prediction of every method. Density-stratified MAE, near-zero LogMAE, and high-density NMAE assess errors across density regimes. We also evaluate the MEP derived from each predicted density. Complete definitions are given in Appendix C.4.

Implementation Details. PI-EDG is implemented with Python 3.8.20, PyTorch 2.4.1+cu121, CUDA Toolkit 12.1, and cuDNN 9.1.0, and is optimized with AdamW under a cosine annealing learning-rate schedule. The reported main results use DS-LMoE with $K = 37$ experts, initialized with one expert per density decade. The dual-supervision loss weights are set to $\lambda_{\text{global}} = 1.0$ and $\lambda_{\text{guide}} = 1 \times 10^{-5}$. All experiments are conducted on four NVIDIA A800 80GB GPUs. Additional implementation details and statistical reporting protocols are provided in Appendices C.2 and C.5.

3.2 PI-EDG Performance Comparison

Comparison with Global Metrics. Table 1 compares PI-EDG with two groups of baselines under their original output protocols. Among general 3D voxel regression models, 3D U-Net obtains the

lowest MAE of 3.874×10^{-3}, while V-Net and Swin3D achieve competitive RMSE and Pearson correlation, respectively. PI-EDG reduces the MAE by 12.1% relative to 3D U-Net and further improves the best RMSE and Pearson results in this group from 1.801×10^{-1} to 1.786×10^{-1} and from 0.689 to 0.691. Among electron-density-specific models, ChargE3Net provides the closest MAE at 3.419×10^{-3}, whereas GPWNO gives the strongest RMSE and Pearson results within this group. PI-EDG reaches 3.405×10^{-3} MAE, 1.786×10^{-1} RMSE, and 0.691 Pearson correlation. Relative to the corresponding group-best values, the MAE and RMSE decrease by 0.4% and 6.0%, and Pearson correlation increases by 2.1%. These results show that PI-EDG maintains consistently strong voxel-wise accuracy across the three global metrics. The following experiment applies a common electron-number normalization to the principal density models for a controlled comparison of their spatial predictions.

Electron-number Normalization under a Common Output Protocol. We apply the same clipping-based electron-number normalization to the raw 64^3 prediction of PI-EDG, ChargE3Net, 3D-ResNet-DFT, 3D U-Net, and GPWNO. Table 2 compares the spatial metrics before and after this operation. PI-EDG's MAE decreases from 3.472×10^{-3} to 3.405×10^{-3}, while ChargE3Net's changes from 3.419×10^{-3} to 3.599×10^{-3}. After common normalization, PI-EDG obtains an RMSE of 1.786×10^{-1} and a Pearson correlation of 0.691. These changes show that output normalization affects spatial errors differently across models, even though it fixes the density integral in every case. The common-normalization formula is provided in Appendix C.6.

Comparison across Extreme Scales. Global metrics combine vacuum, bonding, and core regions whose density magnitudes differ by many orders. Figure 3 reports MAE and RMSE within 37 ordered density bins, from the near-zero boundary bin B_0 to the high-density core bin B_{36}. Both panels use logarithmic vertical axes; the bin construction and metric definitions are given in Appendix C.4.

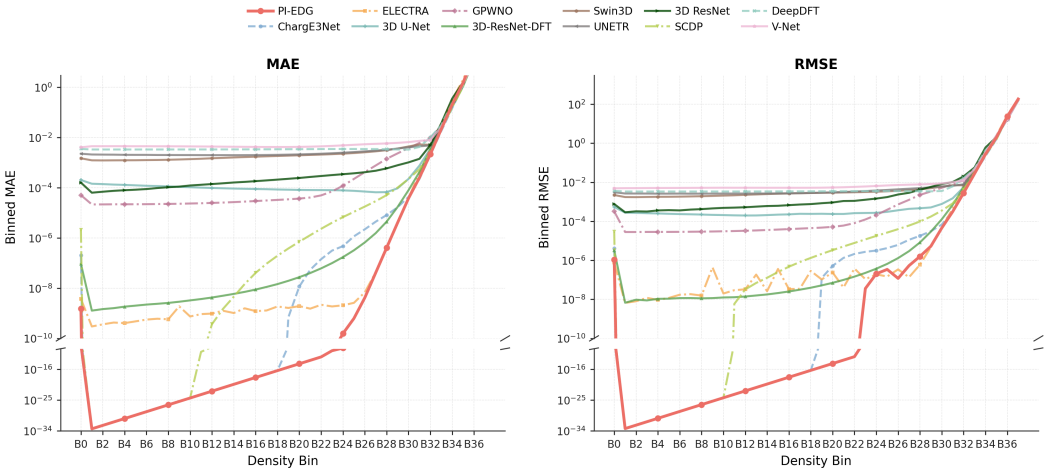
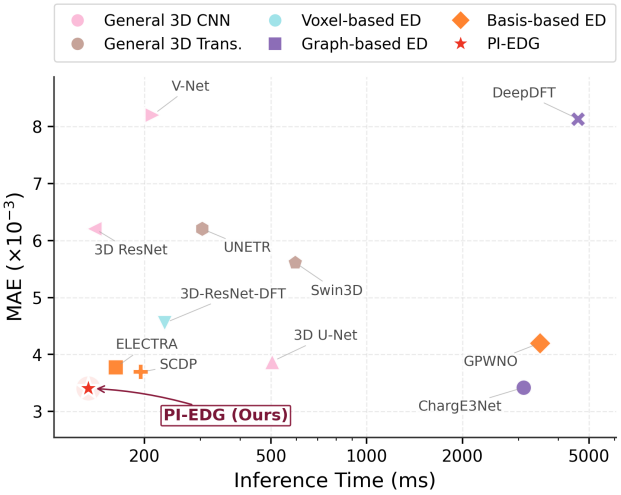


Figure 3: **Density-stratified MAE and RMSE.** Errors are computed from the final inference outputs in logarithmically ordered density bins. B_0 is the near-zero boundary bin, and B_{36} contains densities above 10^2.

PI-EDG traces the lowest error envelope across most near-zero and intermediate-density bins in both panels. The separation is largest through the vacuum-tail region, where several graph, voxel, and basis-based baselines retain errors many orders of magnitude above PI-EDG. ELECTRA, SCDP, ChargE3Net, and 3D-ResNet-DFT become more competitive as the density increases, while the general voxel regressors remain higher across much of the same range. Near B_{34}–B_{36}, all curves rise sharply and the gaps narrow as high-density core voxels dominate the absolute error. The matched ablation in Table 7 quantifies the contribution of density-range modeling: removing DS-LMoE raises near-zero LogMAE from 0.071 to 1.433 and high-density NMAE from 0.509 to 0.603. Appendix C.14 reports the corresponding voxel-wise absolute-error distributions, while Appendices D.1–D.3 provide projected densities, representative slices, and orthogonal spatial error maps.



Method	**Time ↓ (ms)**	**MAE ↓ $\times 10^{-3}$**
PI-EDG	**133.4**	**3.405**
3D ResNet	139.1	6.212
ELECTRA	162.5	3.771
SCDP	195.0	3.692
V-Net	212.4	8.203
3D-ResNet-DFT	231.2	4.550
UNETR	303.3	6.208
3D U-Net	503.5	3.874
Swin3D	595.2	5.615
GPWNO	3503.2	4.201
DeepDFT	4609.5	8.137
ChargE3Net	3109.3	3.419

Figure 4: **Performance–Inference time trade-off on EDBench.** Left: Comparison between inference time and MAE, where lower-left indicates better accuracy and efficiency. A logarithmic scale is used for the horizontal axis. Right: Quantitative results used to generate plot. PI-EDG achieves the fastest inference while maintaining competitive accuracy, offering a strong overall trade-off among general regression models and voxel-based, graph-based, and basis-based electron-density models.

Table 2: **Density errors before and after common output normalization, with downstream MEP errors.** "Raw" denotes the model output before electron-number normalization. MEP MAE is evaluated after common normalization at external probe points in Hartree/e. The best result in each column is bold.

Method	Raw output			Common normalization			MEP
	MAE $\times 10^{-3}$	RMSE $\times 10^{-1}$	Pearson	MAE $\times 10^{-3}$	RMSE $\times 10^{-1}$	Pearson	MAE $\times 10^{-2}$
PI-EDG	3.472	**1.802**	**0.684**	**3.405**	**1.786**	**0.691**	**5.300**
ChargE3Net	**3.419**	1.936	0.673	3.599	1.820	0.674	5.560
3D-ResNet-DFT	4.550	2.203	0.629	3.691	1.886	0.641	6.040
3D U-Net	3.874	1.816	0.679	3.708	1.817	0.689	5.606
GPWNO	4.201	1.900	0.677	4.256	1.798	0.681	8.486

3.3 Extended Experiments

Computational Efficiency and Performance Comparison. Fig. 4 compares MAE and the end-to-end time required to generate a complete 64^3 density grid. General 3D regressors have short inference paths, while coordinate-query and basis-based models include dense evaluation or reconstruction on the common output grid. PI-EDG obtains an MAE of 3.405×10^{-3} at an average end-to-end time of 133.4 ms per molecule. The measured components for each model family are listed in Appendix C.13.

