ATOMISTIC LANGUAGE MODELS UNDERSTAND AND GENERATE MATERIALS

Anonymous authors
Paper under double-blind review

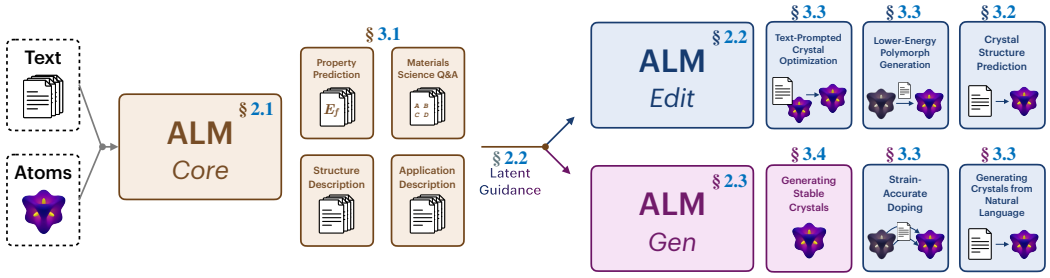

Figure 1: **Atomistic Language Models bridge natural language and 3D atomic coordinates to *understand, generate, and optimize materials*.** This new paradigm allows a single autoregressive backbone to characterize the structure, properties, and applications of a material, as well as guide the discovery of new ones, all without lossy text representations.

ABSTRACT

Atomistic structure and natural language have long been modeled separately, with language models either calling atomistic models as tools or being fine-tuned on lossy textual encodings that discard atomistic information. We introduce **Atomistic Language Models (ALMs)** to pursue native multimodality, in which a single language backbone understands atomistic structures, generates materials from natural language, and optimizes crystal structures as instructed by text. By unifying a pretrained atomistic encoder, large language model, and denoising diffusion model through purely continuous projectors and staged training, ALMs achieve state-of-the-art results on crystal structure prediction and *de novo* generation. ALMs are enabled by a *continuous bridge that maps language model embeddings directly into the steering space of atomistic diffusion*, and are assisted by **Text-to-Crystal Feynman–Kac (T2C-FK)**, a particle-based sampler that scores partial denoising trajectories to enforce stoichiometric targets at inference time. We also introduce **ALM Bench**, the first benchmark for language-instructed crystal generation and materials optimization. ALMs beat frontier LLMs, including GPT-5.6 Sol, on inorganic crystalline doping, inverse design, and generation for desired applications. All code is provided as supplementary material.

1 INTRODUCTION

Domain experts in materials science reason in two modes. One is *continuous* and geometric: atomic coordinates, a periodic unit cell, or the actual structure on which physical laws act. The other is *discrete* and linguistic: a phase, a processing history, or the experimental context that determines whether and how the material can be made. An ideal model of materials would move fluently between these modalities, parsing a structure to predict its properties, generating a structure from a natural language specification, and optimizing materials exactly as desired.

The two modalities carry fundamentally different representational biases. State-of-the-art (SoTA) atomistic models (Wood et al., 2026; Batatia et al., 2025; Rhodes et al., 2025) operate on graphs built from 3D coordinates and periodic boundary conditions, many having inductive priors like symmetry,

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

locality, and stoichiometry baked into their architectures. Language models operate on discrete tokens with no native notion of geometry or periodicity. Beneath these surface incompatibilities lies a deeper one: in matter, distances in real space and distances in semantic space are decoupled. A perturbation of a few atoms can leave a crystal structure nearly unchanged geometrically while moving it across a phase boundary; different polymorphs of the same composition can differ by less than an angstrom in coordinates and by orders of magnitude in conductivity or hardness. Each modality must therefore be modeled in its own, native representation. Previous approaches bridge this gap by training on lossy textual encodings of materials, like CIF files (Antunes et al., 2024), Wyckoff strings (Xu et al., 2026), or other custom string representations (Gruver et al., 2024; Alampara et al., 2024). However, they lose crucial geometric context that determines material behavior (Gupta et al., 2022; Edamadaka et al., 2025) or catastrophically forget their natural language abilities (Ozawa et al., 2024).

The natural next step is to compose pipelines of unimodal specialists: large language models (LLMs) call structure generators as tools (Lu et al., 2026; Xie et al., 2022; Jiao et al., 2023; Zeni et al., 2025), generative models post-process language model outputs (Yang et al., 2024), and property predictors are wrapped behind text interfaces (Deng et al., 2026). Such pipelines inherit *three failure modes*: components communicate only through discrete surface forms, discarding continuous geometric information; they train independently, so structure and language never develop a shared latent space; and generation can only be steered through what a text prompt can express, not through the latent space the model uses to reason. On the other hand, multimodal language models in literature that directly understand 3D atomic coordinates are purely property predictors, and cannot generate new materials (Moro et al., 2025; Suzuki et al., 2025; Tang et al., 2026; Cui et al., 2025).

A different paradigm avoids these failure modes. Atomistic encoders, language models, and atomistic structure generators each excel and fail at different predictive and generative tasks. Combining them into a single, end-to-end model with shared latent spaces addresses those failure modes more naturally than any pipeline can; representations are shared (Liu et al., 2023a), gradient signals flow across modality boundaries during training (Chen et al., 2025), and conditional generation can be steered through the same latent space the model uses to reason (Li et al., 2023b).

The **Atomistic Language Models (ALMs)** realize this paradigm, unifying a pretrained atomistic encoder (Rhodes et al., 2025), language model (Yang et al., 2025), and denoising diffusion decoder (Zeni et al., 2025) through continuous, cross-modal projectors (Fig. 2). ALMs are trained in three stages: *alignment* of the modalities, *multi-pattern instruction tuning* that instills structural reasoning over both text and structure, and *guided generation* in which embeddings extracted from the language model steer the denoising trajectory. **ALM Core** is a language model specialized in characterizing materials structure, properties, and applications by natively taking in 3D atomic coordinates, lattice parameters, and element types. **ALM Edit** is finetuned on **Core**, learning to strongly steer (Ho & Salimans, 2022) a denoising diffusion decoder built for crystal structure prediction, and *is capable of optimizing a crystal as instructed by natural language*. **ALM Gen** replaces this decoder with a weakly steered diffusion model, enabling *de novo* generation of inorganic crystals. To bridge the gap between the instruction-following abilities of **ALM Edit** and the competitive stability of **ALM Gen**'s generated materials, we introduce an inference-time steering method, **Text-to-Crystal Feynman–Kac (T2C-FK)**, to improve how closely **ALM Gen** follows stoichiometric targets when prompted. To evaluate the text-conditioned optimization capabilities of **ALM Edit**, we introduce **ALM Bench**, comprising over 7,000 natural language and inorganic crystal instruction pairs for evaluating language instruction-following in materials discovery.

