CCF Transactions on High Performance Computing (2025) 7:85–99
https://doi.org/10.1007/s42514-025-00219-0

REGULAR PAPER

Star-gen: an HPC-AI framework for constructing large-scale computational materials database

Pin Chen[1] · **Qing Mo**[1] · **Zexin Xu**[1] · **Xianwei Zhang**[1] · **Yutong Lu**[1]

Received: 9 August 2024 / Accepted: 24 February 2025 / Published online: 7 April 2025

Abstract
Constructing large-scale, high-quality computational materials databases is pivotal for advancing material simulation and design. However, two essential challenges are yet to be fully resolved in this field: acquiring comprehensive atomic-level structural information and effectively executing large-scale computational tasks for these material structures on supercomputers. In this study, we present a methodology that adeptly couples Artificial Intelligence (AI) and High-Performance Computing (HPC) to establish a comprehensive computational database with diverse materials data. We propose an AI-driven pipeline with a periodic-E(3)-equivariant diffusion model for structure generation and a transformer-based property prediction model incorporating 3D geometric analysis for material structure evaluation, followed by calculations on selected structures using Density Functional Theory (DFT). Specifically, a high-throughput computing framework was developed for efficient execution of various CPU/GPU/IO-intensive tasks, capitalizing on the heterogeneous computing nodes and shared storage architecture of supercomputers. Based on our HPC-AI strategy, we generated approximately 10 million hypothetical crystal structures, constituting the most extensive crystal material database currently available. By leveraging 2,000 nodes of the Tianhe-2 supercomputer for high-throughput computations, we accomplished about 80,000 DFT calculation datasets within a span of three months. Our approach represents a data-driven paradigm for boosting the materials design practice.

Keywords HPC-AI framework · Computational materials · High-throughput computing · Large-scale materials generation

1 Introduction

Artificial intelligence (AI) plays a transformative role in scientific research, particularly due to its proficiency in efficiently processing high-dimensional data, thereby markedly reducing computational times. In the field of materials science, the applications of AI are extensive, encompassing tasks such as property prediction (Reiser et al. 2022), structure generation (Yan et al. 2023), and reaction forecasting (Zhang et al. 2022), thus establishing it as an indispensable tool. However, AI models are frequently critiqued for their heavy reliance on data, potentially limiting their generalizability. Recently, the focus has shifted towards leveraging computational methods, particularly those grounded in quantum mechanics (QM), to enhance AI models (Singh et al. 2021; Marzari et al. 2021). This strategy is crucial for elucidating structure–property relationships in materials through the creation of a large-scale computational database. The use of high-performance computing (HPC) for data acquisition significantly bolsters the accuracy and generalizability of AI models, offering a more comprehensive and effective methodology for the exploration of materials science.

Efficiently constructing large-scale computational databases on supercomputers presents a significant systemic engineering challenge, particularly within the integrated HPC-AI framework that accommodates diverse and

✉ Yutong Lu
luyutong@mail.sysu.edu.cn

Pin Chen
chenp85@mail.sysu.edu.cn

Qing Mo
qing.mo@nscc-gz.cn

Zexin Xu
zexin.xu@nscc-gz.cn

Xianwei Zhang
zhangxw79@mail.sysu.edu.cn

[1] National Supercomputer Center in Guangzhou, School of Computer Science and Engineering, Sun Yat-sen University, 132 East Circle at University City, Guangzhou 510006, People's Republic of China

intensive tasks involving CPU, GPU, and IO (Reed et al. 2022). In the context of materials science, it is imperative to develop methods that facilitate effective structural acquisition and rapid evaluation. Although AI has made significant strides in the materials domain, offering alternatives to traditional, resource-intensive methods like crystal structure generation, traditional computational techniques continue to hold irreplaceable advantages in addressing AI's limitations regarding generalization and accuracy (Zhang and Limei Wang 2023). Therefore, establishing an effective HPC-AI comping framework becomes essential to overcome the challenges associated with building large-scale computational databases in materials science.

In this study, we propose a novel framework that couples HPC and AI to establish an extensive computational materials database with diverse materials data. Specially, we propose an AI-driven pipeline by integrating a periodic-E(3)-equivariant diffusion model, trained on a dataset of 2 million crystalline material structures, to generate material structures defined by specific elemental compositions and atomic quantities. Additionally, we develop a transformer-based property prediction model that integrates 3D geometric analysis for structure recommendation. Selected materials candidates underwent rigorous high-throughput density functional theory (DFT) calculations on HPC clusters, facilitating the extraction of their comprehensive physicochemical properties for seamless integration into the database. In order to capitalize on the heterogeneous compute nodes and shared storage architecture of supercomputers, we develop a high-throughput computing framework for efficient execution of various CPU/GPU/IO-intensive tasks. Employing our HPC-AI strategy, we constructed approximately 10 million hypothetical crystal structures, constituting the most extensive crystal material database currently available. By leveraging 2,000 nodes of the Tianhe-2 supercomputer for high-throughput computations, we accomplished about 80,000 DFT calculation datasets within a span of three months. To better manage, analyze, and share data, we employ geographically distributed platforms for the deployment and operation of our HPC-AI framework.

Our contributions are summarized as follows:

- We propose an AI-driven, pre-trained modeling pipeline adept at generating novel material structures and identifying candidates for subsequent DFT analysis.
- We develop HydraHT, a job execution framework tailored for augmenting high-throughput computing (HTC) applications, which facilitates the efficient processing of diverse CPU/GPU/IO-intensive tasks across the heterogeneous compute nodes and shared storage systems of supercomputers.
- Based on our proposed method, we construct a database with around 10 million hypothetical crystal structures and perform DFT calculations for a subset of around 80,000 materials on Tianhe-2 supercomputer.

The rest of this paper is organized as follows: we give the necessary background knowledge of computational materials science and the motivation of this paper in Sect. 2. Section 3 describes the preliminaries of this work. Section 4 gives the details of HPC-AI framework. Section 5 presents the evaluation methodology and experimental details for our framework. Section 6 discusses the framework of Star-gen. Section 7 concludes the paper and gives an outlook.

2 Background and motivation

2.1 Computational materials database

The Material Project (MP) encompasses over 154,000 inorganic materials, with around 50,000 being experimental materials derived from experimental databases, including ICSD codes, indicating the presence of approximately 100,000 novel structures (Jain et al. 2013). The OQMD database houses roughly 1 million calculated structures, featuring 30,000 from ICSD, while the remaining novel structures are algorithmically constructed (Kirklin et al. 2015). JARVIS aggregates structures from MP, including about 65,000 structures precisely calculated using high-precision DFT (Choudhary et al. 2018). These databases serve as key resources for AI model training in property prediction and structure generation. However, these databases are limited in scope, with their structural count capped at the million level. Additionally, they exhibit notable biases due to their sources of structure, such as the underrepresentation of elements like actinides and noble gases in MP and OQMD, and a predilection for smaller structures, typically those with fewer than 80 atoms (Chen et al. 2022). Our approach diverges by originating directly from the source of material structures, generating and performing DFT calculations on them to create an unbiased dataset.

2.2 AI for materials science

AI techniques, popularly applied in the realms of small molecules and proteins, have also been extended to the modeling of crystalline materials. These materials, characterized by periodic atomic repetitions in 3D space, are crucial in various industrial applications, including semiconductor electronics, solar cells, and batteries (Butler et al. 2018). The burgeoning industrial demand has propelled materials science into a realm of extensive fundamental research, focusing on predicting material properties, such as formation energy, and designing novel materials with specific target properties.

Current models for predicting material properties, such as CGCNN (Xie and Grossman 2018), MEGNET (Chen et al. 2019), CrystalNet (Chen et al. 2022), ALIGNN (Choudhary and DeCost 2021), Matformer (Yan et al. 2022), utilizes multi-edge and fully connected graphs to encapsulate periodic information. While these models achieve DFT-level computational accuracy, they still underutilize the geometric structural information inherent in crystal structures. Conversely, AI-based advancements in material crystal structure acquisition have been significant, with models like CDVAE (Xie et al. 2022) and DiffCSP (Jiao et al. 2024) capable of directly predicting the structure of crystals based on specified properties or chemical compositions. In this study, we employ a novel property prediction model that incorporates vector edges between encoded atoms to enhance the representation of spatial geometric information in crystals. Additionally, we integrate the cutting-edge DiffCSP crystal structure generation model with our proprietary property prediction model to establish a pipeline for generating potential materials.