Downstream MEP. We evaluate the external-region MEP on a fixed subset of 500 test molecules, using 32 approximately uniformly distributed probes around each molecule. The predicted densities are processed with the same clipping and electron-number normalization, and all MEP values are computed on the same grid with the same integration procedure. PI-EDG obtains 5.300×10^{-2} Hartree/e, compared with 5.560×10^{-2} for ChargE3Net; Table 2 reports all five methods. These values compare the density models under the same protocol. Appendix C.7 gives the full details.

Resolution Study. Separate 32^3, 64^3, and 128^3 models use the same simulation box and a common 64^3 evaluation target. Moving from 32^3 to 64^3 reduces MAE by 14.25%, with 4.39× the forward time and 2.44× the peak memory. From 64^3 to 128^3, MAE decreases by only 0.38%, while for-

Table 3: **Resolution study.**

Grid	Forward time	Peak memory	MAE ($\times 10^{-3}$)
32^3	15.2 ms	0.45 GB	3.971
64^3	66.7 ms	1.10 GB	3.405
128^3	492.8 ms	6.32 GB	**3.392**

ward time and peak memory increase by 7.39× and 5.75×. This diminishing return places 64^3 at a favorable accuracy–cost point for the tested molecule sizes and boxes. Table 3 reports the measurements, and Appendix C.12 gives the complexity analysis.

Reliability across Molecules and Subgroups. PI-EDG has lower MAE than ChargE3Net in every atom-count and element-composition group in Table 4, while both methods show higher error for larger molecules. The absolute gap grows from 2.04×10^{-4} for the small group to 1.93×10^{-4} for the largest-size subset. PI-EDG also retains lower MAE for both the common-element subset and the subset containing B, Si, P, S, Cl, or Br. Under common normalization, its molecule-level MAE is lower than ChargE3Net's on 92.4% of the test set. These comparisons characterize variation within EDBench; Appendices C.10 and C.11 provide the complete subgroup definitions and paired results.

Table 4: **Within-EDBench subgroup performance.**

Subset	Share	MAE ($\times 10^{-3}$) ↓	
		PI-EDG	ChargE3Net
Small (11–25 atoms)	35.50%	**2.882**	3.086
Medium (26–31 atoms)	34.95%	**3.548**	3.733
Large (32–48 atoms)	29.55%	**3.864**	4.057
Largest-size subset (36–48 atoms)	10.15%	**4.015**	4.112
Common elements only	78.35%	**3.514**	3.575
Rare/heavier elements	21.65%	**3.613**	3.687

Expert and Global-modeling Sensitivity. Increasing K from 1 to 8 reduces normalized MAE by 1.13×10^{-4}, whereas the change from $K = 8$ to $K = 16$ is 8.0×10^{-6}. At $K = 37$, learned routing improves MAE by 5.3×10^{-5} over fixed thresholds. In the matched-initialization comparison, the Hybrid FNO-CNN model obtains a raw MAE of 3.549×10^{-3}, compared with 3.630×10^{-3} for the CNN-only variant. Table 5 summarizes the routing results, and Appendix C.9 provides the paired protocols and single-head controls. The largest expert-count gain occurs between $K = 1$ and $K = 8$; subsequent increases produce smaller changes under the same normalized-MAE protocol.

Table 5: **Expert-count and routing sensitivity.** MAE uses common normalization.

Configuration	MAE ($\times 10^{-3}$) ↓
$K = 1$, cube-root single head	3.601
$K = 8$, learned routing	3.488
$K = 16$, learned routing	3.480
$K = 37$, fixed thresholds	3.458
$K = 37$, learned routing	**3.405**

Ablation Studies. Table 6 evaluates the components added to the 3D U-Net backbone using the metrics in Table 1. The output-normalization row reports the raw output of the trained model. The plain 3D U-Net obtains an MAE of 3.874, while the full model obtains 3.405. Removing physics priors increases the MAE to 3.561. Removing the FNO bottleneck changes RMSE from 1.786 to 1.861 and Pearson correlation from 0.691 to 0.673. Appendix C.9 compares the FNO-CNN model with a CNN-only variant using matched initialization and data order within each of the three paired runs. Removing learnable routing gives an MAE of 3.458, while removing the full DS-LMoE raises it to 3.631. The appendix also reports the expert-count and single-head controls. Before output normalization, the model obtains an MAE of 3.472, an RMSE of 1.802, and a Pearson correlation of 0.684; after normalization, the corresponding values are 3.405, 1.786, and 0.691.

Table 6: **Ablation study.** MAE and RMSE are reported in $\times 10^{-3}$ and $\times 10^{-1}$, respectively.

Model Variant	**MAE ↓**	**RMSE ↓**	**Pearson↑**
w/o Physics Priors	3.561	1.790	0.664
w/o FNO Bottleneck	3.634	1.861	0.673
w/o Learnable Routing	3.458	1.874	0.685
w/o DS-LMoE	3.631	1.819	0.681
w/o Output Normalization	3.472	1.802	0.684
only 3D U-Net	3.874	1.816	0.679
PI-EDG (Full)	**3.405**	**1.786**	**0.691**

Table 7 separates the global error from the two ends of the density range. Both single-head controls increase global MAE and near-zero LogMAE. The raw-space head remains closer to the full model in the high-density region, while its global MAE rises to 4.865×10^{-3}; the weighted cube-root head

Table 7: **Density-regime ablation.** Metrics use raw pre-normalization predictions from the matched-control experiment reported in Appendix C.9.

Configuration	Global MAE ($\times 10^{-3}$)	Near-zero LogMAE	High-density NMAE
Full PI-EDG	**3.549**	**0.071**	**0.509**
w/o DS-LMoE	3.697	1.433	0.603
Weighted single head	3.910	4.094	0.611
Raw-space single head	4.865	4.843	0.525

gives a smaller global error but a high-density NMAE of 0.611. The full DS-LMoE model records the lowest value for all three measures. Appendix C.8 defines the density ranges and metrics.

4 Limitations

PI-EDG uses a fixed voxel representation whose memory and runtime grow with the number of grid points. The current evaluation covers neutral EDBench molecules containing 11–48 atoms; charged systems, larger molecules, larger simulation boxes, and external chemical domains remain untested. The MEP experiment provides a relative comparison among density models, while application-specific accuracy thresholds remain to be established. Energy, force, self-consistent-field convergence, and shared machine-learning force-field endpoints remain for future evaluation. Density predictions used in critical applications require validation with established quantum-chemical calculations or experiments.

5 Conclusion

This work proposes the Physics-Informed Electron Density Generator (PI-EDG) to map molecular geometries directly to full-space electron-density grids without precomputed density-like inputs. First, PI-EDG introduces quantum-mechanical prior fields into a hybrid FNO-CNN backbone to model local density variations near nuclei and long-range dependencies across the grid. Moreover, DS-LMoE handles density values spanning many orders of magnitude, followed by inference-time normalization that matches the density integral to the neutral electron count. On EDBench, PI-EDG obtains an MAE of 3.405×10^{-3} under the original output protocol and the lowest MAE among the evaluated methods under common electron-number normalization. It also achieves the lowest external-region MEP MAE in the reported comparison and generates a complete 64^3 density grid in an average of 133.4 ms.

6 Broader Impact

Fast generation of full-space electron-density grids could make density-based analysis easier to incorporate into high-throughput molecular workflows. Predicted grids can provide inputs for molecular electrostatic potential evaluation, charge-distribution analysis, intermolecular interaction studies, and visual interpretation of electronic structure. They can also be combined with property predictors, molecular generators, and force-field models, giving these systems access to a spatial electronic descriptor alongside atomic coordinates. Beyond molecular electron density, PI-EDG provides a framework for learning physical fields. With suitable physical priors and training data, related designs could be explored for spin density, electrostatic potential, charge redistribution, and field prediction in periodic materials. Extensions to charged species, larger systems, and adaptive or multiresolution grids could further connect learned field generation with quantum chemistry, materials discovery, and multiscale simulation.

7 Acknowledgment

This work was supported by Yuelushan Laboratory Breeding Program (Grant No. YLS-2026-ZY01001), Fundamental and Interdisciplinary Disciplines Breakthrough Plan of the Ministry of Education of China (Grant No. JYB2025XDXM602), the National Natural Science Foundation of China (Grant No. 62425204; U22A2037; 62450002; 62432011; 625B2067), the National Natural Science Foundation of China (No.62502491), USTC School of Software Youth Innovation Fund Project (YN2260080011).

References

[1] P. Hapala, M. Švec, O. Stetsovych, N. J. van der Heijden, M. Ondráček, J. van der Lit, P. Mutombo, I. Swart, and P. Jelínek, "Mapping the electrostatic force field of single molecules from high-resolution scanning probe images," *Nature Communications*, vol. 7, p. 11560, May 2016.