ALM Core matches or outperforms prior language model approaches and unimodal machine learning interatomic potentials (MLIPs; Rubungo et al., 2025; Tang et al., 2026). **ALM Edit** achieves state-of-the-art crystal structure prediction performance on MP-20 (Xie et al., 2022) and MPTS-52 (Jiao et al., 2023) while beating frontier language model baselines on several tasks in **ALM Bench**. **ALM Gen** achieves competitive performance on *de novo* generation and beats previous language-based approaches.

2 Atomistic Language Models

Atomistic Language Modeling couples three components through trained, continuous projectors (Fig. 2A): a machine-learning interatomic potential that encodes each atom of a crystal ($\mathcal{E}$, OrbV3;

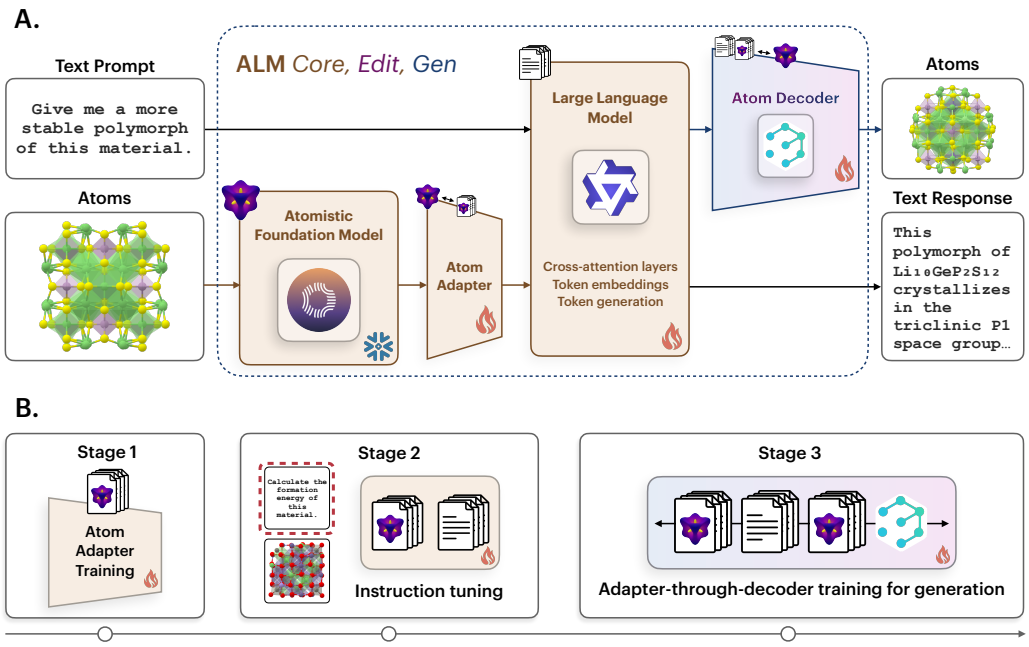

Figure 2: **Atomistic Language Models understand atoms as soft tokens from a machine learning interatomic potential and generate inorganic crystals by steering diffusion models with classifier-free guidance.** A. The ALM architecture. B. The staged training curriculum, which enables property prediction and structure generation in the same model.

Rhodes et al., 2025), a causal language model that serves as the central backbone (ϕ, Qwen3-8B; Yang et al., 2025), and a denoising diffusion model that decodes the language model's latent instructions into 3D crystal structures ($\mathcal{D}$, MatterGen; Zeni et al., 2025). **ALM Core** (§2.1) is the instruction-tuned core that can take in crystals as represented by continuous unit cell parameters $\mathbf{L}$, continuous fractional coordinates $\mathbf{X}$ of each of the atoms in the material, and a discrete atomic-number assignment $\mathbf{A}$ that corresponds to their element types (an exhaustive list of notation is available in Table 8). Two generative variants bridge this base to diffusion decoders: a strongly text-conditioned crystal structure prediction model (**ALM Edit**, §2.2) and a weakly text-conditioned *de novo* generator (**ALM Gen**, §2.3).

Standard cross-modal interfaces discretize each modality (Alayrac et al., 2022; Liu et al., 2023a; Chen et al., 2025) into codebooks generated autoregressively at inference time (van den Oord et al., 2017). Crystalline matter resists both ideas: TiO_2 polymorphs differ by small distances between atoms, yet have band gaps that are hundreds of meV apart, and single-atom defects in large supercells shift formation energy without moving many structural fingerprints (Drautz, 2019). Therefore, all latent representations are kept continuous in ALMs, with no rounding or vector quantization (Appendix B.1.3). Aligning the encoder, backbone, and decoder in this single, continuous latent space also confines all domain-specific inductive biases to the encoder $\mathcal{E}$ and decoder $\mathcal{D}$. Therefore, although this work tackles crystalline inorganic materials, ALMs can extend to other types of matter by retraining on new structures or utilizing different encoders and decoders.