2.3 High throughput computing

The construction of large-scale materials computational databases critically depends on the efficiency of HTC, necessitating robust and versatile computational tools and frameworks. HTCondor, an open-source framework for HTC, offers flexible resource configuration but encounters performance bottlenecks in large-scale concurrent tasks due to its serial process initiation and single-task management per process (Fajardo et al. 2015). Slurm, noted for its scalability, often employs coarse-grained resource allocation, as seen in supercomputing environments like Tianhe-2 with its 24-core nodes, leading to inflexibility in managing fine-grained jobs (Yoo et al. 2003). Fireworks, designed for running high-throughput workflows in supercomputing settings, primarily suffer from limitations due to its reliance on a central database server "pull" procedure, which is further complicated by stringent network security at some computing centers (Jain et al. 2015). Existing research predominantly concentrates on high-throughput jobs in homogeneous resources. In contrast, this study focuses on efficiently processing diverse CPU/GPU/IO-intensive tasks within an HPC-AI framework in supercomputers.

3 Preliminaries

3.1 Notations and definitions

Task and job for computation. The tasks referred to this study mainly concern programs or processes within automated workflows, including script-based format converters, scientific computing applications, system commands, and so on. Each task has a unique identifier called a process identifier (PID) that allows the system to keep track of it. On the other hand, a job typically refers to a series of tasks that are executed sequentially, concurrently, or a combination of both.

Representation of crystal structures. A crystal structure is a description of the ordered arrangement of atoms in 3D space, and the repeating pattern of structural units is termed as a *unit cell*. In Fig. 1, we illustrate a 2D case of periodic patterns for clarity. A *unit cell* can be represented by $\mathcal{M} = (\boldsymbol{A}, \boldsymbol{X}, \boldsymbol{L})$, where $\boldsymbol{A} = [\boldsymbol{a}_1, \boldsymbol{a}_2, ..., \boldsymbol{a}_N] \in \mathbb{R}^{h \times N}$ is the chemical feature matrix, where h is the dimension of embedding vector. $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_N] \in \mathbb{R}^{3 \times N}$ is the Cartesian coordinates for atoms in 3D space, and lattice matrix $\boldsymbol{L} = [\boldsymbol{l}_1, \boldsymbol{l}_2, \boldsymbol{l}_3] \in \mathbb{R}^{3 \times 3}$ is used to describe its periodic patterns.

Invariances for crystal structure. The symmetries of translations, rotations and/or inversion are evident in 3D atom graphs due to the underlying physical law governing the dynamics of atoms or particles remaining consistent regardless of their position. To effectively handle such data, it is crucial to incorporate the inductive bias of symmetry into the model's design.

Here, we consider the unit cell E(3) and periodic invariances for crystal structure as follows, and illustrate the unit cell E(3) invariance in Fig. 1.

Definition 1 (Rotation/Inversion/Transformation Invariance) The function $\boldsymbol{f}: (\boldsymbol{A}, \boldsymbol{X}, \boldsymbol{L}) \rightarrow \rho$ is said to be unit cell E(3) invariant if it satisfies the following condition: for all $\mathrm{O} \in \mathbb{R}^{3 \times 3}$, $|O| = \pm 1$ and $\mathrm{t} \in \mathbb{R}^3$, we have $\boldsymbol{f}(\boldsymbol{A}, \boldsymbol{X}, \boldsymbol{L}) = \boldsymbol{f}(\boldsymbol{A}, \boldsymbol{OX} + t, \boldsymbol{OL})$, where O is rotation and inversion transformations, and t is translation transformations in 3D space.

Definition 2 (Periodic Invariance) A unit cell invariant function $\boldsymbol{f}: (\boldsymbol{A}, \boldsymbol{X}, \boldsymbol{L}) \rightarrow \rho$ is periodic invariant if $\boldsymbol{f}(\boldsymbol{A}, \boldsymbol{X}, \boldsymbol{L}) = \boldsymbol{f}(\boldsymbol{A}', \boldsymbol{X}', \boldsymbol{L}')$, where $\boldsymbol{A}' = [\boldsymbol{a}_1', \boldsymbol{a}_2', ..., \boldsymbol{a}_N]' \in \mathbb{R}^{h \times N}$,

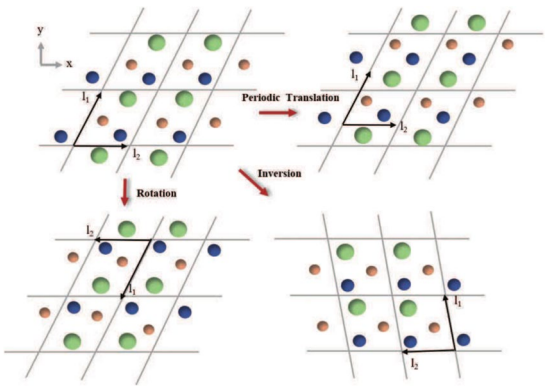

Fig. 1 Illustration of unit cell E(3) invariance

$\boldsymbol{X}^{'} = [\boldsymbol{x}_1^{'}, \boldsymbol{x}_2^{'}, ..., \boldsymbol{x}_N^{'}] \in \mathbb{R}^{3\times N}$, and $\boldsymbol{L}^{'} = \boldsymbol{kL}$, we have $\{(\boldsymbol{a}_i', \boldsymbol{x}_i') | \boldsymbol{a}_i' = \boldsymbol{a}_i, \boldsymbol{x}_i' = \boldsymbol{x}_i + \boldsymbol{kL}, \forall \boldsymbol{k} \in \mathbb{Z}^{3\times 1}\}$.

CSP task definition. The task of CSP predicts for each unit cell the lattice matrix $\boldsymbol{L}$ and the fractional matrix $\boldsymbol{F}$ given its chemical composition $\boldsymbol{A}$, namely, learning the conditional distribution $p(\boldsymbol{L}, \boldsymbol{F} \mid \boldsymbol{A})$.

Ab initio crystal generation definition. The ab initio crystal generation does not have any input. Instead, it obtains a stable crystal structure $\mathcal{M}$ from the hidden distribution of the pre-trained model.

Property prediction task definition. The core objective of this task is to predict the physical or chemical properties $\mathcal{P}$ of a crystal structure, given the input crystal structure $\mathcal{M}$.

4 Methodology

4.1 A high-level overview

The architecture of our proposed HPC-AI framework is systematically depicted in Fig. 2. The framework, named Star-gen, comprises two principal components: an AI-driven pipeline dedicated to generating potential candidate materials, and a sophisticated job execution framework designed for HTC on supercomputers. Detailed elaboration of these components and their functionalities will be presented in the subsequent sections.

4.2 Structure generative model

We first introduce the model for structure generation within the AI-driven pipeline.

Structure generative method. DiffCSP, as introduced in Jiao et al. (2023) Jiao et al. (2024), represents a diffusion methodology designed for learning stable structure distributions pertinent to the Crystal Structure Prediction (CSP) task. This approach innovatively employs a periodic E(3) equivariant model, facilitating the joint optimization of the lattice matrix $\boldsymbol{L}$ and fractional coordinates $\boldsymbol{F} = \boldsymbol{L}^{-1}\boldsymbol{X}$. Comprehensive elaboration of these methods is available in the original publication. Building upon the foundational DiffCSP framework, we have developed an augmented architecture that significantly enhances its capability for constructing large-scale databases.

The composition $\boldsymbol{A} = [\boldsymbol{a}_1, \boldsymbol{a}_2, ..., \boldsymbol{a}_N]$ plays a crucial role in building a database with a uniform distribution of elements a_i and consistent structure sizes N within a unit cell. In DiffCSP method, we consider the composition $\boldsymbol{A}$ as a continuous variable in real space $\mathbb{R}^{h\times N}$, which enables the application of standard DDPM-based method Hoogeboom et al. (2022). The forward diffusion process is described by

$$q(\boldsymbol{A}_t|\boldsymbol{A}_0) = \mathcal{N}\Big(\boldsymbol{L}_t|\sqrt{\bar{\alpha}_t}\boldsymbol{A}_0, (1-\bar{\alpha}_t)\boldsymbol{I}\Big). \tag{1}$$