[2] P. Hohenberg and W. Kohn, "Inhomogeneous electron gas," *Phys. Rev.*, vol. 136, pp. B864–B871, Nov 1964.

[3] N. Schuch and F. Verstraete, "Computational complexity of interacting electrons and fundamental limitations of density functional theory," *Nature Physics*, vol. 5, pp. 732–735, Oct. 2009.

[4] R. Pederson, B. Kalita, and K. Burke, "Machine learning and density functional theory," *Nature Reviews Physics*, vol. 4, pp. 357–358, June 2022.

[5] W. Gao, C. Addiego, H. Wang, X. Yan, Y. Hou, D. Ji, C. Heikes, Y. Zhang, L. Li, H. Huyan, T. Blum, T. Aoki, Y. Nie, D. G. Schlom, R. Wu, and X. Pan, "Real-space charge-density imaging with sub-ångström resolution by four-dimensional electron microscopy," *Nature*, vol. 575, pp. 480–484, Nov. 2019.

[6] H. Xiang, K. Li, M. Liu, Z. Cheng, B. Yao, W. Du, J. Xia, L. Zeng, X. Jin, *et al.*, "Edbench: Large-scale electron density data for molecular modeling," *Advances in Neural Information Processing Systems*, vol. 38, 2026.

[7] L. J. Bartolotti and K. Flurchick, *An Introduction to Density Functional Theory*, pp. 187–216. John Wiley Sons, Ltd, 1996.

[8] M. G. Medvedev, I. S. Bushmarinov, J. Sun, J. P. Perdew, and K. A. Lyssenko, "Density functional theory is straying from the path toward the exact functional," *Science*, vol. 355, no. 6320, pp. 49–52, 2017.

[9] P. Febrer, P. Jørgensen, M. Pruneda, A. Garcia, P. Ordejon, and A. Bhowmik, "Graph2Mat: Universal graph to matrix conversion for electron density prediction," Mar. 2025.

[10] P. B. Jørgensen and A. Bhowmik, "Equivariant graph neural networks for fast electron density estimation of molecules, liquids, and solids," *npj Computational Materials*, vol. 8, p. 183, Aug. 2022.

[11] T. Koker, K. Quigley, E. Taw, K. Tibbetts, and L. Li, "Higher-order equivariant neural networks for charge density prediction in materials," *npj Computational Materials*, vol. 10, p. 161, July 2024.

[12] I. Mitnikov and J. Jacobson, "E3STO: Orbital Inspired SE(3)-Equivariant Molecular Representation for Electron Density Prediction," Oct. 2024. ADS Bibcode: 2024arXiv241006119M.

[13] Z. Qiao, A. S. Christensen, M. Welborn, F. R. Manby, A. Anandkumar, and T. F. Miller, "Informing geometric deep learning with electronic interactions to accelerate quantum chemistry," *Proceedings of the National Academy of Sciences*, vol. 119, p. e2205221119, Aug. 2022.

[14] J. Elsborg, L. Thiede, A. Aspuru-Guzik, T. Vegge, and A. Bhowmik, "ELECTRA: A cartesian network for 3d charge density prediction with floating orbitals," in *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[15] X. Fu, A. Rosen, K. Bystrom, R. Wang, A. Musaelian, B. Kozinsky, T. Smidt, and T. Jaakkola, "A Recipe for Charge Density Prediction," May 2024.

[16] S. Kim and S. Ahn, "Gaussian Plane-Wave Neural Operator for Electron Density Estimation," June 2024. arXiv:2402.04278 [physics].

[17] Ö. Çiçek, A. Abdulkadir, S. S. Lienkamp, T. Brox, and O. Ronneberger, "3d u-net: Learning dense volumetric segmentation from sparse annotation," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2016* (S. Ourselin, L. Joskowicz, M. R. Sabuncu, G. Unal, and W. Wells, eds.), (Cham), pp. 424–432, Springer International Publishing, 2016.

[18] T. Wu, J. Wang, X. Pan, X. Xu, C. Theobalt, Z. Liu, and D. Lin, "Voxurf: Voxel-based Efficient and Accurate Neural Surface Reconstruction," Aug. 2023. arXiv:2208.12697 [cs].

[19] A. V. Sinitskiy and V. S. Pande, "Deep Neural Network Computes Electron Densities and Energies of a Large Set of Organic Molecules Faster than Density Functional Theory (DFT)," Sept. 2018. arXiv:1809.02723 [physics].

[20] C. Li, O. Sharir, S. Yuan, and G. K.-L. Chan, "Image super-resolution inspired electron density prediction," *Nature Communications*, vol. 16, p. 4811, May 2025.

[21] J. M. Parrilla-Gutiérrez, J. M. Granda, J.-F. Ayme, M. D. Bajczyk, L. Wilbraham, and L. Cronin, "Electron density-based GPT for optimization and suggestion of host–guest binders," *Nature Computational Science*, vol. 4, pp. 200–209, Mar. 2024.

[22] F. Milletari, N. Navab, and S.-A. Ahmadi, "V-Net: Fully Convolutional Neural Networks for Volumetric Medical Image Segmentation," June 2016.

[23] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.

[24] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi, "Photo-realistic single image super-resolution using a generative adversarial network," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 105–114, 2017.

[25] Y.-Q. Yang, Y.-X. Guo, J.-Y. Xiong, Y. Liu, H. Pan, P.-S. Wang, X. Tong, and B. Guo, "Swin3D: A Pretrained Transformer Backbone for 3D Indoor Scene Understanding," Aug. 2023. arXiv:2304.06906 [cs].

[26] A. Hatamizadeh, Y. Tang, V. Nath, D. Yang, A. Myronenko, B. Landman, H. Roth, and D. Xu, "UNETR: Transformers for 3D Medical Image Segmentation," Oct. 2021. arXiv:2103.10504 [eess].

[27] S. Wu, X. Yang, X. Zhao, Z. Li, M. Lu, X. Xie, and J. Yan, "Applications and advances in machine learning force fields," *Journal of Chemical Information and Modeling*, vol. 63, 09 2023.

[28] A. Kabylda, V. Vassilev-Galindo, S. Chmiela, I. Poltavsky, and A. Tkatchenko, "Efficient interatomic descriptors for accurate machine learning force fields of extended molecules," *Nature Communications*, vol. 14, p. 3562, June 2023.

[29] J. T. Frank, O. T. Unke, K.-R. Müller, and S. Chmiela, "A Euclidean transformer for fast and stable machine learned force fields," *Nature Communications*, vol. 15, p. 6539, Aug. 2024.

[30] T. Zheng, A. Wang, X. Han, Y. Xia, X. Xu, J. Zhan, Y. Liu, Y. Chen, Z. Wang, X. Wu, S. Gong, and W. Yan, "Data-driven parametrization of molecular mechanics force fields for expansive chemical space coverage," *Chemical Science*, vol. 16, pp. 2730–2740, Feb. 2025.

[31] V. G. Satorras, E. Hoogeboom, and M. Welling, "E(n) Equivariant Graph Neural Networks," in *Proceedings of the 38th International Conference on Machine Learning*, pp. 9323–9332, PMLR, July 2021.

[32] K. Schütt, H. E. Sauceda, P.-J. Kindermans, A. Tkatchenko, and K.-R. Müller, "SchNet – A deep learning architecture for molecules and materials," *The Journal of Chemical Physics*, vol. 148, p. 241722, June 2018.

[33] J. Gasteiger, S. Giri, J. T. Margraf, and S. Günnemann, "Fast and Uncertainty-Aware Directional Message Passing for Non-Equilibrium Molecules," Apr. 2022. arXiv:2011.14115 [cs].

[34] K. T. Schütt, O. T. Unke, and M. Gastegger, "Equivariant message passing for the prediction of tensorial properties and molecular spectra," June 2021. arXiv:2102.03150 [cs].

[35] J. Gasteiger, F. Becker, and S. Günnemann, "GemNet: Universal Directional Graph Neural Networks for Molecules," June 2024. arXiv:2106.08903 [physics].

[36] Y. Liu, L. Wang, M. Liu, X. Zhang, B. Oztekin, and S. Ji, "Spherical Message Passing for 3D Graph Networks," Nov. 2022. arXiv:2102.05013 [cs].

A Related Work

A.1 Machine Learning Force Fields

Machine learning force fields have substantially accelerated molecular simulation by learning energy and force functions from atomistic data [27–30]. Most existing models represent molecules as sparse atomic graphs and encode interactions through local neighborhoods, message passing, or equivariant geometric operations [31–36]. These designs are effective for many-body modeling and typically operate on atom-level representations. A full electron-density field provides a separate spatial representation of the electronic structure. PI-EDG directly predicts this voxelized field and can complement atom-level force-field models.