2.1 ALM Core: Understanding materials using their native 3D structure

ALM Core is an instruction-tuned multimodal language model that reads each material's atomic coordinates, lattice parameters, and element types, answering in natural language and serving as the foundation for our generative variants (**Edit** and **Gen**). The frozen encoder $\mathcal{E}$ maps each material ($\mathbf{L}$, $\mathbf{X}$, $\mathbf{A}$) to per-atom embeddings $\mathbf{H} \in \mathbb{R}^{N_p \times d_\mathcal{E}}$, which a two-layer GELU MLP P_{in} projects into the LLM's token space as soft tokens, a lower-parameter method than the gated attention-based bridges of prior multimodal materials models (Tang et al., 2026). The model is trained in two stages (Fig. 2B,

1-2). P_{in} is first aligned alone on deterministic structural descriptions (Ganose & Jain, 2019) from LLM4Mat-Bench (Rubungo et al., 2025). The LLM is then instruction-tuned via LoRA (Hu et al., 2022) on those descriptions, as well as narratives about applications, property prediction, and three text-only tasks that prevent catastrophic forgetting (Liu et al., 2025). Loss functions, bucket weights, optimizer settings, and ablations are in Appendix B.1 and C.1.

2.2 ALM EDIT: TEXT-CONDITIONED CRYSTAL GENERATION

ALM Edit finetunes **ALM Core** to enable inverse materials design, guided by prompts consisting of natural language *and* materials. This unlocks text-conditioned structural optimization, in which an inputted material is edited according to natural language instructions. MatterGen is used as the denoising diffusion model $\mathcal{D}$ for ALMs due to its extensibility to different conditioning heads via classifier-free guidance (CFG; Zeni et al., 2025; Appendix B.2). To build **ALM Edit**, $\mathcal{D}$ was trained from scratch with one architectural modification: denoising only over atom coordinates $\mathbf{X}$ and unit cell parameters $\mathbf{L}$.

Instead of denoising over element types and counts, the language model generates the composition ($\mathbf{A}$) that $\mathcal{D}$ observes. MatterGen cannot reliably reproduce desired stoichiometries and positions provided through CFG, so this choice ensures **Edit** produces the structures it was prompted to generate (discussed in §2.3 and Appendix B.2.2). $\mathcal{D}$ is trained on MP-20, MPTS-52, and all other structures that **Core** was trained to understand (Appendix C.2.1). The language model autoregressively generates the composition that $\mathcal{D}$ observes.

We introduce a *producer–consumer* bridge (Fig. 3, Eq. 14) to connect **Core** to this composition-observing decoder. To encode the textual task and structural information about the inputted material, $K = 8$ "atomistic" tokens (A_1 through A_8 in Fig. 3) are teacher-forced onto the LLM's response, causally attending to the prompt and inputted material's soft tokens. Their final-layer hidden states $\mathbf{Z} \in \mathbb{R}^{K \times d_{\text{LM}}}$ are extracted as continuous latents to guide $\mathcal{D}$ without passing through a discrete-vocabulary bottleneck (Chen et al., 2025). Specifically, the *producer* cross-attends to K atomistic, composition, and prompt token embeddings. The *consumer* then injects the fixed-size conditioning vector $\mathbf{C}$ into every block of $\mathcal{D}$'s score network through a cross-attention branch via CFG.

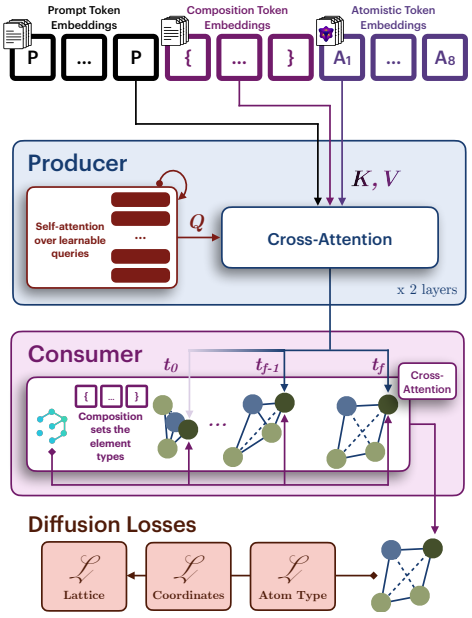

Figure 3: **A language-to-atomistic bridge enables the steering of crystal generation. ALM Edit** uses all components above. **ALM Gen** swaps the Q-Former-style (Li et al., 2023b) producer for a lightweight per-token MLP (no learned queries or prompt context) feeding the same consumer, and does not emit composition embeddings.

During training (Stage 3 in Fig. 2B), $\mathcal{D}$'s **denoising diffusion losses backpropagate through the consumer, the producer, and the language model's atomistic-token hidden states Z**. Additional losses, like per-element composition-count, anchor $\mathbf{Z}$ to a structurally meaningful direction; without this auxiliary, the atomistic token hidden states collapse to a near-constant direction across prompts (Appendix B.2.1). **ALM Edit** is trained on seven buckets of tasks, including crystal structure prediction and all tasks in **ALM Bench** (listed in Appendix C.2.1).

2.3 ALM GEN: *de novo* CRYSTAL GENERATION

While conditional models like **ALM Edit** generate crystals as instructed by text, *de novo* generative models that unconditionally generate large numbers of stable, novel crystals are essential for screening campaigns to discover new materials with desirable properties. **ALM Gen** unlocks *de*

novo generation by relaxing the strong conditioning achieved by **ALM Edit**. The producer (Fig. 3) is replaced by a lightweight per-token MLP. Each of the K atomistic-token hidden states is projected independently into a K-token conditioning sequence, with no learned queries and no prompt-context window, feeding the same consumer as **Edit**. In addition, the LLM no longer produces compositions when prompted to generate structures (Appendix B.2.2). This is deliberately *weak* conditioning, causing text prompts to bias, rather than dictate, sampled structures. **ALM Gen** is trained on the same data as **Edit** in different concentrations (Appendix C.2.1), sampling input textual prompts at inference time from the respective eval partitions to generate crystals.

2.4 T2C-FK: Steering generation with Feynman–Kac

ALM Edit is designed to output a material with the desired element set and stoichiometry ratio. **ALM Gen**, on the other hand, is designed to produce more stable structures, but not necessarily with the exact stoichiometry used to prompt the model. This dichotomy arises from MatterGen's denoising, not the inherent ALM architecture (Appendix B.3.2). We introduce **Text-to-Crystal Feynman–Kac** (**T2C-FK**) to close this gap at inference time, intercepting and reweighting denoising trajectories without retraining.