Here, the variance is modulated by $\beta_t \in (0,1)$, while $\bar{\alpha}_t = \prod_{s=1}^{t}\alpha_t = \prod_{s=1}^{t}(1-\beta_t)$, aligned with the cosine scheduler Nichol (2021). The backward generation process is given by

$$p(\boldsymbol{A}_{t-1}|\mathcal{M}_t) = \mathcal{N}(\boldsymbol{A}_{t-1}|\mu_{\boldsymbol{A}}(\mathcal{M}_t), \sigma_{\boldsymbol{A}}^2(\mathcal{M}_t)\boldsymbol{I}), \tag{2}$$

where $\mu_{\boldsymbol{A}}(\mathcal{M}_t) = \frac{1}{\sqrt{\alpha_t}}\Big(\boldsymbol{A}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\hat{\boldsymbol{\epsilon}}_{\boldsymbol{A}}(\mathcal{M}_t, t)\Big)$, $\sigma_{\boldsymbol{A}}^2(\mathcal{M}_t) = \beta_t\frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}$. The model predicts the denoising term $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{A}}(\mathcal{M}_t, t) \in \mathbb{R}^{h\times N}$. Training focuses on minimizing the one-hot diffusion loss

$$\mathcal{L}_{\boldsymbol{A},\text{continuous}} = \mathbb{E}_{\boldsymbol{\epsilon}_{\boldsymbol{A}}\sim\mathcal{N}(0,\boldsymbol{I}),t\sim\mathcal{U}(1,T)}[\|\boldsymbol{\epsilon}_{\boldsymbol{A}} - \hat{\boldsymbol{\epsilon}}_{\boldsymbol{A}}(\mathcal{M}_t, t)\|_2^2]. \tag{3}$$

The combined training objective for the joint diffusion model, encompassing $\boldsymbol{L}, \boldsymbol{F}, \boldsymbol{A}$ is formulated as follows:

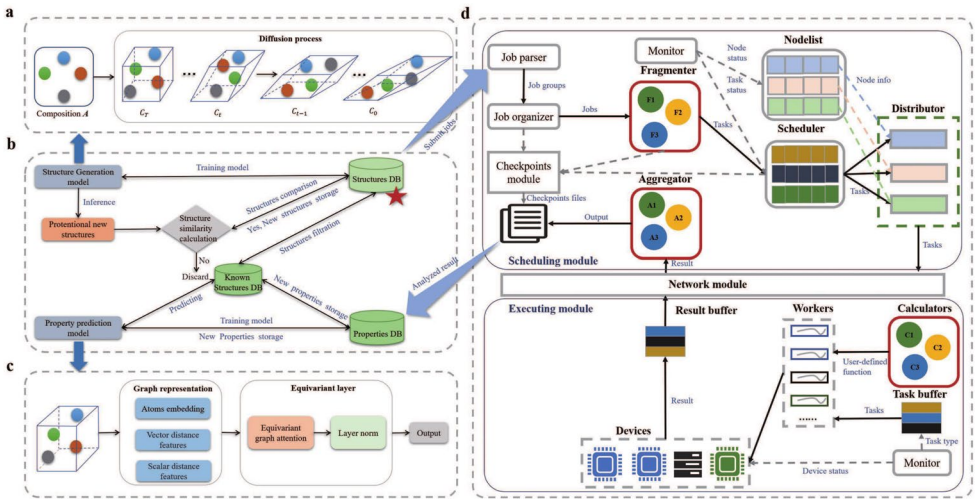

Fig. 2 The architecture of Star-gen. **a**, Schematic of the crystal diffusion model. **b**, Pipeline for AI-driven crystal structure generation and property prediction. **c**, Framework of the property prediction model. **d**, Architecture of the high-throughput job scheduler

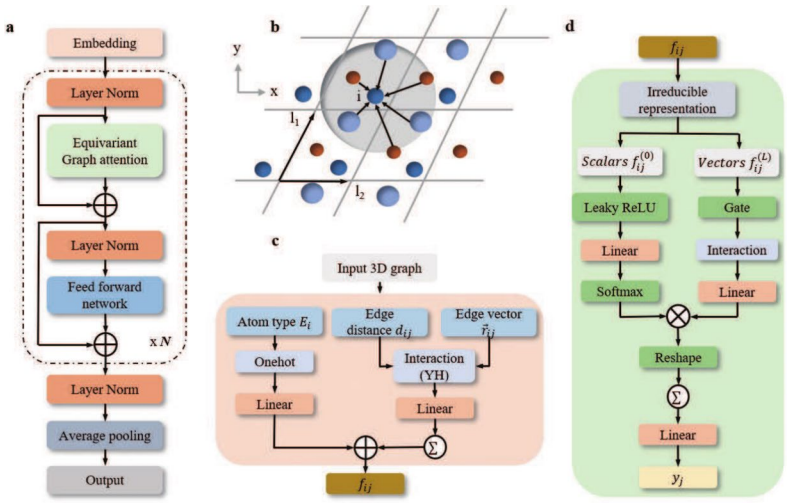

Fig. 3 **a**, Overall architecture of Crysformer. **b**, Schematic of *K*NN algorithm. **c**, Methodology for 3D graph embedding method. **d**, Equivariant graph attention layer. In the context, $\bigotimes$ represents multiplication, $\bigoplus$ signifies addition, and $\sum$ within a circle denotes summation over all neighbors

$$\mathcal{L}_{\mathcal{M}} = \lambda_L \mathcal{L}_L + \lambda_F \mathcal{L}_F + \lambda_A \mathcal{L}_A. \quad (4)$$

For the CSP task, we set $\lambda_L = \lambda_F = 1$, $\lambda_A = 0$, as $\boldsymbol{A}$ remains constant during generation. Conversely, for the ab initio generation task, λ_L and λ_F are kept at 1, while $\boldsymbol{A}$ is increased to larger number to equilibrate the scales of the loss components.

Initially faced with the challenge of acquiring a substantial quantity of effective components $\boldsymbol{A}$ for structure generation via the CSP method, we turn to extracting $\boldsymbol{A}$ from the hidden space distribution of a pre-trained model, thereby enabling the rapid generation of a multitude of structures. Following this, we apply the CSP technique to specifically focus on generating structures for the underrepresented $\boldsymbol{A}$ within our database, using the following detailed methodology:

1. **Extensive sampling by Ab initio method**. This approach entails predicting structures de novo, utilizing a predefined atomic distribution from the pre-training model. In our process, since the number of atoms per unit cell N remains constant during generation, we initially select N based on its distribution in the training set, in line with the approach in Hoogeboom et al Hoogeboom et al. (2022). This procedure leads to a sampled distribution expressed as $p(\mathcal{M}, N) = p(N)p(\mathcal{M}|N)$, where $p(N)$ is derived from a pre-computed data distribution, and $p(\mathcal{M}|N)$ is effectively modeled using DiffCSP.
2. **Fine-tuning data by CSP generation**. A CSP approach to fine-tune the elemental distribution in the database, specifically by using element $\boldsymbol{A}$ as an input parameter for sampling. Specifically, this process involves the elements or compositions that are underrepresented in the pre-trained dataset, we manually select specific elements and their proportions as input parameters for CSP generation.

Pre-Training. We pre-train the model with about 1.14 million non-redundant 3D crystals obtaining from the existing database (including Materials Project,[1] OQMD,[2] Matgen[3] and ICSD). Furthermore, by utilizing both the conventional unit cell and the primitive cell, the dataset size is increased to 2.2 million.

4.3 Properties prediction model

The Structure properties prediction model constitutes another vital component within the AI-driven pipeline. Trained on DFT-based properties, this model is equipped to rapidly assess structure characteristics, identifying potential candidates for further DFT analysis.

Structure properties predicting method. The Crysformer, our proposed model, utilizes SE(3)-equivariant graph attention for 3D geometric analysis. This equivariance is realized through irreducible representation-based features and adaptable operations, supported by e3nn Geiger and Smidt (2022). Figure 3 displays the model's primary components, with extended descriptions following:

1. **Node embedding**. In our graph network methodology, we utilize the k-hot embedding technique Chen et al. (2022) to form the feature vector ($\boldsymbol{a}_{i,k}$), effectively encoding the atomic properties for each atom species.
2. **Edge embedding**. Next, we analyze 3D geometric characteristics, including both the interatomic distances and

[1] https://next-gen.materialsproject.org.

[2] https://www.oqmd.org.

[3] (https://matgen.nscc-gz.cn)

vectors $\vec{r}_{ij}$, using spherical harmonics in the following way:

$$\boldsymbol{E} = \boldsymbol{RBF}(\|\vec{r}_{ij}\|), \tag{5}$$

$$\boldsymbol{x}_{ij} = \varphi(\boldsymbol{h}_i) + \varphi(\boldsymbol{h}_j), \tag{6}$$

$$\boldsymbol{f}_{ij} = \varphi_f(\boldsymbol{x}_{ij} \otimes_{c\boldsymbol{E}}^{TP} \boldsymbol{SH}(\vec{r}_{ij})) \tag{7}$$

In the given expression, $\boldsymbol{RBF}(\|\vec{r}_{ij}\|)$ represents the Radial Basis Function (RBF) expansion for the distance between atoms. The initial edges are formed using the k-nearest neighbor (kNN) approach, as detailed in Yan et al. (2022).φ represents a multi-layer perceptron (MLP). $\boldsymbol{x}_{ij}$ merges the characteristics of nodes i (target) and j (source) using linear layers, to create the initial message. $\boldsymbol{SH}(\vec{r}_{ij})$ signifies the spherical harmonics embeddings (SH) Gasteiger et al. (2020) for the relative position $\vec{r}_{ij}$, and $c\boldsymbol{E}$ is the weight parameterized by $\boldsymbol{E}$. Ultimately, we derive $\boldsymbol{f}_{ij}$ to generate non-linear messages and attention weights.