A.2 Electron Density Prediction

Electron density is the central variable in density functional theory and provides a direct description of the spatial distribution of electrons [2, 4]. Learning the mapping from molecular geometries to electron density can therefore accelerate density-related quantum-chemical workflows [19, 6]. Existing electron-density predictors differ mainly in their output representation. Coordinate-query methods evaluate density values at sampled spatial points [10, 11]; basis-based methods predict coefficients of predefined functions or orbital-inspired expansions [12, 14, 15]; voxel-based methods directly generate a density grid [20]. PI-EDG belongs to the voxel-based family and combines physical priors, non-local spectral modeling, magnitude-aware routing, and inference-time electron-number normalization in a direct voxel-generation pipeline.

A.3 Basis-based and Graph-based Electron Density Models

Graph-based methods such as DeepDFT and ChargE3Net use molecular graphs, equivariant message passing, or coordinate-query mechanisms to predict electron density [10, 11]. They benefit from structure-aware molecular encoders, but generating a full high-resolution density field requires dense coordinate queries over the entire space. This cost becomes significant when evaluating all voxels in a 64^3 grid or above.

Basis-based methods such as ELECTRA, SCDP, and E3STO predict density through atom-centered, Cartesian, or orbital-inspired basis expansions [14, 15, 12]. These methods inject useful physical structure, but they depend on predefined mathematical bases and require full-grid reconstruction after coefficient prediction. Such reconstruction often involves dense basis evaluation, spherical harmonics, orbital-related operations, or floating-orbital representations. GPWNO further explores function-space mapping with Gaussian and plane-wave representations, but full-grid reconstruction remains necessary for voxel-wise evaluation [16]. In contrast, PI-EDG directly produces the density grid in one dense forward pass and avoids repeated coordinate queries or explicit basis expansion.

A.4 3D Regression Methods for Physical Fields

3D regression architectures such as 3D U-Net, V-Net, 3D ResNet, Swin3D, and UNETR are widely used for volumetric prediction [17, 22, 23, 25, 26]. Their main advantage is dense parallel computation on regular grids. Electron-density fields span more than thirty orders of magnitude and contain sharp nuclear cusps. Recent voxel-based electron-density models establish the feasibility of grid-based density prediction and commonly use density-like intermediate inputs such as spherical neutral atomic density grids [19, 20]. PI-EDG combines dense voxel prediction with physical conditioning, non-local modeling, adaptive magnitude routing, and inference-time normalization to the known neutral electron count.

B Additional Methodological Details

B.1 Fourier-domain Derivation of the Hybrid FNO-CNN Bottleneck

Electron density contains both local singular structures and non-local electrostatic coupling. The local part appears near nuclei and chemical bonds, while the non-local part is reflected in Hartree-type

interactions. The Hartree energy is

$$E_H[\rho] = \frac{1}{2}\int \rho(\mathbf{r})V_H(\mathbf{r})d\mathbf{r}, \tag{17}$$

where V_H is the Hartree potential generated by the electron density.

The Hartree potential can also be written as a Coulomb convolution:

$$V_H(\mathbf{r}) = \int \frac{\rho(\mathbf{r}')}{\|\mathbf{r}-\mathbf{r}'\|}d\mathbf{r}'. \tag{18}$$

This expression is non-local: the potential at $\mathbf{r}$ depends on the density at every $\mathbf{r}'$. A direct dense evaluation over N grid points requires pairwise interactions and scales as $\mathcal{O}(N^2)$.

The same potential is equivalently obtained by solving the Poisson equation

$$\nabla^2 V_H(\mathbf{r}) = -4\pi\rho(\mathbf{r}). \tag{19}$$

Let $\mathcal{F}[\cdot]$ denote the Fourier transform and let $\boldsymbol{\xi}$ be a reciprocal-space wavevector. Using the identity

$$\mathcal{F}[\nabla^2 f](\boldsymbol{\xi}) = -\|\boldsymbol{\xi}\|^2\mathcal{F}[f](\boldsymbol{\xi}), \tag{20}$$

applying the Fourier transform to Eq. (19) gives

$$-\|\boldsymbol{\xi}\|^2 V_H(\boldsymbol{\xi}) = -4\pi\rho(\boldsymbol{\xi}). \tag{21}$$

Thus,

$$V_H(\boldsymbol{\xi}) = \frac{4\pi}{\|\boldsymbol{\xi}\|^2}\rho(\boldsymbol{\xi}), \qquad \boldsymbol{\xi} \neq \mathbf{0}. \tag{22}$$

The zero-frequency mode is handled by the boundary convention or by numerical stabilization. Eq. (22) shows that the non-local Hartree response becomes a pointwise spectral multiplication. This is the key motivation for using a Fourier branch: long-range electrostatic coupling has a compact representation in reciprocal space.

On a discrete voxel grid with $N = DHW$ points, let $\mathbf{x} \in \mathbb{R}^{C\times D\times H\times W}$ be a 3D feature tensor with C channels. A spectral operator layer first computes

$$\hat{\mathbf{x}} = \mathcal{F}(\mathbf{x}), \tag{23}$$

then applies a learnable complex-valued kernel on a truncated set of low-frequency modes $\mathcal{K}$:

$$\hat{\mathbf{y}}_{c_{\text{out}}}(\boldsymbol{\xi}) = \sum_{c_{\text{in}}=1}^{C} \mathbf{W}_{c_{\text{out}},c_{\text{in}}}(\boldsymbol{\xi})\hat{\mathbf{x}}_{c_{\text{in}}}(\boldsymbol{\xi}), \qquad \boldsymbol{\xi} \in \mathcal{K}. \tag{24}$$

Here, c_{in} and c_{out} index input and output channels, $\mathbf{W}$ contains the learned complex weights, and $\hat{\mathbf{y}}$ denotes the resulting Fourier coefficients. For modes outside $\mathcal{K}$, the coefficients are discarded or left unchanged according to the implementation. The output is transformed back to real space:

$$\mathbf{z}_{\text{fno}} = \mathcal{F}^{-1}(\hat{\mathbf{y}}). \tag{25}$$

The computational cost is dominated by FFT and inverse FFT with $\mathcal{O}(N \log N)$ scaling; dense pairwise interactions have $\mathcal{O}(N^2)$ scaling.

Eq. (24) learns a data-adaptive spectral kernel that approximates non-local response patterns in the latent feature space. The analogy to Eq. (22) motivates the design: global electrostatic effects can be efficiently represented through low-frequency spectral mixing.

However, truncating the spectrum can oversmooth high-frequency structures. Electron density has sharp local variations near nuclei, where high-frequency components are important. We therefore run a parallel 3D CNN branch:

$$\mathbf{z}_{\text{cnn}} = \text{Conv3D}(\mathbf{x}), \tag{26}$$

which preserves local geometric details. The FNO and CNN branches are fused as

$$\mathbf{z} = \phi\left(\text{Conv}_{1\times1\times1}\left[\mathbf{z}_{\text{fno}} \,\|\, \mathbf{z}_{\text{cnn}}\right]\right), \tag{27}$$

where $\|$ denotes channel concatenation, ϕ is a nonlinear activation, and $\mathbf{z}$ is the fused bottleneck feature passed to the decoder. This hybrid block combines global spectral mixing with local convolutional refinement. The FNO branch captures long-range dependencies, while the CNN branch preserves nuclear cusps and bonding-region sharpness.

B.2 Normalized Sigmoid-difference Routing

DS-LMoE computes density-range routing weights from the log-space guide field $g(\mathbf{r})$. The finite expert bounds are $\{t_i\}_{i=0}^{K}$, while the router uses separate sentinel boundaries

$$s_0 = -\infty, \qquad s_K = +\infty, \qquad s_k = t_k \text{ for } 1 \le k \le K-1. \tag{28}$$

For expert k, the routing weight is

$$\pi_k(\mathbf{r}) = \mathrm{sigmoid}\left(\tau(g(\mathbf{r}) - s_{k-1})\right) - \mathrm{sigmoid}\left(\tau(g(\mathbf{r}) - s_k)\right), \tag{29}$$

where τ controls boundary sharpness. The sentinel boundaries give

$$\begin{aligned} \sum_{k=1}^{K} \pi_k(\mathbf{r}) &= \sum_{k=1}^{K} [\mathrm{sigmoid}(\tau(g(\mathbf{r}) - s_{k-1})) - \mathrm{sigmoid}(\tau(g(\mathbf{r}) - s_k))] \\ &= \mathrm{sigmoid}(\tau(g(\mathbf{r}) - s_0)) - \mathrm{sigmoid}(\tau(g(\mathbf{r}) - s_K)) \\ &= 1. \end{aligned} \tag{30}$$

Thus, the routing weights form a normalized soft partition over the K experts.

B.3 Monotonic Threshold Parameterization

The K expert output intervals use the finite bounds $\{t_i\}_{i=0}^{K}$. We fix $t_0 = -34$ and parameterize the remaining bounds with unconstrained learnable increments α_j as

$$t_i = t_0 + \sum_{j=1}^{i} \exp(\alpha_j), \qquad i = 1, \ldots, K. \tag{31}$$

Since $\exp(\alpha_j) > 0$, the finite bounds remain strictly ordered:

$$t_0 < t_1 < \cdots < t_K. \tag{32}$$

We initialize $\alpha_j = 0$, which gives $t_i = t_0 + i$ and one decade per expert. The bounds are then optimized jointly with the density predictor.