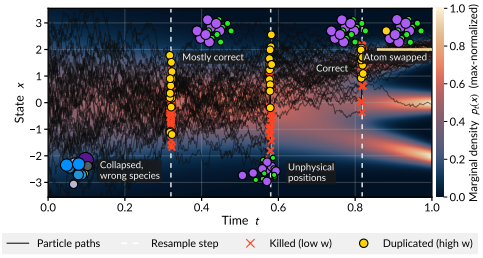

Figure 4: **Text-to-Crystal Feynman–Kac (T2C-FK) enables ALM Gen, a *de novo* model, to generate structures with desired element sets and stoichiometry ratios.** A. Unphysical structures are removed throughout sampling, and any differences from the reference stoichiometry are fixed at the last step via Hungarian scoring.

T2C-FK replaces MatterGen's single denoising trajectory with an N-particle bootstrap Sequential Monte Carlo sampler (Wu et al., 2023; Singhal et al., 2025): every S steps, it reweights and resamples particles by a reward on the Tweedie-estimated clean structure $\hat{x}_0$, deferring scoring until the atomic-number distribution leaves its high-noise regime. The reward scores stoichiometric agreement of the score network's per-atom element distributions to the target multiset, and a final Hungarian override snaps each atom to its assigned target element, leaving lattice and coordinates untouched. A real example of **T2C-FK** is shown in Fig. 4, and the full sampler (Algorithm 1) with its posterior-correction guarantee, three reward components, potential function, and hyperparameter sweeps is given in Appendix B.3.

3 Experiments

By instruction tuning across several tasks, Atomistic Language Models both natively understand materials and can lift their knowledge into text-conditional materials generation. **ALM Core** matches or beats unimodal predictors on property prediction, **ALM Edit** achieves state-of-the-art at crystal structure prediction, and **ALM Gen** beats even atomistic models' performance at *de novo* generation. **ALM Edit** also beats frontier language model baselines on several tasks in **ALM Bench**, designed to rigorously test text-conditioned materials optimization.

3.1 Increased property prediction performance and training token efficiency

On LLM4Mat-Bench (Rubungo et al., 2025), an extensive crystal property prediction benchmark, **ALM Core** is one of the first natural language models to outperform previously published unimodal graph neural network (GNN) baselines on several tasks (Table 19, Fig. 5A, C, D), breaking the so-called "GNN–LLM wall" (Rubungo et al., 2025). **ALM Core** also improves on previously published language model baselines, which are orders-of-magnitude smaller architectures finetuned on CIFs with poor reasoning and natural language skills (Rubungo et al., 2025). **ALM Core** is also competitive with GNNs and stronger than previous language model property predictors on the Mat2Props (Park et al., 2024) benchmark as well (Table 13).

MatterChat (Tang et al., 2026) is a recent method that also projects MLIP embeddings as soft tokens into LLMs for property prediction. When finetuned on 2 epochs of the MatterChat training data

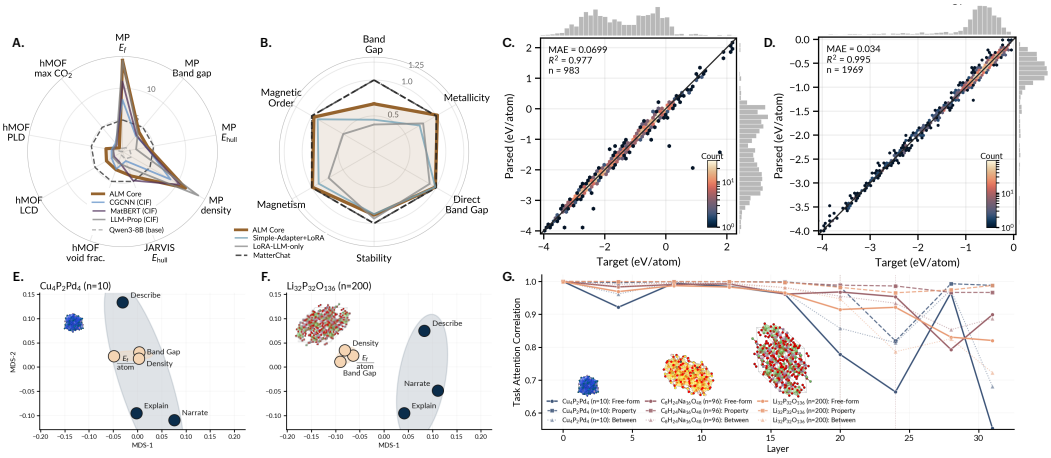

Figure 5: **Atomistic Language Models can accurately predict physical properties of materials.** Materials property prediction tasks from LLM4Mat-Bench (Rubungo et al., 2025; MAD/MAE, with a performance threshold of ≥ 5) in A. and MatterChat (MAE, baseline from and normalized to Tang et al. (2026)) in B. **Core** performs well at formation energy per atom (E_f) regression on both C. Materials Project (Jain et al., 2013) and D. GNoME (Merchant et al., 2023) data. Activations are similar across property-prediction (tan) and natural-language (dark blue) tasks for both E. small and F. large structures. G. shows attention scores per layer across tasks and materials sizes.

(after training on 20 epochs' worth of other instruction tuning data, relative to MatterChat's reported total 50 epochs), **ALM Core** matches MatterChat on $5/9$ tasks and beats it on 2 (Fig. 5B). Its underperformance on the other two tasks reflects the low LoRA rank chosen to preserve natural language skills and the limited chemical expressivity of the underlying atomistic encoder.

The language model backbone of **ALM Core** learns attention scores that depend on the question *and* chemical structure of the inputted material. For example, **ALM Core** attends to an inputted oxide structure's cations 1.5–2.5× more for a natural language output task compared to a property-prediction prompt. This asymmetry flips for intermetallic materials. Further, **Core** learns similar soft token activations among property prediction tasks and separately similar activations among free-form natural language tasks, like structure description and writing narratives about a material's properties and applications. This effect scales with input crystal size (Fig. 5E vs. F) and is exacerbated towards later transformer layers of the language model (Fig. 5G).