4.3.1 Equivariant graph attention

Given $\boldsymbol{f}_{ij}$, which includes multiple type-L vectors representing SE(3)-equivariant irreducible representations (irreps) features, we divide $\boldsymbol{f}_{ij}$ into two parts: $\boldsymbol{f}_{ij}^L$ and $\boldsymbol{f}_{ij}^0$. The component $\boldsymbol{f}_{ij}^0$ is scalar and remains unaffected by input variations. In contrast, $\boldsymbol{f}_{ij}^L$ is comprised of type-L vectors, capable of disrupting equivariance. Following the approach in Liao and Smidt (2023), we adopt distinct operations to each subset of $\boldsymbol{f}_{ij}$.

1. **Type-0 features.** Considering $\boldsymbol{f}_{ij}^0$, we utilize the leaky ReLU activation function and apply a softmax operation to compute α_{ij}:

$$\zeta_{ij} = \alpha^\top LeakReLU(\boldsymbol{f}_{ij}^0), \tag{8}$$

$$\alpha_{ij} = \frac{exp(\zeta_{ij})}{\sum_{k\in\mathcal{N}(i)} exp(\zeta_{ik})} \tag{9}$$

Where α is a trainable vector matching the dimensionality of $\boldsymbol{f}_{ij}^0$, and ζ_{ij} is a scalar value.

2. **Type-L features.** A non-linear transformation is applied to $\boldsymbol{f}_{ij}^L$ to generate a non-linear message:

$$\mu_{ij} = Gate(\boldsymbol{f}_{ij}^L), \tag{10}$$

$$\upsilon_{ij} = \varphi_f(\mu_{ij} \otimes_{\omega}^{TP} \boldsymbol{SH}(\vec{r}_{ij})) \tag{11}$$

We employ the gate activation mechanism Weiler et al. 2018 as the equivariant activation function. Standard activation functions are applied to type-0 vectors. For higher-order vectors ($L > 0$), equivariance is achieved by scaling these vectors with non-linearly transformed type-0 vectors. Specifically, given an input x consisting of C_L type-L vectors ($0 < L \leq L_{\max}$) and $(C_0 + P_L \sum_{L=1}^{L_{\max}} C_L)$ type-0 vectors, we apply the SiLU activation function Elfwing et al. 2018 to the first C_0 type-0 vectors and a sigmoid function to the remaining $P_L \sum_{L=1}^{L_{\max}} C_L$ type-0 vectors. This process generates non-linear weights, which are subsequently used to scale each type-L vector. Following the gate activation, the number of channels for type-0 vectors is reduced to C_0. Subsequently, a method analogous to that in eq. 7 is utilized to derive υ_{ij}.

In the final step, both α_{ij} and υ_{ij} are converted into scalars through a multiplication process. We then operate a mean aggregation across all nodes to predict the property value.

$$\mathcal{P}(i) = \frac{1}{|\mathcal{N}(i)|} \sum_{j\in\mathcal{N}(i)} \alpha_{ij} \cdot \upsilon_{ij}, \tag{12}$$

$$\mathcal{P} = \frac{1}{|\mathcal{V}|} \sum_{i\in\mathcal{V}} T_c(i) \tag{13}$$

4.4 High-throughput computing framework

As shown in Fig. 2d, HydraHT adopts a master-worker architecture, where the modules deployed on master and worker nodes are called the scheduling and executing modules, respectively. We describe the details as follows:

1. **Scheduling module**. The *Job parser* converts job descriptions into groups. The *Job organizer* then uses a breadth-first algorithm to order and assign jobs to *Fragmenter*. The *Fragmenter* breaks down jobs into tasks for the *Scheduler*, which manages task queues. The *Distributor* allocates tasks from the *Scheduler* to worker nodes organized by the *Nodelist*. Finally, the *Aggregator* processes results from nodes, performing reduction and post-processing.
2. **Execution model**. The tasks from the scheduling module are stored in a task buffer. *Workers* retrieve tasks from the buffer, linking them with *Calculators* for user-defined computation. HydraHT's API allows users to customize these calculations. Meanwhile, the *Monitor* tracks computing device usage and sends periodic updates to the scheduling module, aiding in task allocation optimization.

Algorithm 1 Task distributing on scheduling module.

```
Initialization: number_of_node_groups, job_id, number_of_tasks_to_send
send_per_group = number_of_tasks_to_send / number_of_node_groups
node_list = get_nodes()
strategy = get_distribute_strategy()
for p_thread in distribute_threads do
    pid = p_thread.id
    node_group = node_list[pid]
    distributor = distributors[pid]
    tasks = load_tasks(send_per_group, job_id)
    for node in node_group do
        task_type = get_task_type(tasks[i])
        status = check_node_status(node, task_type)
        if status == IDLE then
            send_task(node, tasks, strategy)
        end if
        if not tasks then
            exit loop
        end if
    end for
end for
```

We focus on task and IO optimization, with details presented as follows:

Task optimization. Tasks delegated through the HydraHT framework initially undergo scheduling at the node level, entailing more granular resource allocation (encompassing CPU cores, GPUs, and IO) within the individual worker nodes. This approach effectively diminishes the workload of the scheduling module by circumventing the direct oversight of a multitude of fine-grained resources. This is achieved through:

1. **Hierarchical Scheduler**: As delineated in *Algorithm* 1, the HydraHT framework initiates *N* distributor instances corresponding to the number of nodes *N*, with each distributor overseeing the task management of a single node. This methodology facilitates a streamlined management approach for *M* high-throughput tasks, necessitating the oversight of only *M*/*N* distributor instances within the task management framework. Within the computational nodes, as per the user-specified task type, a new execution process is instantiated to consume tasks from the task queue, as delineated in *Algorithm* 2.
2. **Best-fit Device Strategy**. This strategy efficiently assigns tasks-CPU-intensive, GPU-intensive, or IO-intensive-to the best-fit device for simultaneous execution. As depicted in the Fig. 2d, the *Fragmenter* categorizes various tasks according to user definitions and subsequently distributes them to different hardware via the Executing module within the computational nodes. For instance, AI training tasks are executed on GPU devices, while DFT computing tasks are executed on CPU devices.

Algorithm 2 Task processing on executing module.

```
Input: task: Task received from scheduling module
task_type = get_task_type(task)
put_task_in_buffer(task_type, task)
for p_thread in cpu_thread do
  if cpu_pool is not empty then
    compute_task = load_task_from_cpu_buffer()
    submit_task(compute_task)
    send_status_to_scheduling()
  end if
end for
for p_thread in gpu_thread do
  if gpu_pool is not empty then
    compute_task = load_task_from_gpu_buffer()
    submit_task(compute_task)
  else
    send_status_to_scheduling()
  end if
end for
for p_thread in io_thread do
  if io_pool is not empty then
    compute_task = load_task_from_io_buffer()
    submit_task(compute_task)
    send_status_to_scheduling()
  end if
end for
```

IO optimization. We have conducted I/O optimization for the storage architecture of large-scale clusters, such as Tianhe-2, as detailed below:

1. **In-memory computing**. In shared storage clusters like Tianhe-2, where nodes share storage but not memory, efficiency is improved by loading common data into memory for reuse across tasks. Tasks requiring the same data are sent to the same node, reducing redundant data loading. This strategy benefits applications with spatial locality.
2. **Preassigned storage device**. Within clusters featuring heterogeneous architecture, tasks are preemptively designated as IO-intensive by the *Fragmenter* and are executed on the appropriate IO devices. Notably, for mitigating the IO burden in such tasks, we employ local disks or SSD storage solutions.

Fault tolerance. The Hydra framework handles fault tolerance using two main strategies:

1. **Error detection and retry**. Hydra monitors tasks in real time, logs errors for failed tasks, and retries them a predefined number of times. This approach addresses transient issues like network interruptions or resource contention.
2. **Checkpointing**. For long-running tasks, Hydra periodically saves progress as checkpoints. If a failure occurs, the task resumes from the latest checkpoint instead of restarting, reducing computation overhead.