B.4 Bounded Expert Remapping

Each expert predicts a unit-interval value:

$$y_k(\mathbf{r}) = \mathrm{sigmoid}(\mathrm{Expert}_k(\mathbf{h}(\mathbf{r}))) \in [0, 1]. \tag{33}$$

The finite log-density bounds are mapped to cube-root bounds:

$$a_k = 10^{t_{k-1}/3}, \qquad b_k = 10^{t_k/3}. \tag{34}$$

The expert output is

$$E_k(\mathbf{r}) = a_k + y_k(\mathbf{r})(b_k - a_k). \tag{35}$$

Therefore,

$$E_k(\mathbf{r}) \in [a_k, b_k]. \tag{36}$$

This remapping keeps each expert output within its assigned cube-root density interval.

B.5 Inference-time Neutral-electron Normalization

Let $\tilde{\rho}_i$ be the raw network prediction at voxel i. After the forward pass, negative numerical values are clipped:

$$\rho_i^{+} = \max(\tilde{\rho}_i, 0). \tag{37}$$

The final inference density is

$$\rho_i^{\mathrm{final}} = N_e \frac{\rho_i^{+}}{\sum_j \rho_j^{+} \Delta V + \epsilon_{\mathrm{num}}}, \tag{38}$$

where i and j index voxels, $N_e = \sum_A Z_A$ for the neutral molecule, ΔV is the voxel volume, and ϵ_{num} provides numerical stability. This operation is applied once after network inference and is excluded from the training computation graph. The controlled comparison in Appendix C.6 applies the same clipping-based formula to every evaluated method.

C Experimental Details and Additional Results

C.1 Dataset and Preprocessing

We evaluate all methods on EDBench. Each sample contains a molecular geometry and a DFT-computed electron-density field. The molecular geometry is represented by atomic coordinates $\mathbf{R} = \{\mathbf{R}_A\}_{A=1}^{M}$ and nuclear charges $\mathbf{Z} = \{Z_A\}_{A=1}^{M}$. The density field is discretized on a regular $64 \times 64 \times 64$ voxel grid for all models. All methods are evaluated under the same test split and output resolution.

For PI-EDG, the molecular input is converted into voxel-aligned physical fields. These include the Gaussian atom grid, the external Coulomb potential field, and the nuclear charge peak field. The target density is transformed into cube-root space for global regression and into $\log_{10}(\rho + \epsilon)$ with $\epsilon = 10^{-34}$ for guide supervision. For numerical integration, we denote the voxel volume as ΔV. The predicted total electron number is computed as

$$\hat{N}_e = \sum_i \hat{\rho}_i \Delta V. \tag{39}$$

C.2 Implementation Details

PI-EDG is implemented with Python 3.8.20, PyTorch 2.4.1+cu121, CUDA Toolkit 12.1, and cuDNN 9.1.0. The model builds on a physics-aware 3D encoder–decoder backbone with a U-Net-style topology. At each resolution level, the external potential and nuclear charge peak fields are injected through multi-scale physical fusion, enabling the network to incorporate quantum-mechanical priors throughout the hierarchy. At the bottleneck, a Hybrid FNO-CNN module combines a local convolutional branch with a Fourier branch, allowing the model to capture both high-frequency local structures and long-range electrostatic interactions.

On top of the backbone, PI-EDG employs a DS-LMoE output layer. Specifically, a guide-field predictor estimates the electron density in logarithmic space, a learnable log-space router produces soft routing weights, and the expert outputs are dynamically remapped to density intervals determined by the current routing thresholds. The reported configuration uses $K = 37$ experts, with the initial density intervals spanning one decade each. Unless otherwise specified, all main results use this setting. The model is trained with AdamW and a cosine annealing learning-rate schedule. An automatic learning-rate finder can optionally be applied before training in the single-GPU setting. The training loss is evaluated on the raw DS-LMoE prediction with the soft routing weights defined in Appendix B.2. At inference, the completed density grid is rescaled once to the neutral electron count. All experiments are conducted on four NVIDIA A800 80GB GPUs. Table 8 lists the full configuration used for the reported experiments.

C.3 Baseline Protocols

We compare PI-EDG with 11 baselines. All methods are evaluated on the same 64^3 output grid.

Graph-based models. DeepDFT and ChargE3Net predict density through graph or coordinate-query mechanisms. For full-space evaluation, their predictions are evaluated or reconstructed on the full voxel grid before computing metrics.

Basis-based models. ELECTRA, SCDP, and GPWNO predict density through basis-related, orbital-inspired, or operator-style representations. Their outputs are converted to the same voxel grid for evaluation. The cost of basis or grid reconstruction is included in inference-time reporting.

Voxel regression models. 3D U-Net, V-Net, 3D ResNet, Swin3D, UNETR, and 3D-ResNet-DFT are evaluated as dense voxel regression baselines. They share the same output resolution as PI-EDG. For baselines that require intermediate density-like inputs, the input construction time is included in the end-to-end inference time.

Table 8: **Hyperparameter configuration of PI-EDG.**

Hyperparameter	**Value**
Grid resolution	64^3
Backbone width n_{channels}	64
Number of experts K	37
Log-density floor ϵ	1×10^{-34}
Initial lower log-bound t_0	-34
Initial expert interval width	One decade
Optimizer	AdamW
Initial learning rate	1×10^{-5}
Weight decay	1×10^{-2}
Learning-rate schedule	Cosine annealing
Warm-up epochs	None
Batch size	8
Training epochs	30
FNO retained modes	4 per spatial axis
λ_{global}	1.0
λ_{guide}	1×10^{-5}
Threshold LR scale	0.1
Activation in residual blocks / bottleneck	SiLU
Expert output remapping	Sigmoid with dynamic interval remapping
Expert output activation	Sigmoid masking in cube-root space
Hardware	NVIDIA A800 80GB GPUs

C.4 Evaluation Metrics

Let M denote the number of test molecules and N_m the number of voxels for molecule m. The predicted and reference densities at voxel i are denoted by $\hat{\rho}_{m,i}$ and $\rho_{m,i}$, respectively. The global MAE and RMSE are computed over all voxels in the test set:

$$\text{MAE} = \frac{\sum_{m=1}^{M} \sum_{i=1}^{N_m} \left|\hat{\rho}_{m,i} - \rho_{m,i}\right|}{\sum_{m=1}^{M} N_m}, \tag{40}$$

and

$$\text{RMSE} = \sqrt{\frac{\sum_{m=1}^{M} \sum_{i=1}^{N_m} \left(\hat{\rho}_{m,i} - \rho_{m,i}\right)^2}{\sum_{m=1}^{M} N_m}}. \tag{41}$$

Pearson correlation is calculated separately for each molecule by flattening its predicted and reference density grids. For molecule m, it is defined as

$$r_m = \frac{\sum_{i=1}^{N_m} \left(\hat{\rho}_{m,i} - \bar{\hat{\rho}}_m\right)\left(\rho_{m,i} - \bar{\rho}_m\right)}{\sqrt{\sum_{i=1}^{N_m} \left(\hat{\rho}_{m,i} - \bar{\hat{\rho}}_m\right)^2} \sqrt{\sum_{i=1}^{N_m} \left(\rho_{m,i} - \bar{\rho}_m\right)^2}}, \tag{42}$$

where

$$\bar{\hat{\rho}}_m = \frac{1}{N_m} \sum_{i=1}^{N_m} \hat{\rho}_{m,i}, \qquad \bar{\rho}_m = \frac{1}{N_m} \sum_{i=1}^{N_m} \rho_{m,i}. \tag{43}$$

The reported Pearson correlation is the average over the test molecules:

$$\text{Pearson} = \frac{1}{M} \sum_{m=1}^{M} r_m. \tag{44}$$

To evaluate errors across the density range, we assign each test-set voxel to one of the 37 ordered bins according to its reference density. The bins are denoted by $B_0, \ldots, B_{36}$: B_0 is the near-zero boundary bin, B_{36} contains reference densities above 10^2, and the intermediate bins follow the logarithmic ordering used in Fig. 3. The density-stratified MAE for bin B_k is

$$\text{MAE}_k = \frac{1}{|B_k|} \sum_{(m,i) \in B_k} \left|\hat{\rho}_{m,i} - \rho_{m,i}\right|. \tag{45}$$

Thus, each bin aggregates all test-set voxels whose reference density falls within the corresponding interval.