Table 1: **ALM retains zero-shot natural language capabilities and scientific knowledge despite multimodal finetuning.** The final, "Judge" column is an eval included in **ALM Bench**, measuring materials science knowledge retention as judged by GPT-4o (Appendix E). GPQA is from Rein et al. (2024).

Model	MMLU	GSM8K	GPQA	Judge
CrystalReasoner (Wu et al., 2026)	0.375	0.020	**0.289**	0.053
Qwen2.5-3B base	*0.550*	*0.310*	*0.221*	*0.816*
ALM Core	**0.595**	0.775	0.247	**0.921**
ALM Edit	0.485	**0.780**	0.228	**0.921**
Qwen3-8B base	*0.595*	*0.705*	*0.286*	*0.895*

A common failure mode of multimodal language models is catastrophically forgetting post-trained natural language skills (Liu et al., 2025). Many systems avoid this by training small contrastive heads on frozen LMs, paying for it with reduced cross-modal control, or accepting language degradation as a cost of integration (Liu et al., 2025). **ALM Core** resists catastrophic forgetting and requires neither trade-off by training on text-only CAMEL (Li et al., 2023a) scientific question-answering and JARVIS (Choudhary et al., 2020) materials science arXiv abstracts. **ALM Core** and **Edit** improve multi-step reasoning and arithmetic multiple-choice question skills, as well as free-form materials science knowledge as evaluated by a frontier LLM judge, compared to their base language model (Table 1). Other language-based crystal generative models (Wu et al., 2026) regress strongly on these tasks compared to their base models.

3.2 State-of-the-art crystal structure prediction performance

Table 2: **ALM Edit achieves SoTA performance at crystal structure prediction on MP-20 and MPTS-52.** Match rate MR (%, ↑) and RMSE (Å, ↓) to MP-20 and MPTS-52 test sets are scored at K=1 and best-of-K=20. Matcher, definitions, baselines, and conventions are available in Appendix E.2. **Bold** and underlined denote the best and second-best performance scores.

	MP-20				MPTS-52			
Model	MR@1 (%)↑	RMSE@1↓	MR@20 (%)↑	RMSE@20↓	MR@1 (%)↑	RMSE@1↓	MR@20 (%)↑	RMSE@20↓
CDVAE (Xie et al., 2022)	33.90	0.1045	66.95	0.1026	5.34	0.2106	20.79	0.2085
DiffCSP (Jiao et al., 2023)	51.49	0.0631	77.93	0.0492	12.19	0.1786	34.02	0.1749
FlowMM (Miller et al., 2024)	61.39	0.0566	—	—	17.54	0.1726	—	—
CrystaLLM-large (Antunes et al., 2024)	58.70	0.0408	73.97	0.0349	19.21	0.1110	33.75	0.1059
CrystalFlow (Luo et al., 2025)	62.02	0.0710	78.34	0.0577	21.00	0.1613	37.81	0.1584
OMatG (Höllmer et al., 2025)	63.75	0.0720	—	—	25.15	0.1931	—	—
MCFlow-L (Seong et al., 2026)	**64.08**	0.0561	76.08	0.0383	**27.16**	0.1401	41.45	0.1296
ALM Edit	45.6	**0.021**	**83.2**	**0.034**	22.7	**0.022**	**45.7**	**0.038**
ALM Gen + T2C-FK	22.3	0.025	41.0	0.012	6.0	0.040	10.0	0.011

Building on the performance and rich, aligned latent spaces of **ALM Core**, **ALM Edit** establishes a new state-of-the-art for unseen crystal structure prediction on MP-20, as well as MPTS-52, a significantly harder benchmark with structures over twice as large as MP-20 (Table 2). Although many of the target polymorphs are relatively energetically stable compared to other geometries, the exact choice defined by each dataset is somewhat arbitrarily set (Martirossyan et al., 2025). Therefore, models are over-penalized for generating physically realistic unit-cell doublings, global rotations, or other stable polymorphs. However, **ALM Edit** not only achieves SoTA RMSE, outputting geometrically similar polymorphs to dataset targets, but also high best-of-$K = 20$ match rates, indicating that it learns a valid distribution of polymorphs, one of which is likely to be the target polymorph set by the MP-20 and MPTS-52 evals. **ALM Edit** received the desired crystal's chemical composition *and symmetry space group* during training, but only the chemical composition at inference, helping the model learn a relationship between crystal symmetry groups expressed as text tokens and generated symmetric polymorphs. Table 2 also reports **ALM Gen** with FK enabled, which trades crystal structure prediction performance for increased crystal stability and novelty.

3.3 Unlocking text-guided materials optimization and inverse design with ALM Bench

ALM Edit unlocks the ability to *edit and generate materials as instructed by natural language*. We introduce **ALM Bench** to evaluate this capability, testing **Edit** and several frontier LLMs with thinking mode enabled on producing valid polymorphs of the inputted material with properties adjusted in a particular direction, e.g., "increase the formation energy of this crystal" (corresponding to "E_f ↑" below). **ALM Bench** also evaluates the model's one-shot ability to generate polymorphs of a given crystal with $E_{\text{hull}} < 0$ ("Polymorph"), as well as doping tasks, where models are prompted to dope a given crystal with a new element. The structural and compositional match of the generated and true doped crystal ("Doping"), as well as the strain of each ("Strain"), score success at this task (Appendix D).

Table 3: **ALM Bench evaluates models on atomistic editing tasks as guided by language.** Directional editing per property (E_f, ρ, V) and direction (↑/↓) are indicated (N=5×200). OpenAI models were prompted to generate CIFs.