5 Experimental evaluation

Experimental platforms: Our experiments was evaluated on Tianhe-2 supercomputer. The computing nodes contain two Intel(R) Xeon(R) CPU E5-2692, 24 physical cores in total. The AI models are performed on NVIDIA Tesla A800 platform.

Experimental details for DiffCSP. The model takes as input the atomic positions, lattice vectors, and atomic types of a crystal structure. The model comprises 6 layers with 512 hidden states for datasets not explicitly specified otherwise. Fourier Embedding Dimension: Set to 256, to encode the relative positional information in an equivariant manner. The framework adopts DDPM (Diffusion-Driven Probabilistic Modeling) with a cosine scheduler (0.008) to control the variance of the diffusion process on the latent representation

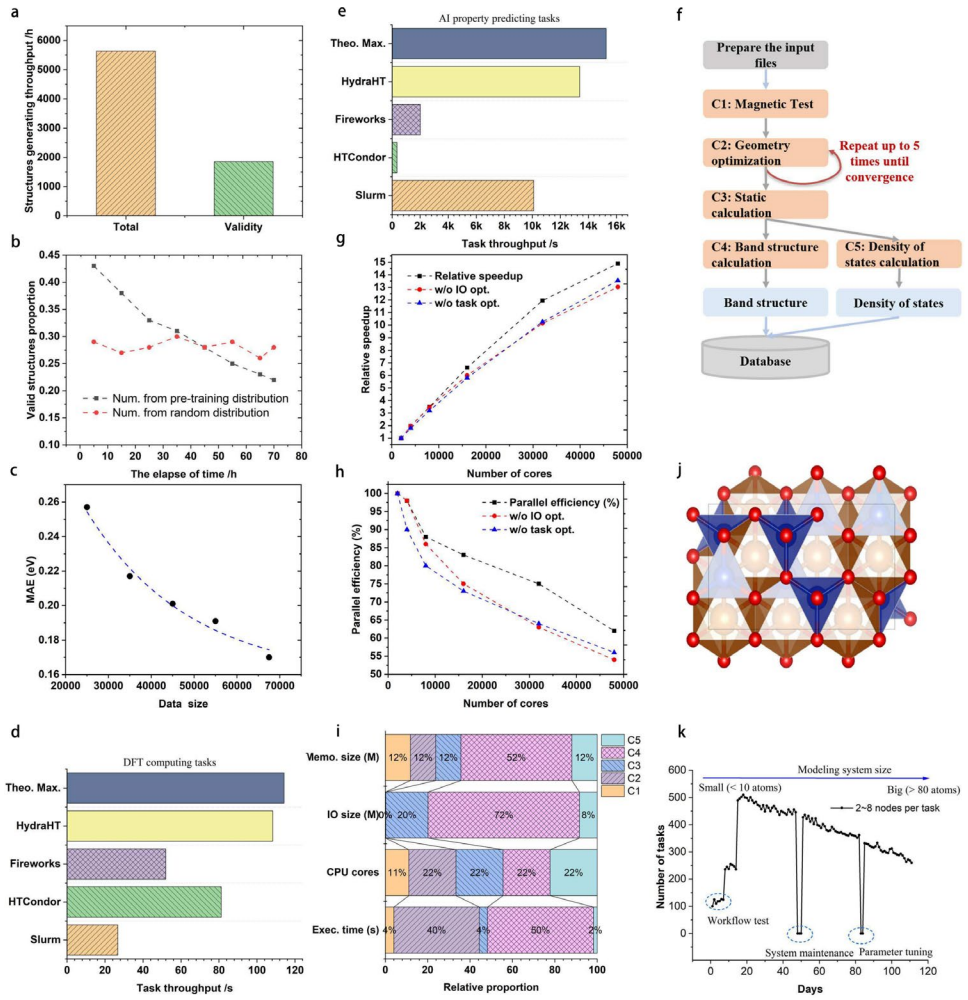

Fig. 4 Experimental results. **a**, Throughput of structure generation in DiffCSP. **b**, Proportion of valid structures evolving over time. **c**, MAE performance in band-gap property prediction with dataset size increase. **d**, Throughput of DFT computing tasks. **e**, Throughput of AI-based property prediction tasks. **f**, Workflow for DFT computing tasks. C1-C4 represent computing steps. Strong scaling performance of **g**, speedup and **h**, parallel efficiency compared to scenarios without IO (w/o IO opt.) and task management optimization (w/o task opt.). **i**, Performance bottleneck analysis for a specific application case. **j**, Snapshot of $Mg_2Si_2O_6$ material. **k**, Evolution of DFT computing task number over time

Table 1 AI-based properties predicting models performance

	Material Project		
		Formation energy	Bandgap
Method	Data size	E (eV/atom)	E_g (eV)
CGCNN Xie and Grossman (2018)	16,485	0.039	0.388
MEGNet Chen et al. (2019)	36,720	0.028	0.33
CrystalNet Chen et al. (2022)	60,000	0.030	0.285
ALIGNN Choudhary and DeCost (2021)	60,000	0.022	0.218
Matformer Yan et al. (2022)	60,000	0.021	0.211
Crysformer	60,000	**0.020**	**0.207**

of lattice. To control the noise scale for the score matching process on atomic coordinate, we use an exponential scheduler with parameters $\sigma_1 = 0.005, \sigma_T = 0.5$. The diffusion step is set to 1000 to ensure stable convergence. The training follows a 6:2:2 split for training, validation, and testing datasets.

Experimental details for Crysformer. During the training process, we utilized a batch size of 64 and trained the model for 150 epochs. A neighborhood radius of 8.0 was employed to define the local environment around each crystal. To represent the features of the crystals, 128 basis functions were used. To mitigate overfitting, a weight decay of 5×10^{-3} was applied. The learning rate was initialized at 5×10^{-5} and reduced to a minimum value of 1×10^{-6}. The AdamW optimizer was employed for efficient optimization. The model architecture consisted of 6 Transformer blocks,

Table 2 Comparison with different models under the message passing framework

Method	Node	Agg'	Edge
CGCNN	$\mathbf{h}_i^{l+1} = \phi(\mathbf{h}_i^l, \mathbf{m}_i)$	$\mathbf{m}_i = \sum_{j=1}^{N} \mathbf{m}_{ij}$	$\mathbf{m}_{ij} = \phi_{rbf}(\Vert\vec{r}_{ij}\Vert)\phi(\mathbf{h}_i^l, \mathbf{h}_j^l)$
MEGNet	$\mathbf{h}_i^{l+1} = \phi(\mathbf{h}_i^l, \mathbf{m}_i)$	$\mathbf{m}_i = \sum_{j=1}^{N} \mathbf{m}_{ij}$	$\mathbf{m}_{ij} = \phi_{rbf}(\Vert\vec{r}_{ij}\Vert)\phi(\mathbf{h}_i^l, \mathbf{h}_j^l, \mathbf{f}_{global})$
CrystalNet	$\mathbf{h}_i^{l+1} = \phi_{comm.}(\mathbf{h}_i^l, \mathbf{m}_i)$	$\mathbf{m}_i = \sum_{j=1}^{N} \mathbf{m}_{ij}$	$\mathbf{m}_{ij} = \phi_{rbf}(\Vert\vec{r}_{ij}\Vert)\phi(\mathbf{h}_i^l, \mathbf{h}_j^l)$
ALIGNN	$\mathbf{h}_i^{l+1} = \phi(\mathbf{h}_i^l, \mathbf{m}_i)$	$\mathbf{m}_i = \sum_{j=1}^{N} \mathbf{m}_{ij}$	$\mathbf{m}_{ij} = \phi_{rbf}(\Vert\vec{r}_{ij}\Vert)\phi_{rbf}(\mathbf{r}_{ijk}^{angle})\phi(\mathbf{h}_i^l, \mathbf{h}_j^l)$
Matformer	$\mathbf{h}_i^{l+1} = \phi(\mathbf{h}_i^l, \mathbf{m}_i)$	$\mathbf{m}_i = \sum_{j=1}^{N} \mathbf{a}_{ij}\mathbf{m}_{ij}$	$\mathbf{m}_{ij} = \phi_{rbf}(\Vert\vec{r}_{ij}\Vert)\phi_{rbf}(\mathbf{r}_{ijk}^{angle})\phi(\mathbf{h}_i^l, \mathbf{h}_j^l)$
Crysformer	$\mathbf{h}_i^{l+1} = \phi(\mathbf{h}_i^l, \mathbf{m}_i)$	$\mathbf{m}_i = \sum_{j=1}^{N} \mathbf{a}_{ij}\mathbf{m}_{ij}$	$\mathbf{m}_{ij} = \phi_{rbf}(\Vert\vec{r}_{ij}\Vert)\phi(\mathbf{h}_i^l, \mathbf{h}_j^l, \vec{r}_{ij})$

each equipped with 8 attention heads. This design enabled the model to effectively capture the intricate relationships and dependencies within the crystal structures. The irreducible representations (irreps) features were constructed using channels of vectors with degrees up to $L_{\max}$. Specifically, we denote C_L type-L vectors as (C_L, L) and $C_{(L,p)}$ type-(L, p) vectors as $(C_{(L,p)}, L, p)$, where brackets denote concatenations of vectors. In this study, the irreps features were defined as containing 512 type-0 vectors and 128 type-1 vectors, which can be represented as:[(512, 0), (128, 1)].