For the raw pre-normalization density-regime analysis, we define

$$\mathcal{B}_{\text{low}} = \left\{(m,i) \mid 0 \le \rho_{m,i} < 10^{-4}\right\} \tag{46}$$

and report

$$\text{LogMAE} = \frac{1}{|\mathcal{B}_{\text{low}}|} \sum_{(m,i)\in\mathcal{B}_{\text{low}}} \left|\log_{10}(\hat{\rho}_{m,i}+\epsilon) - \log_{10}(\rho_{m,i}+\epsilon)\right|. \tag{47}$$

For high-density voxels, we define

$$\mathcal{B}_{\text{high}} = \{(m,i) \mid \rho_{m,i} \ge 10\} \tag{48}$$

and report

$$\text{NMAE} = \frac{\sum_{(m,i)\in\mathcal{B}_{\text{high}}} |\hat{\rho}_{m,i} - \rho_{m,i}|}{\sum_{(m,i)\in\mathcal{B}_{\text{high}}} |\rho_{m,i}| + \epsilon}, \tag{49}$$

where ϵ is the numerical constant used in the log-space density transform.

C.5 Statistical Reporting

All experiments are repeated in three independent runs with different random seeds, using the same train/validation/test split and voxel resolution. Table 1 reports the mean and standard deviation. Tables that do not display a standard deviation report the mean over the three runs. Timing experiments report the average end-to-end inference time per molecule on the same hardware.

C.6 Common-normalization Protocol and Results

Table 1 follows the original output protocol of each method. PI-EDG's original protocol applies neutral-electron normalization after the network forward pass. For the controlled comparison, let $\tilde{\rho}_i$ denote a method's raw full-grid prediction. We clip negative values and rescale the full grid to the neutral electron count:

$$\hat{\rho}_i^{\text{common}} = N_e \frac{\max(\tilde{\rho}_i, 0)}{\sum_j \max(\tilde{\rho}_j, 0)\Delta V + \epsilon_{\text{num}}}. \tag{50}$$

This operation is applied to every evaluated method on the same 64^3 grid after the model forward pass. PI-EDG's optimization objective is evaluated on the raw DS-LMoE output, and electron-number normalization begins after inference. Table 9 reports the density metrics under this shared protocol.

Table 9: **Comparison under common clipping-based electron-number normalization.** The normalization fixes the density integral to the neutral electron count; the table reports spatial density metrics.

Method	MAE ($\times 10^{-3}$) ↓	RMSE ($\times 10^{-1}$) ↓	Pearson ↑
PI-EDG	**3.405**	**1.786**	**0.691**
ChargE3Net	3.599	1.820	0.674
3D-ResNet-DFT	3.691	1.886	0.641
3D U-Net	3.708	1.817	0.689
GPWNO	4.256	1.798	0.681

C.7 External-region MEP

For a probe position $\mathbf{r}_p$, the MEP combines the nuclear and electronic contributions:

$$V(\mathbf{r}_p) = \sum_A \frac{Z_A}{\|\mathbf{r}_p - \mathbf{R}_A\|} - \sum_i \frac{\rho_i \Delta V}{\|\mathbf{r}_p - \mathbf{r}_i\|}. \tag{51}$$

We use a fixed subset of 500 test molecules. Each molecule is evaluated at 32 approximately uniformly distributed probes on a sphere centered at the molecular centroid. Its radius is $R_{\max} + 2$ Å, where

$R_{\max}$ is the largest distance from the centroid to any atom in the molecule. Every probe is therefore at least 2 Å from each nucleus. Predicted and DFT reference densities use the same grid, are clipped at zero, and are normalized to the neutral electron count before integration. Coordinates and the voxel-cell volume are converted to atomic units. We first average the absolute error over the 32 probes of each molecule and then average over molecules. Table 10 reports the resulting MAE in Hartree/e.

Table 10: **MEP MAE in the external molecular region.**

Method	MEP MAE ($\times 10^{-2}$ Hartree/e) $\downarrow$
PI-EDG	**5.300**
ChargE3Net	5.560
3D U-Net	5.606
3D-ResNet-DFT	6.040
GPWNO	8.486

C.8 Density-regime Ablation

All metrics in Table 7 are computed from the same raw pre-normalization predictions. The near-zero region contains voxels with $0 \leq \rho < 10^{-4}$, and the high-density region contains voxels with $\rho \geq 10$. LogMAE and NMAE follow the definitions in Appendix C.4.

C.9 Architectural Sensitivity

Table 11 varies the number of experts and the routing thresholds under the common neutral-electron normalization used in the main text. The $K = 1$ setting uses a cube-root single head.

Table 11: **Expert-count and routing sensitivity.**

Configuration	Common-normalized MAE $\downarrow$ ($\times 10^{-3}$)
$K = 1$, cube-root single head	3.601
$K = 8$, learned routing	3.488
$K = 16$, learned routing	3.480
$K = 37$, fixed-threshold routing	3.458
$K = 37$, learned routing	**3.405**

Table 12 reports a separate matched-control experiment comparing the hybrid bottleneck with a CNN-only variant. Within each of the three paired runs, the two models use the same data split, training budget, data order, and initialization for shared layers. DFT-grid normalization uses the integrated electron count of the reference density on the evaluation grid. Nominal-electron normalization uses $N_e = \sum_A Z_A$ for the neutral molecule.

Table 12: **Paired Hybrid FNO-CNN and CNN-only comparison.**

Output protocol	Hybrid FNO-CNN MAE ($\times 10^{-3}$) $\downarrow$	CNN-only MAE ($\times 10^{-3}$) $\downarrow$
Raw output	**3.549**	3.630
DFT-grid normalized	**3.405**	3.450
Nominal-electron normalized	**3.405**	3.502

C.10 Within-EDBench Subgroup Performance

The test set covers H, B, C, N, O, F, Si, P, S, Cl, and Br. Molecules containing only H/C/N/O/F account for 78.35% of the test set; 21.65% contain at least one of B/Si/P/S/Cl/Br. Table 13 reports groups defined within EDBench.

Errors increase with atom count for both methods. These groups characterize performance within the molecule sizes and element combinations represented in EDBench.

Table 13: **MAE across EDBench molecule-size and element subgroups.**

Subset	Test share	MAE ($\times 10^{-3}$) ↓	
		PI-EDG	ChargE3Net
Small (11–25 atoms)	35.50%	**2.882**	3.086
Medium (26–31 atoms)	34.95%	**3.548**	3.733
Large (32–48 atoms)	29.55%	**3.864**	4.057
Largest-size subset (36–48 atoms)	10.15%	**4.015**	4.112
Common elements only	78.35%	**3.514**	3.575
Rare/heavier elements included	21.65%	**3.613**	3.687

C.11 Paired Molecule-level MAE

For each baseline, Table 14 reports PI-EDG minus baseline aggregate normalized MAE and the percentage of test molecules on which PI-EDG has lower MAE. The win rate is a descriptive paired statistic.

Table 14: **Paired molecule-level normalized-MAE comparison.**

Baseline	MAE difference ($\times 10^{-3}$) ↓	PI-EDG win rate ↑
ChargE3Net	-0.194	92.4%
GPWNO	-0.851	100.0%
3D-ResNet-DFT	-0.286	82.2%
3D U-Net	-0.303	95.4%

C.12 Resolution and Complexity

Let $V = DHW$ be the number of voxels. Convolutional operations scale linearly with V for fixed kernel sizes and channel widths, while the FFT in the FNO branch has time complexity $O(V \log V)$. Doubling all three spatial dimensions increases V by a factor of eight. The three models in Table 15 are trained separately over the same simulation box. Their MAEs follow the common 64^3 target protocol described in the main text.

Table 15: **Accuracy, forward time, and peak allocated memory across grid resolutions.**

Resolution	Voxels	Forward time ↓	Memory ↓	MAE ($\times 10^{-3}$) ↓
32^3	32,768	15.2 ms	0.45 GB	3.971
64^3	262,144	66.7 ms	1.10 GB	3.405
128^3	2,097,152	492.8 ms	6.32 GB	**3.392**

C.13 Computational Cost and Inference Time Decomposition

Our timing protocol measures end-to-end full-grid generation time for the complete 64^3 electron-density field used in voxel-wise evaluation. For coordinate-query and basis-based methods, the reported time includes dense grid evaluation or reconstruction. Inference time is measured from molecular geometries to the final electron-density grid. For each method, the reported time includes all steps needed to produce the density cube used for evaluation. Table 16 summarizes the components included for each method family.

For PI-EDG,

$$T_{\text{total}}^{\text{PI-EDG}} = T_{\text{phys_grid}} + T_{\text{network}} + T_{\text{output_norm}}, \tag{52}$$

where $T_{\text{phys_grid}}$ includes atom-grid construction, external-potential construction, and nuclear-peak construction. T_{network} includes the 3D backbone, Hybrid FNO-CNN bottleneck, and DS-LMoE prediction. $T_{\text{output_norm}}$ is the inference-time neutral-electron normalization.

For general voxel regressors,

$$T_{\text{total}}^{\text{voxel}} = T_{\text{grid}} + T_{\text{forward}}. \tag{53}$$

This applies to 3D U-Net, V-Net, 3D ResNet, Swin3D, and UNETR.