Model	E_f ↑	E_f ↓	ρ ↑	ρ ↓	V ↑	V ↓	Polymorph	Doping	Strain
ALM Edit	**0.586** ±0.024	0.626 ±0.026	**0.380** ±0.011	**0.370** ±0.040	**0.451** ±0.059	**0.355** ±0.033	0.224 ±0.022	**0.879** ±0.012	**0.151** ±0.028
GPT-4.1	0.465 ±0.032	0.496 ±0.025	0.007 ±0.009	0.239 ±0.034	0.276 ±0.035	0.040 ±0.016	0.083 ±0.013	0.003 ±0.004	0.000 ±0.000
GPT-5.2	0.437 ±0.073	0.414 ±0.042	0.058 ±0.026	0.244 ±0.019	0.006 ±0.005	0.032 ±0.015	0.118 ±0.023	0.002 ±0.002	0.000 ±0.000
GPT-5.6-sol	0.513 ±0.046	**0.636** ±0.032	0.220 ±0.019	0.238 ±0.019	0.050 ±0.021	0.028 ±0.005	**0.444** ±0.043	0.004 ±0.002	0.000 ±0.000

As shown in Table 3, **ALM Edit** produces valid edits to crystal geometry and structure, as each **ALM Bench** metric scores structurally invalid generations, trivial lattice rescalings, and unphysical atom relabelings as failures. GPT-5.6 Sol beats **Edit** at "E_f ↓" and "Polymorph" by generating memorized ground states of requested materials.

Table 4: **ALM Bench also evaluates models on crystal generation tasks.** Consistency of generated materials (N=5×200) to requested application area (LLM Judge-assessed), materials description, and adversarial composition input (Appendix D).

Model	Application	Describe (Comp.)	Describe (Struct.)	OOD (Comp.)	OOD (Struct.)
ALM Edit	**0.323** ±0.020	**0.730** ±0.031	0.412 ±0.041	0.641 ±0.032	0.316 ±0.022
GPT-4.1	0.224 ±0.032	0.254 ±0.019	0.090 ±0.018	0.228 ±0.029	0.050 ±0.005
GPT-5.2	0.252 ±0.051	0.356 ±0.014	0.162 ±0.013	0.366 ±0.026	0.108 ±0.008
GPT-5.6-sol	0.263 ±0.067	0.728 ±0.022	**0.479** ±0.004	**0.696** ±0.030	**0.357** ±0.026

ALM Bench also evaluates models on text-conditioned generation, including asking for a crystal that belongs to an "Application" area (e.g., "generate a perovskite") and difficult crystal structure prediction prompts, ranging from long narratives about properties and structure ("Describe") to adversarially designed, terse prompts ("OOD", e.g. "generate MgO structure at 3.58 g/cm^3"). Although **ALM Edit** leads or is second best on every task (Table 4), it loses to 5.6 Sol when prompted about materials and applications it has not been post-trained on, reflecting a deficiency in the underlying Qwen3-8B language model, not in the ALM architecture (Appendix D.7).

3.4 Competitive *de novo* generation for stable crystal generation

To enable large-throughput screening campaigns of *de novo* generated crystals, we deliberately relax **ALM Edit**'s strong conditioning into **ALM Gen** (§2.3), in which prompts bias the distribution of sampled structures without dictating each sample. These prompts are drawn uniformly from the eval set of prompts for **Edit**. **ALM Gen** achieves state-of-the-art performance for simultaneously stable, unique, and novel (SUN) structures, where stability is measured by $E_{\text{hull}} < 0.016$ on the MP-20 hull (Jain et al., 2013). Crucially, turning guidance on to $g = 0.5$ and steering $\mathcal{D}$ using natural language *improves* generation quality over the $g = 0$ MatterGen base (Table 5), highlighting that learned language model priors improve upon unimodal, atomistic generation. On the harder, broader-chemistry LeMat-GenBench (Betala et al., 2026) protocol (Table 21), it tops the field on metastable ($E_{\text{hull}} < 0.1$) yield while being second to the *de novo*-specialist flow models at strict SUN, where strict stability is defined as $E_{\text{hull}} < 0$. Metastable yield on MP-20 is reported in full in Appendix E.3.

Table 5: ***De novo* generation against the MP-20 hull** (Wu et al., 2026), in which stability S is defined by $E_{\text{hull}} \leq 0.016$ eV/atom and structures are pre-relaxed. Evaluated on $N = 10 \times 1000$ samples (Appendix E.3).

Method	E_{hull} (eV)↓	U (%)↑	V_{struct} (%)↑	V_{chem} (%)↑	**SUN** (%)↑
CrystalTextLLM (Gruver et al., 2024)	0.61 ±0.003	47.40 ±0.30	90.01 ±0.21	91.59 ±0.05	0.38 ±0.05
PLAID++ Wyckoff (Xu et al., 2026)	0.57 ±0.003	40.70 ±0.30	89.06 ±0.21	91.59 ±0.04	0.50 ±0.05
CrysReas-Base (SFT only)	0.58 ±0.004	35.25 ±0.31	84.03 ±0.23	90.36 ±0.10	0.57 ±0.05
CrysReas-Thinking (SFT+CoT)	0.52 ±0.003	38.64 ±0.29	91.29 ±0.19	91.72 ±0.04	0.59 ±0.06
CrysReas-RL (SFT+GRPO)	0.53 ±0.003	82.49 ±0.19	89.85 ±0.20	91.10 ±0.07	1.23 ±0.07
CrysReas (Wu et al., 2026)	0.45 ±0.003	87.23 ±0.14	94.92 ±0.15	**91.78** ±0.03	1.70 ±0.08
MatterGen (Base)	**0.079** ±0.002	93.50 ±0.90	**100.00** ±0.00	86.50 ±2.90	5.53 ±1.17
ALM Gen	0.085 ±0.003	**98.90** ±0.70	**100.00** ±0.00	83.20 ±2.90	**7.80** ±0.44
ALM Gen + FK-stoich	0.086 ±0.005	73.80 ±2.20	**100.00** ±0.00	84.50 ±1.00	5.21 ±1.18

4 Discussion

Atomistic Language Models are, to our knowledge, the first natively multimodal models to unify atomistic understanding with text-instructed, materials-conditioned generation through continuous latent bridges, resolving the three failure modes of §1: continuous representations break the "GNN–LLM wall" and recover GNN-level property prediction (§3.1); projectors and a late-layer task-mode classifier expose a rich joint latent space (§3.1); and the language-to-atomistic bridge

steers the decoder in that space (§3.3). **ALM Bench** will be released to evaluate this ability and invite new architectures.