Experimental details for DFT computation. Density functional theory (DFT) calculations were performed using the Vienna Ab initio Simulation Package (VASP) Wang and Pickett (1983); Chan and Ceder (2010). The crystal structures were fully relaxed employing the generalized gradient approximation (GGA) with the SCAN meta-GGA functional Perdew et al. (1996), using pseudopotentials based on the projector augmented wave (PAW) method Blöchl (1994). A plane-wave energy cutoff of 500 eV was applied for all simulations. Brillouin-zone integrations were conducted using the Γ-centered Monkhorst-Pack (MP) scheme Monkhorst and Pack (1976). The calculations were initialized with a k-point mesh featuring a dense sampling density of $2\pi \times 0.04$. The convergence criteria were set to 0.1 meV for total energy and 0.001 eV/Å for ionic forces, ensuring accurate and reliable results.

5.1 Structures generating throughput

As illustrated in Fig. 4a, the implementation of DiffCSP facilitates the generation of approximately 5,663 structures per hour, of which around 1,859 are potentially validated on an hourly basis. For the structures generation, preliminary validity assessment is initially conducted using the following methods.

1. **Reasonable chemical compositions**: We remove the structure with number of chemical components that is more than 10.
2. **Electroneutrality approach**: We calculate the oxidation states of each element using the SCMAT toolkit Davies et al. (2019) and remove structures with charge imbalance.
3. **Exhibiting symmetry**: Observing that materials in nature exhibit high symmetry, structures with space group equal to 1 will be excluded.
4. **Structural similarity**. A structural similarity algorithm is employed to further eliminate structures that are either similar or identical.

Furthermore, an evaluation was conducted for $p(N)$ from the pre-trained data distribution or random sampling. As illustrated in Fig. 4b, the sampling method reliant on pre-trained data distribution initially yields a higher proportion of viable structures (constituting 43% of the total structures). However, as the generation time progresses, the ratio of valid structures diminishes to 22%. In contrast, the purely random generation method maintains a relatively consistent rate, approximately 28%. These results indicate that an initial sampling based on pre-trained data distribution followed by random sampling is a more efficient approach for generating valid structures (Table 1).

5.2 Evaluation of properties predicting model

We use the MP dataset with 69,239 crystal structures, which has been used to train many other GNN models, to evaluate Crysformer. As shown Table 2, our Crysformer achieves the best performance on prediction of formation energy and bandgap. Specially, Crysformer slightly outperform the second best model matformer, which is another transformer based model, indicating that Crysfomer's ability to capture 3D geometric content with edge vector representation.

We further validate the contribution of data size on the Crysformer model in Fig. 4c, the MAE performance decreases from 0.31 eV to 0.207 eV when the training sample size increases from 25,000 to 60,000. Thus, the data-driven AI models can benefit from constructing larger data repositories.

5.3 Overall evaluation of HydraHT

Throughput comparison. To assess the throughput efficiency of HydraHT in comparison to HTCondor, Slurm, and Fireworks, benchmarking tests were conducted on the Tianhe-2 supercomputer, employing 2,000 nodes (48,000 cores in total). The findings are illustrated in Fig. 4d. The term 'theoretical maximum throughput' refers to the task throughput attainable under the hypothesis of job execution on a solitary worker node. This maximum throughput, $\text{Throughput}_{\text{max}}$, is calculated as

$$\text{Throughput}_{\text{max}} = \frac{N_t}{T_{\text{best}}} \tag{14}$$

where $T_{\text{best}} = \frac{T_c}{N_c}$. Here, N_t represents the aggregate number of tasks, T_{best} denotes the theoretical optimal job execution time, T_c signifies the time required for a single core to execute N_t tasks, and N_c indicates the count of processors. When dispatching DFT tasks through Slurm, the task throughput registers at 26.81 tasks/s, markedly inferior to that of HTCondor, Fireworks. This inefficiency stems from Slurm's random task allocation across nodes, neglecting the nodes' load and task duration, thereby precipitating a pronounced load imbalance. In contrast, task throughputs for Fireworks and HTCondor stand at 52.34 and 81.23 tasks/s, respectively. Remarkably, HydraHT achieves a throughput of 105.15 tasks/s, which is 1.3 times higher than that of HTCondor, edging closer to the theoretical maximum throughput.

We utilized properties prediction tasks by Crysformer to assess task throughput. These tasks, characterized by short computation times ranging from 1 to 20 s and involving a few kilobytes of textual input and floating-point values outputs, were used as benchmarks. Owing to Slurm's limitation in directly scheduling single-core processes, we randomly assigned 24 tasks to each of the 24 cores of a single Tianhe-2 node and then submitted them in batches. As illustrated in Fig. 4e, Slurm's throughput appears relatively high when executing Cryformer for property prediction. Slurm tends to frequently create and destroy processes during task execution. In contrast, HydraHT merely utilizes threads from its thread pool to retrieve tasks from the task buffer, incurring minimal overhead. Consequently, HydraHT's throughput marginally surpasses that of Slurm. When submitting Cryformer tasks to HTCondor, the task throughput is observed at only 346.26 tasks/s, which is merely one-fortieth of that achieved by HydraHT. This trend is predominantly due to the significant performance bottleneck encountered by HTCondor's task scheduler when the computing cluster is scaled to a certain extent, and the task execution time is short. Furthermore, we also observed that Fireworks achieves a task throughput of approximately 2010.34 tasks/s, substantially lower than HydraHT. This is primarily attributed to Fireworks' single-tier scheduling and the atomic nature of its task distribution process, facilitated through MongoDB's atomic operations.

5.4 Real-world HTC application

We first conduct the HTC tasks to validate our optimization on runtime workload system. We perform about 10 million DFT computing tasks on Tianhe-2 supercomputer. The whole DFT computing pipeline as shown in Fig. 4f using 1∼ 144 CPU cores per task based on material's unit cell sizes and set a low precision in the DFT calculation parameters to ensure that each task can be completed in less than half an hour. As shown in Fig. 4g and Fig. 4h, the relative speedup as well as parallel efficiency decrease when we do not optimize the IO and task management. Specially, we can observe that task management optimization had a positive impact on all scales of testing, while IO management optimization shows a noticeable improvement starting at 16,000 CPU cores.

As depicted in Fig. 4i, we trace the IO characteristics of application case studies by $Mg_2Si_2O_6$ material (Fig. 4j). We identified that the C4 task (band calculation) within the DFT computational process represents a computational bottleneck, entailing extensive IO operations. Consequently. For this segment of the workload, we employed 100 large memory nodes (128 GB) within the Tianhe-2 system, utilizing local RAMDISK as a temporary storage solution for data. This strategy effectively alleviates the pressure on shared storage resources.

Ultimately, we executed stringent and standardized DFT calculations on 80,000 crystal structures, while closely monitoring the runtime tasks for DFT computing in Fig. 4k. In practice, we observe that one of the bottleneck for performing large scale multi-step DFT computations on HPC systems is how to deal with failed tasks, which are caused by HPC hardware or DFT computing parameters. For instance, we meet two mass job failures as shown in Fig. 4k, one is caused by system maintenance, and the other is parameter tuning for big material size.