Table 16: **End-to-end inference-time decomposition.** The measured time includes all operations required to map a molecular geometry to a full electron-density grid.

Method family	Methods	End-to-end time components
PI-EDG	PI-EDG	$T_{\text{phys_grid}} + T_{\text{network}} + T_{\text{output_norm}}$
General voxel regression	3D U-Net, V-Net, 3D ResNet, Swin3D, UNETR	$T_{\text{grid}} + T_{\text{forward}}$
Voxel ED baseline	3D-ResNet-DFT	$T_{\text{SAD/input}} + T_{\text{forward}}$
Graph-based ED	DeepDFT, ChargE3Net	$T_{\text{graph}} + T_{\text{query}} + T_{\text{grid_eval}}$
Basis-based ED	ELECTRA, SCDP	$T_{\text{basis_input}} + T_{\text{coef}} + T_{\text{basis_eval}}$
Operator-style ED	GPWNO	$T_{\text{operator_input}} + T_{\text{operator}} + T_{\text{grid_recon}}$

For 3D-ResNet-DFT,

$$T_{\text{total}}^{\text{3D-ResNet-DFT}} = T_{\text{SAD/input}} + T_{\text{forward}}, \tag{54}$$

where $T_{\text{SAD/input}}$ denotes construction of the intermediate density-like input.

For graph-based models,

$$T_{\text{total}}^{\text{graph}} = T_{\text{graph}} + T_{\text{query}} + T_{\text{grid_eval}}. \tag{55}$$

Here, T_{graph} is molecular graph construction, T_{query} is coordinate-wise density evaluation, and $T_{\text{grid_eval}}$ is dense evaluation on the full voxel grid.

For basis-based models,

$$T_{\text{total}}^{\text{basis}} = T_{\text{basis_input}} + T_{\text{coef}} + T_{\text{basis_eval}}. \tag{56}$$

Here, $T_{\text{basis_input}}$ is basis-related input construction, T_{coef} is coefficient prediction, and $T_{\text{basis_eval}}$ is full-grid density reconstruction.

For GPWNO,

$$T_{\text{total}}^{\text{GPWNO}} = T_{\text{operator_input}} + T_{\text{operator}} + T_{\text{grid_recon}}. \tag{57}$$

C.14 Absolute-error Distributions

Fig. 5 reports the voxel-wise absolute-error probability distributions of the final inference outputs from PI-EDG and all evaluated baselines. The distributions show that many baselines exhibit broad or shifted high-error modes, whereas PI-EDG concentrates more probability mass in the low-error region and suppresses pronounced high-error tails. Each panel shows the binned probability of absolute prediction errors for one method, together with a smoothed trend curve. These distributions reveal whether errors are concentrated in the low-error region or dispersed into broad high-error tails.

D Additional Visualizations

The PI-EDG visualizations use the final density produced after inference-time neutral-electron normalization.

D.1 Projected Full-space Density Visualizations

Fig. 6 presents projected full-space electron-density visualizations for representative test molecules. Each row shows the molecular geometry, the projected ground-truth density, the PI-EDG prediction, and the corresponding absolute error. The molecular geometries are rendered in a ball-and-stick style, while the density and error fields are displayed in logarithmic scale to reveal both high-density core regions and low-density tails. These visualizations show that PI-EDG recovers the overall spatial support and molecular-scale density morphology while keeping the major errors localized within the molecular density region.

D.2 Density Slices

Fig. 7 reports representative two-dimensional density slices for the same test molecules. For each sample, we compare the ground-truth density slice, the PI-EDG prediction, and the absolute-error map on the selected plane. The results show that PI-EDG captures the dominant density support and the main high-density regions, while the absolute-error maps highlight residual local deviations around nuclear and bonding regions.

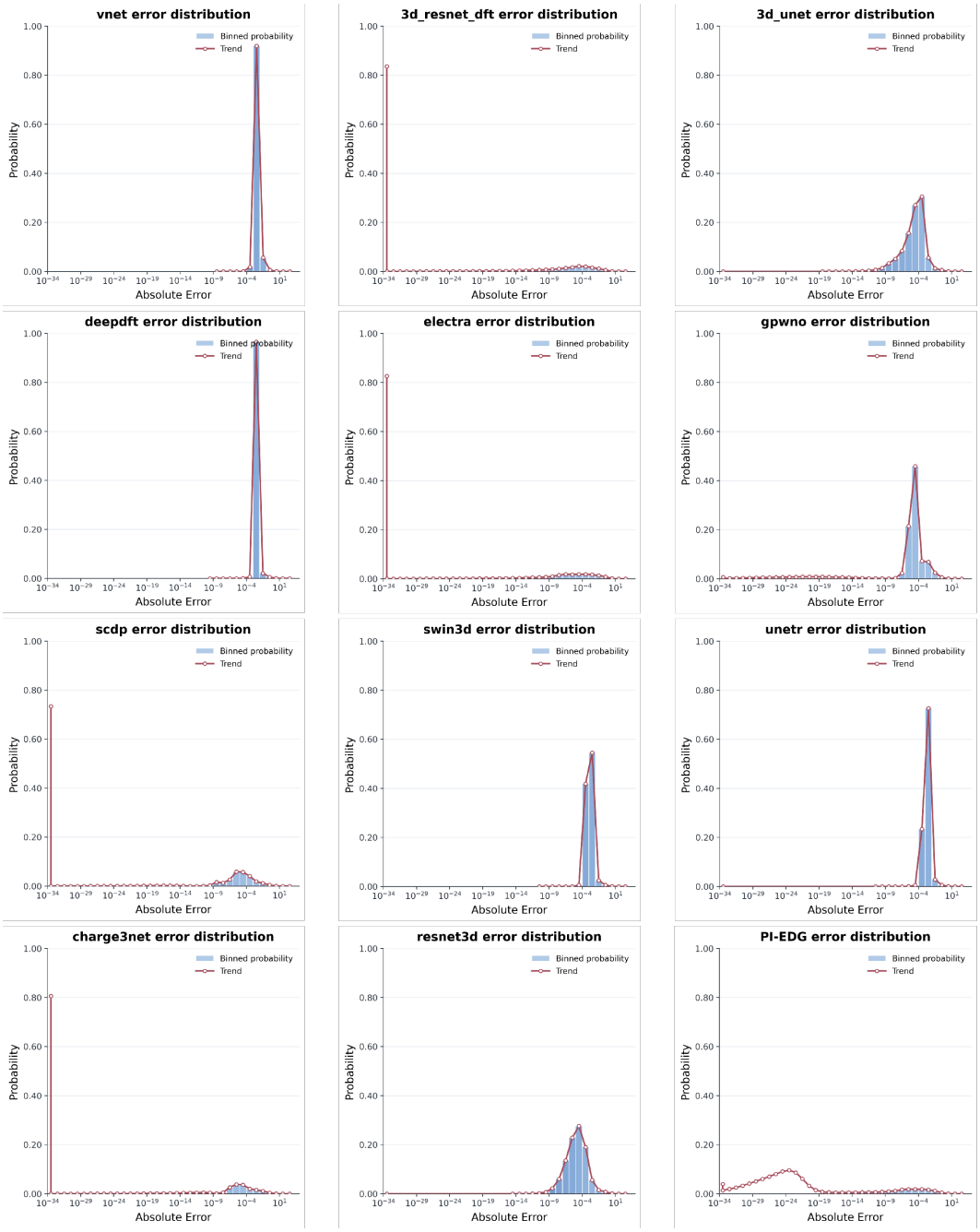


Figure 5: **Absolute-error probability distributions of PI-EDG and all evaluated baselines.** Broad tails or secondary high-error modes indicate a higher probability of local density failures, while a distribution concentrated in the low-error region reflects more stable voxel-wise prediction.