ALM Edit shows that language models can guide materials generation through last-layer hidden-state tokens, though the architecture and recipe that maximize this steering remain open. Across bridge ablations (Appendix B.2.5), only the proposed bridge amplified task and structural signal enough to guide generation, and the signal is highly sensitive to the teacher-forcing order of the atomistic tokens (Fig. 16). A different decoder would remove the need for two (Appendix B.2.2).

T2C-FK, an inference-time steering layer, bridges **ALM Gen**'s stability and **ALM Edit**'s instruction-following. Being decoder-agnostic, it extends beyond stoichiometry to any constraint expressible as a per-step reward, including charge balance and cell symmetry. Applied to **ALM Edit**'s formation-energy editing with a MatterSim energy reward, it *improved* $E_f \downarrow$ success to 72.5% and $E_f \downarrow$ to 71.3%.

Language model capability is driven by scaling. *Accordingly, ALM property prediction scales with language model size*: Fig. 6A shows performance rising monotonically with parameter count (further properties in Fig. 18), a promising sign that more capabilities may emerge with data and model scale (Wei et al., 2022).

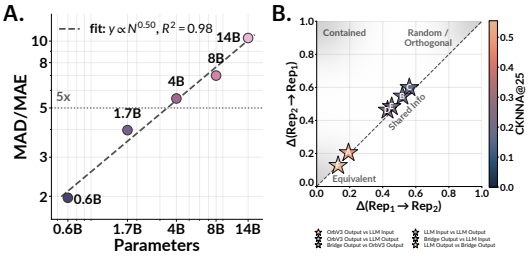

Figure 6: **Strong scaling emerges under fixed training and evaluation for several property prediction tasks.** A. For increasing Qwen3 model size, property prediction performance improves monotonically in MAD/MAE on LLM4Mat-Bench. B. Representational analysis of embeddings extracted from each continuous latent space throughout **ALM Edit** for 2,000 prompts from **ALM Bench**.

By learning continuous latent spaces across every modality within a single architecture, Atomistic Language Modeling makes representational alignment measurable without an artificial contrastive objective. Information imbalance (Glielmo et al., 2022), a global similarity metric over sets of embeddings, shows that **ALM Edit**'s atomistic-to-language adapter and language-to-atomistic bridge preserve the information content they translate (Fig. 6B), and that the MLIP's predictive embeddings, the language model's atomistic-token embeddings, and the diffusion decoder's steering vectors share substantially more information with one another than independently trained language and atomistic models do (Edamadaka et al., 2025). No pair reaches informational equivalence, though: each latent space keeps information the others lack.

Several limitations frame **ALM Core, Edit,** and **Gen** as starting points. ALMs learn in continuous latent spaces; discretizing a rich atomistic space into a learned codebook, as in VQ-VAE (van den Oord et al., 2017) or JANUS (Chen et al., 2025), is an alternate design for simple inorganic crystals. ALMs are trained only on inorganic crystals but extend in principle to any atomistic system by swapping in a more foundational encoder such as UMA or PET-MAD, or a less locally biased one (Kreiman et al., 2025; Appendix B.1.2); their continuous interface may be especially valuable for large, amorphous, or defect-ridden systems that are more compactly described in language than all-atom form. Finally, thinking was disabled throughout for lack of chain-of-thought data; building reasoning traces from tool-assisted frontier models and RL-finetuning ALMs is a promising direction (Wu et al., 2026).

5 Conclusion

Atomistic Language Modeling unifies a pretrained atomistic encoder, language model, and denoising diffusion decoder into shared latent spaces through continuous projectors. **ALM Core** matches atomistic graph networks at property prediction without losing language ability; **ALM Edit** sets a new crystal-structure-prediction SoTA (Match@K=20 and RMSE on MP-20 and MPTS-52) and optimizes crystals from natural-language prompts; and **ALM Gen** generates SUN crystals at higher rates than prior atomistic and language models, with **T2C-FK** steering it toward stoichiometric targets at inference time, establishing a scalable paradigm for materials discovery steered by natural language.

AI USE STATEMENT

Generative AI tools have not been used in our final manuscript. Although initial prototyping and figure generation code was generated with coding agents, all code was re-written and bug-checked by hand by two different people. Therefore, all results herein are not generated by any generative AI tools.

REPRODUCIBILITY STATEMENT

All code is submitted as supplementary material, and all code, data, and model checkpoints will be released publicly with the camera-ready draft. All results are reported with error bars or 95% confidence intervals over multiple seeds wherever possible (e.g., **ALM Bench** is scored over 5 seeds $\times$ 200 prompts per task). Every metric, including property prediction, crystal structure prediction, *de novo* generation, and representational alignment, is defined in full in Appendix E, together with the structure matchers and stability thresholds each uses. The exact positioning of this work relative to prior single-modality, system-level multimodal, and inference-time-control approaches is given in the related work (Appendix A). Each component of the architecture, namely the atomistic encoder and projector, the producer–consumer bridge, the language-model finetuning for steering, and the T2C-FK sampler, is detailed with its ablations in Appendix B, and the training data and mixtures for every stage are described in Appendix C. **ALM Bench** is released for the community, and each of its 11 tasks and their scoring are specified in Appendix D.

REFERENCES

Nawaf Alampara, Santiago Miret, and Kevin Maik Jablonka. Mattext: Do language models need more than text & scale for materials modeling?, 2024. URL https://arxiv.org/abs/2406.17295v1. arXiv:2406.17295; v3 (2025) retitled "Less can be more for predicting properties with large language models".

Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katie Millican, Malcolm Reynolds, Roman Ring, Eliza Rutherford, Serkan Cabi, Tengda Han, Zhitao Gong, Sina Samangooei, Marianne Monteiro, Jacob Menick, Sebastian Borgeaud, Andrew Brock, Aida Nematzadeh, Sahand Sharifzadeh, Mikolaj Binkowski, Ricardo Barreira, Oriol Vinyals, Andrew Zisserman, and Karen Simonyan. Flamingo: a visual language model for few-shot learning. In *Advances in Neural Information Processing Systems 35 (NeurIPS 2022)*, 2022. arXiv:2204.14198.