6 Discussions

AI-driven pipeline. In our pursuit of discovering valid and novel material structures, we have developed a sophisticated AI-driven pipeline. At its core, this pipeline commences with an advanced structure generation model, adept at rapidly generating a wide array of material structures. Following this, an AI-based evaluation phase ensues, employing a model trained to predict properties based on DFT. This step is crucial for filtering structures, prioritizing those with

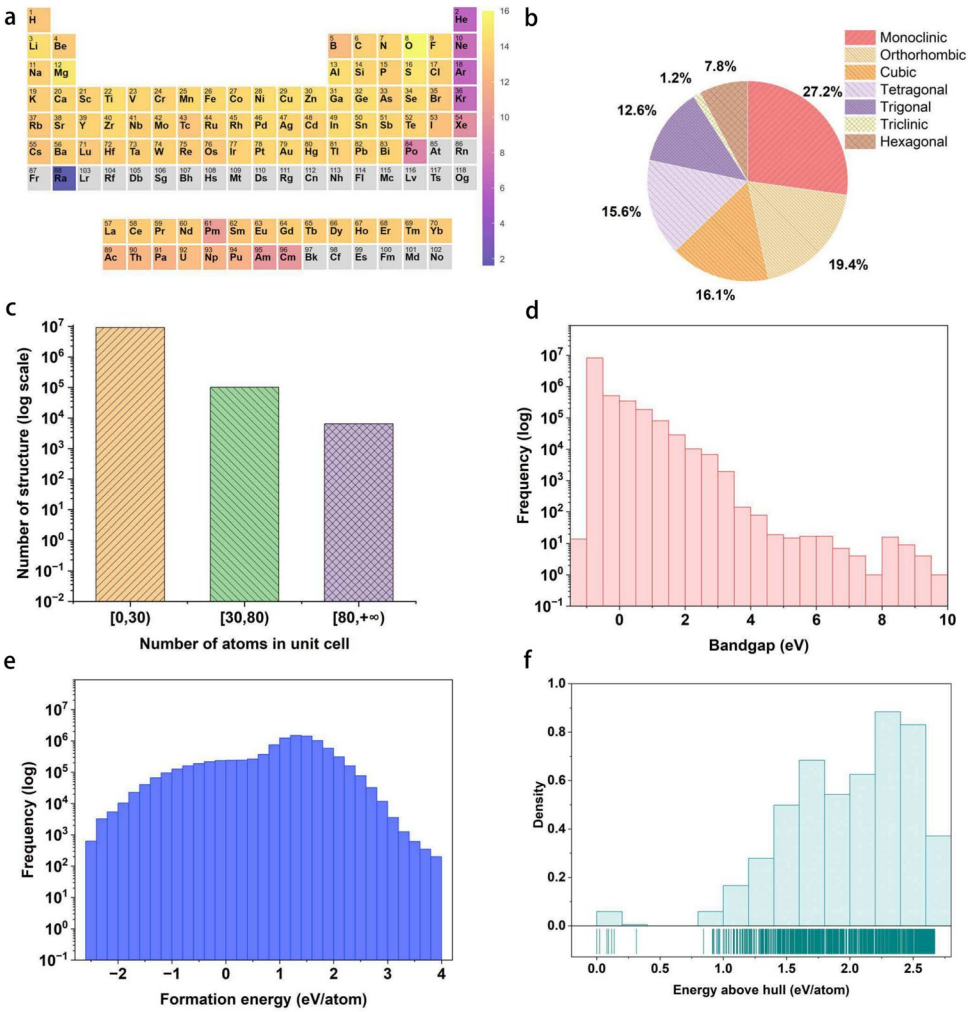

Fig. 5 Hypothetical database data statistics. **a**, Frequency of chemical species. **b**, Distribution of crystal system classifications. **c**, Distribution of atom number in primary cell. **d**, Frequency of predicted bandgap values. **e**, Frequency of predicted formation energy values. **f**, Distribution of energy above hull values of generated structures using MP dataset as energy reference

lower composition energy for in-depth DFT analysis. Such a strategic approach significantly enhances the efficiency in pinpointing potential material candidates.

Our findings show that DiffCSP, using an ab initio generative method, autonomously produces a wide range of potential material structures without requiring initial data. Enhancing this process with random sampling or sampling from pre-trained distributions broadens the variety of structures generated. Furthermore, incorporating the CSP method in our framework enables targeted generation of structures based on given elemental compositions. This versatile approach diversifies potential structures and adds flexibility to meet specific structural needs.

We evaluated kernel methods from the literature based on node, aggregation, and edge update operations as summarized in Table 2. Most algorithms, except CrystalNet, use the same node update operations. Crysformer and Matformer, both based on the transformer architecture, update messages using attention weights $\boldsymbol{a}_{ij}$ in the aggregation phase. The primary difference lies in edge update operations: CGCNN, MEGNet, and CrystalNet use interatomic distances as edge features, limiting their sensitivity to bond angles and local geometry. In contrast, ALIGNN and Matformer, by incorporating bond angle data, demonstrate the value of 3D geometric content in atomic graph representation. Crysformer further enhances this by encoding 3D geometry through the edge vector $\vec{r}_{ij}$, offering a more detailed structural representation.

Hybrid task scheduling. Our HydraHT framework showcases superior efficacy in managing both AI-driven and DFT computations on the Tianhe-2 supercomputer. Its distinctive feature lies in the strategic allocation of tasks to specific hardware, tailored to their CPU, GPU, and IO demands, thereby optimizing execution efficiency. HydraHT transcends traditional schedulers such as Slurm or HTCondor, offering heightened flexibility in task scheduling and maximizing the use of diverse hardware resources including GPUs, CPUs, and local storage. Uniquely, HydraHT diverges from domain-specific tools like Fireworks by adopting a hierarchical scheduling approach, significantly boosting the scalability and distribution of tasks across complex computational landscapes.

Computational materials database. Using our HPC-AI framework, we produced around 10 million hypothetical crystal structures in a week, determining properties like bandgap and formation energy via the Crysformer model. As shown in Fig. 5a–e, we analyzed chemical elements, crystal cell types, atomic counts, and property predictions. Our database, with a balanced composition of 94 element types, surpasses the 89 in MP and OQMD and includes elements like Cm, Rn, Ra, Po, and Am. It features more large-system structures (atomic number over 80) compared to MP's 129 and OQMD's absence. The property distribution, with bandgaps of 0 to 9.57 eV and formation energies from −2.26 to 3.99 eV/atom, aligns with MP and approximates OQMD, suggesting potential stability. To further validate the effectiveness of the generated structures, we performed DFT calculations on 1,024 sampled structures. Referencing MP dataset energy, we calculated the convex hull energies and evaluated whether the structures are on-hull. As shown in the Fig. 5f, approximately 0.59% of the structures are on the hull, and 2.44% are within 1 eV/atom from the hull.

Using the HPC-AI framework, we achieved around 80,000 DFT data points in three months on the Tianhe-2 supercomputer. This demonstrates the framework's capability for efficient high-throughput calculations, contributing to data-driven materials design. This framework skillfully orchestrated a balance between AI-driven property prediction tasks and the more demanding CPU and IO-intensive DFT calculations. Notably, our system demonstrated robust scalability and performance, with a parallel efficiency surpassing 89% on Tianhe-2. This efficiency is indicative of the framework's sophisticated task management capabilities and its adept use of the supercomputer's extensive computational resources.

7 Conclusion and future works

In summary, we present an HPC-AI framework to construct an expansive, varied computational materials database. Our pipeline integrates a periodic-E(3)-equivariant diffusion model for structure generation and a transformer-based model for 3D geometric property prediction, culminating in DFT evaluations. Efficiently executing diverse, intensive tasks, our high-throughput computing framework capitalizes on supercomputers' heterogeneous nodes and shared storage. This HPC-AI framework has produced approximately 10 million novel hypothetical crystal structures, establishing the most extensive database of its kind. Harnessing 2,000 Tianhe-2 supercomputer nodes, we achieved around 80,000 DFT data points in three months, marking a significant leap in data-driven materials design.

In the following work, we aim to continue high-throughput DFT computations on AI-generated structures to build an extensive computational materials database. Concurrently, we will enhance our HPC-AI framework's functionality and adaptability, integrating advanced workflow management tools for complex computational tasks and developing robust fault-tolerance mechanisms for efficient management and recovery from failures. We will also leverage our comprehensive database of computed materials and their properties to refine our generative models and property prediction algorithms, enhancing their generalization capabilities and predictive accuracy. We will also enhance the framework's versatility by deploying it on a wider range of supercomputing hardware, such as supercomputing systems based on Sunway or Kunpeng processors. These ongoing improvements are anticipated to markedly increase our framework's effectiveness and utility in materials science.

Author Contributions Yutong Lu managed this project. Pin Chen conceived and implemented the framework. Qing Mo, Zexin Xu and Xianwei Zhang analyzed the data. All authors read and approved the manuscript.

Funding This research is supported by the National Key R&D Program of China (Grant No. 2023YFB3002202), the Guangdong Provincial Key Area R&D Program of Guangdong Provincial (Grant No. 2024B0101040005) and the National Natural Science Foundation of China (Grant No. 62461146204).

Data availability All data is available from the Matgen platform: (https://matgen.nscc-gz.cn.)

Declarations

Conflict of interest No.

Ethics Approval and Consent to Participate Not applicable.