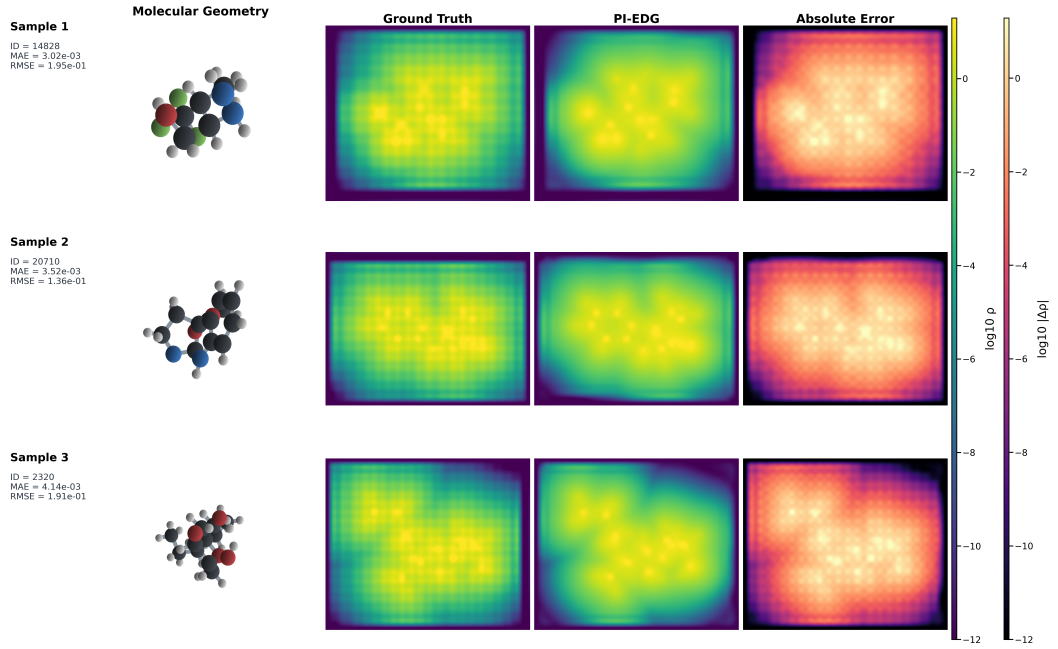


Figure 6: **Projected full-space electron-density visualization.** Representative examples comparing molecular geometries, projected ground-truth density, PI-EDG prediction, and absolute error. Density and error values are shown in logarithmic scale for better visualization across the broad dynamic range of electron density.

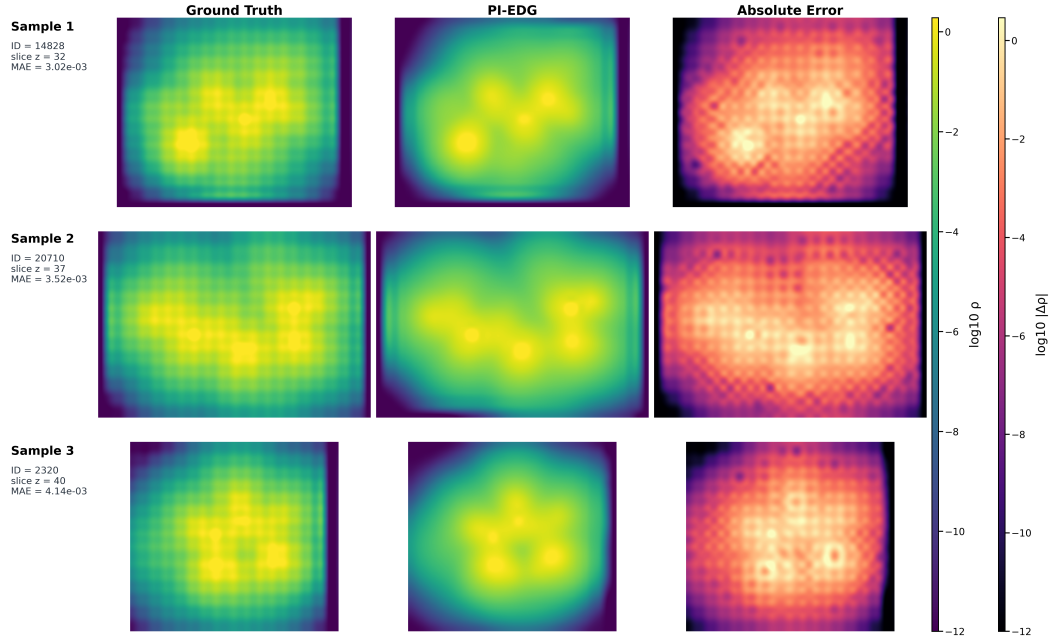


Figure 7: **Representative density slices.** Each row shows the selected two-dimensional slice of the ground-truth density, the PI-EDG prediction, and the corresponding absolute error for one test molecule. Density and error maps are visualized in logarithmic scale.

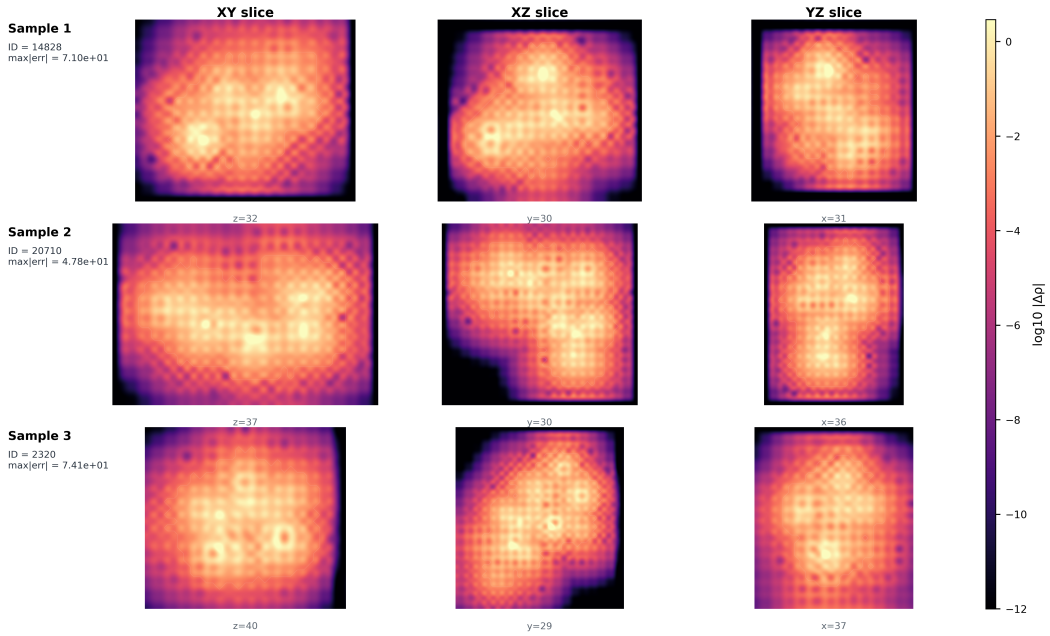


Figure 8: **Spatial error maps.** Absolute-error slices along the XY, XZ, and YZ planes for representative test molecules. The maps illustrate the spatial localization of prediction errors and complement the quantitative error analyses.

D.3 Spatial Error Maps

Fig. 8 further visualizes the spatial distribution of prediction errors along three orthogonal planes. For each representative molecule, we show absolute-error slices in the XY, XZ, and YZ directions. These maps complement the density-stratified MAE curves and absolute-error distributions by identifying where local deviations occur in real space. The errors are mainly concentrated within the molecular density support and near structurally dense regions, with smaller errors across the remaining voxel grid.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state the main goal, method components, and empirical claims of PI-EDG. The claims are supported by the reported quantitative comparisons, ablation studies, and qualitative visualizations.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper includes a dedicated Limitations section discussing fixed-grid resolution and scaling to larger molecules or higher-resolution settings.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: The paper does not present formal theorems, but provides mathematical formulations and methodological derivations for the proposed modules. Additional derivations for the Fourier-domain bottleneck, routing, remapping, and normalization are provided in Appendix B.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: The paper specifies the dataset, baselines, evaluation metrics, training settings, and hyperparameters used for the main experiments. Additional implementation details, baseline protocols, and metric definitions are provided in Appendix C.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in

some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The experiments are based on the public EDBench dataset and standard baseline implementations. We provide an anonymized code repository in the abstract.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: The main experimental settings are described, including the dataset, baselines, metrics, optimizer, loss weights, and hardware. Full hyperparameter settings and implementation details are further reported in Appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Quantitative results are reported as mean $\pm$ standard deviation over three independent runs with different random seeds. The statistical reporting protocol is described in Appendix C.5.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.

- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The paper reports the GPU type used for the experiments and provides end-to-end inference-time measurements. Appendix further decomposes the inference-time components for different model families.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The work uses molecular simulation data and does not involve human subjects, private information, or safety-sensitive personal data. The study is conducted as a computational modeling work and follows standard research ethics for model evaluation and reporting.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: The Broader Impact section discusses potential benefits, including reducing the cost of quantum-chemical workflows and supporting downstream molecular modeling. It also notes possible risks from inaccurate density predictions and recommends validation with established quantum-chemical calculations or experiments.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: The paper does not release high-risk models, scraped datasets, or generative systems with direct misuse potential. Therefore, no additional safeguard mechanism is required beyond standard validation and responsible scientific use.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: The paper cites the datasets, baseline models, and prior methods used in the experiments.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [N/A]

Justification: The paper does not introduce a new dataset, benchmark, or externally released asset. The contribution is a model architecture and empirical evaluation built on existing datasets and baselines.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: The work does not involve crowdsourcing, user studies, or experiments with human participants. No participant instructions, compensation details, or human-subject protocols are therefore applicable.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The work does not involve human subjects or any intervention with human participants. IRB approval or equivalent human-subject review is therefore not applicable.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: The core methodology does not use LLMs as a model component, training signal, or experimental baseline. Any language editing or formatting assistance does not affect the scientific method, results, or originality of the work.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.