References

Blöchl, P.E.: Projector augmented-wave method. Phys. Rev. B **50**(24), 17953 (1994)

Butler, K.T., Davies, D.W., Cartwright, H., Isayev, O., Walsh, A.: Machine learning for molecular and materials science. Nature **559**(7715), 547–555 (2018)

Chan, M., Ceder, G.: Efficient band gap prediction for solids. Phys. Rev. Lett. **105**(19), 196403 (2010)

Chen, C., Ye, W., Zuo, Y., Zheng, C., Ong, S.P.: Graph networks as a universal machine learning framework for molecules and crystals. Chem. Mater. **31**(9), 3564–3572 (2019)

Chen, P., Chen, J., Yan, H., Mo, Q., Xu, Z., Liu, J., Zhang, W., Yang, Y., Lu, Y.: Improving material property prediction by leveraging the large-scale computational database and deep learning. J. Phys. Chem. C **126**(38), 16297–16305 (2022)

Choudhary, K., DeCost, B.: Atomistic line graph neural network for improved materials property predictions. npj Computational Materials 7(1), 1–8 (2021)

Choudhary, K., Zhang, Q., Reid, A.C., Chowdhury, S., Van Nguyen, N., Trautt, Z., Newrock, M.W., Congo, F.Y., Tavazza, F.: Computational screening of high-performance optoelectronic materials

using optb88vdw and tb-mbj formalisms. Sci. Data **5**(1), 1–12 (2018)
Davies, D.W., Butler, K.T., Jackson, A.J., Skelton, J.M., Morita, K., Walsh, A.: Smact: semiconducting materials by analogy and chemical theory. J. Open Source Softw. **4**(38), 1361 (2019)
Elfwing, S., Uchibe, E., Doya, K.: Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. Neural Networks 107, 3–11 (2018) https://doi.org/10.1016/j.neunet.2017.12.012 . Special issue on deep reinforcement learning
Fajardo, E., Dost, J., Holzman, B., Tannenbaum, T., Letts, J., Tiradani, A., Bockelman, B., Frey, J., Mason, D.: How much higher can htcondor fly? In: Journal of Physics: Conference Series, vol. 664, p. 062014 (2015). IOP Publishing
Gasteiger, J., Groß, J., Günnemann, S.: Directional message passing for molecular graphs. arXiv preprint arXiv:2003.03123 (2020)
Geiger, M., Smidt, T.E.: e3nn: Euclidean neural networks. CoRR **abs/2207.09453** (2022) https://doi.org/10.48550/arXiv.2207.09453
Hoogeboom, E., Satorras, V.G., Vignac, C., Welling, M.: Equivariant diffusion for molecule generation in 3d. In: International Conference on Machine Learning, pp. 8867–8887 (2022). PMLR
Jain, A., Ong, S.P., Hautier, G., Chen, W., Richards, W.D., Dacek, S., Cholia, S., Gunter, D., Skinner, D., Ceder, G.: Commentary: The materials project: A materials genome approach to accelerating materials innovation. APL Mater. **1**(1), 011002 (2013)
Jain, A., Ong, S.P., Chen, W., Medasani, B., Qu, X., Kocher, M., Brafman, M., Petretto, G., Rignanese, G.-M., Hautier, G.: Fireworks: a dynamic workflow system designed for high-throughput applications. Concurr. Comput. Pract. Exp. **27**(17), 5037–5059 (2015)
Jiao, R., Huang, W., Lin, P., Han, J., Chen, P., Lu, Y., Liu, Y.: Crystal structure prediction by joint equivariant diffusion. Adv. Neural Inf. Process. Syst. **36**, 8 (2024)
Kirklin, S., Saal, J.E., Meredig, B., Thompson, A., Doak, J.W., Aykol, M., Rühl, S., Wolverton, C.: The open quantum materials database (oqmd): assessing the accuracy of dft formation energies. npj Computational Materials 1(1), 1–15 (2015)
Liao, Y.-L., Smidt, T.: Equiformer: Equivariant graph attention transformer for 3d atomistic graphs. In: International Conference on Learning Representations (2023)
Marzari, N., Ferretti, A., Wolverton, C.: Electronic-structure methods for materials design. Nat. Mater. **20**(6), 736–749 (2021)
Monkhorst, H.J., Pack, J.D.: Special points for brillouin-zone integrations. Phys. Rev. B **13**(12), 5188 (1976)
Nichol, A.Q., Dhariwal, P.: Improved denoising diffusion probabilistic models. In: International Conference on Machine Learning, pp. 8162–8171 (2021). PMLR
Perdew, J.P., Burke, K., Ernzerhof, M.: Generalized gradient approximation made simple. Phys. Rev. Lett. **77**(18), 3865 (1996)
Reed, D., Gannon, D., Dongarra, J.: Reinventing high performance computing: challenges and opportunities (2022)
Reiser, P., Neubert, M., Eberhard, A., Torresi, L., Zhou, C., Shao, C., Metni, H., Hoesel, C., Schopmans, H., Sommer, T.: Graph neural networks for materials science and chemistry. Commun. Mater. **3**(1), 93 (2022)
Singh, R., Sharma, A., Singh, P., Balasubramanian, G., Johnson, D.D.: Accelerating computational modeling and design of high-entropy alloys. Nat. Comput. Sci. **1**(1), 54–61 (2021)
Wang, C., Pickett, W.: Density-functional theory of excitation spectra of semiconductors: application to si. Phys. Rev. Lett. **51**(7), 597 (1983)
Weiler, M., Geiger, M., Welling, M., Boomsma, W., Cohen, T.S.: 3d steerable cnns: Learning rotationally equivariant features in volumetric data. Adv. Neural Inf. Process. Syst. **31** (2018)
Xie, T., Fu, X., Ganea, O., Barzilay, R., Jaakkola, T.S.: Crystal diffusion variational autoencoder for periodic material generation. In: The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022 (2022)
Xie, T., Grossman, J.C.: Crystal graph convolutional neural networks for an accurate and interpretable prediction of material properties. Phys. Rev. Lett. **120**(14), 145301 (2018)
Yan, K., Liu, Y., Lin, Y.-C., Ji, S.: Periodic graph transformers for crystal material property prediction. ArXiv **abs/2209.11807** (2022)
Yan, D., Smith, A.D., Chen, C.-C.: Structure prediction and materials design with generative neural networks. Nat. Comput. Sci. **3**(7), 572–574 (2023)
Yoo, A.B., Jette, M.A., Grondona, M.: Slurm: Simple linux utility for resource management. In: Workshop on Job Scheduling Strategies for Parallel Processing, pp. 44–60 (2003). Springer
Zhang, X., Limei Wang, ...e.: Artificial intelligence for science in quantum, atomistic, and continuum systems. CoRR **abs/2307.08423** (2023) https://doi.org/10.48550/ARXIV.2307.08423arXiv:2307.08423
Zhang, X., Tian, Y., Chen, L., Hu, X., Zhou, Z.: Machine learning: a new paradigm in computational electrocatalysis. J. Phys. Chem. Lett. **13**(34), 7920–7930 (2022)

Pin Chen received the PhD degrees in computer science from the School of Computer Science and Engineering, Sun Yat-sen University, Guangzhou, China. He is currently an Associate Research Fellow at the National Supercomputer Center in Guangzhou. His research interests include scientific numerical simulations, intelligent model development, and domain-specific platform development.

Qing Mo graduated with a Bachelor's degree from Tianjin University and pursued her Master's degree at Northeastern University in the United States. She is currently employed at the National Supercomputer Center in Guangzhou. Her research interests encompass software development and the exploration of intelligent models in interdisciplinary fields.

Zexin Xu graduated from the School of Computer Science and Engineering, Sun Yat-sen University, Guangzhou, China. He is currently working in the High-Performance Computing Department at the National Supercomputing Center in Guangzhou. His research focuses on visualization technologies, high-performance computing, and artificial intelligence.

Xianwei Zhang received his Ph.D. in Computer Science in 2017 from the University of Pittsburgh, USA, and B.S. degree in 2011 from Northwestern Polytechnical University, Xi'an, China. He is currently an Associate Professor in the School of Computer Science and Engineering at Sun Yat-sen University, Guangzhou, China. Prior to joining academia, he worked as a researcher and engineer in AMD Inc. His research interests include computer architecture and systems, GPU, compilation, high-performance computing, and intelligent computing.

Yutong Lu (Member, IEEE) received the MSc and PhD degrees in computer science from the National University of Defense Technology (NUDT), Changsha, China. She is currently a professor with the School of Computer Science and Engineering, Sun Yat-sen University, Guangzhou, China. She is also the director of National Supercomputer Center in Guangzhou. Her research interests include parallel system management, high-speed communication, distributed file systems, and advanced programming environments with the MPI.