Luis M. Antunes, Keith T. Butler, and Ricardo Grau-Crespo. Crystal structure generation with autoregressive large language modeling. *Nature Communications*, 15(1):10570, December 2024. ISSN 2041-1723. doi: 10.1038/s41467-024-54639-7. URL https://www.nature.com/articles/s41467-024-54639-7.

Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, pp. 17981–17993, 2021. arXiv:2107.03006.

Ilyes Batatia, Philipp Benner, Yuan Chiang, Alin M. Elena, Dávid P. Kovács, Janosh Riebesell, Xavier R. Advincula, Mark Asta, Matthew Avaylon, William J. Baldwin, Fabian Berger, Noam Bernstein, Arghya Bhowmik, Filippo Bigi, Samuel M. Blau, Vlad Cărare, Michele Ceriotti, Sanggyu Chong, James P. Darby, Sandip De, Flaviano Della Pia, Volker L. Deringer, Rokas Elijošius, Zakariya El-Machachi, Edvin Fako, Fabio Falcioni, Andrea C. Ferrari, John L. A. Gardner, Mikolaj J. Gawkowski, Annalena Genreith-Schriever, Janine George, Rhys E. A. Goodall, Jonas Grandel, Clare P. Grey, Petr Grigorev, Shuang Han, Will Handley, Hendrik H. Heenen, Kersti Hermansson, Cheuk Hin Ho, Stephan Hofmann, Christian Holm, Jad Jaafar, Konstantin S. Jakob, Hyunwook Jung, Venkat Kapil, Aaron D. Kaplan, Nima Karimitari, James R. Kermode, Panagiotis Kourtis, Namu Kroupa, Jolla Kullgren, Matthew C. Kuner, Domantas Kuryla, Guoda Liepuoniute, Chen Lin, Johannes T. Margraf, Ioan-Bogdan Magdău, Angelos Michaelides, J. Harry Moore, Aakash A. Naik, Samuel P. Niblett, Sam Walton Norwood, Niamh O'Neill, Christoph Ortner, Kristin A. Persson, Karsten Reuter, Andrew S. Rosen, Louise A. M. Rosset, Lars L. Schaaf, Christoph Schran, Benjamin X. Shi, Eric Sivonxay, Tamás K. Stenczel,

Christopher Sutton, Viktor Svahn, Thomas D. Swinburne, Jules Tilly, Cas van der Oord, Santiago Vargas, Eszter Varga-Umbrich, Tejs Vegge, Martin Vondrák, Yangshuai Wang, William C. Witt, Thomas Wolf, Fabian Zills, and Gábor Csányi. A foundation model for atomistic materials chemistry. *The Journal of Chemical Physics*, 163(18):184102, 2025. doi: 10.1063/5.0297006. URL https://doi.org/10.1063/5.0297006. arXiv:2401.00096.

Siddharth Betala, Samuel P. Gleason, Ali Ramlaoui, Andy Xu, Georgia Channing, Daniel Levy, Clémentine Fourrier, Nikita Kazeev, Chaitanya K. Joshi, Sékou-Oumar Kaba, Félix Therrien, Alex Hernandez-Garcia, Rocío Mercado, N. M. Anoop Krishnan, and Alexandre Duval. Lemat-genbench: A unified evaluation framework for crystal generative models, 2026. URL https://arxiv.org/abs/2512.04562.

Onur Boyar, Indra Priyadarsini, Seiji Takeda, and Lisa Hamada. Llm-fusion: A novel multimodal fusion model for accelerated material discovery, 2025. URL https://arxiv.org/abs/2503.01022.

Xiaokang Chen, Zhiyu Wu, Xingchao Liu, Zizheng Pan, Wen Liu, Zhenda Xie, Xingkai Yu, and Chong Ruan. Janus-pro: Unified multimodal understanding and generation with data and model scaling, 2025. URL https://arxiv.org/abs/2501.17811.

Kamal Choudhary, Kevin F. Garrity, Andrew C. E. Reid, Brian DeCost, Adam J. Biacchi, Angela R. Hight Walker, Zachary Trautt, Jason Hattrick-Simpers, A. Gilad Kusne, Andrea Centrone, Albert Davydov, Jie Jiang, Ruth Pachter, Gowoon Cheon, Evan Reed, Ankit Agrawal, Xiaofeng Qian, Vinit Sharma, Houlong Zhuang, Sergei V. Kalinin, Bobby G. Sumpter, Ghanshyam Pilania, Pinar Acar, Subhasish Mandal, Kristjan Haule, David Vanderbilt, Karin Rabe, and Francesca Tavazza. The joint automated repository for various integrated simulations (jarvis) for data-driven materials design. *npj Computational Materials*, 6(1):173, November 2020. ISSN 2057-3960. doi: 10.1038/s41524-020-00440-1. URL https://www.nature.com/articles/s41524-020-00440-1.

Jiyu Cui, Fang Wu, Haokai Zhao, Minggao Feng, Xenophon Evangelopoulos, Andrew I. Cooper, and Yejin Choi. $L^2m^3$of: A large language multimodal model for metal-organic frameworks, 2025. URL https://arxiv.org/abs/2510.20976.

Bowen Deng, Bohan Li, Matthew Cox, Hoje Chun, Juno Nam, Artur Lyssenko, Sathya Edamadaka, Jurgis Ruza, Xiaochen Du, Nofit Segal, Jesus Diaz Sanchez, Mingrou Xie, Ty Perez, Yu Yao, Miguel Steiner, Sauradeep Majumdar, Charles B. Musgrave III, Anirban Chandra, Abhirup Patra, Detlef Hohl, Connor W. Coley, Ju Li, and Rafael Gómez-Bombarelli. Harnessing atomisticskills for agentic atomistic research, 2026. URL https://arxiv.org/abs/2605.24002.

Prafulla Dhariwal and Alexander Quinn Nichol. Diffusion models beat gans on image synthesis. In *Advances in Neural Information Processing Systems 34 (NeurIPS 2021)*, pp. 8780–8794, 2021. arXiv:2105.05233.

Ralf Drautz. Atomic cluster expansion for accurate and transferable interatomic potentials. *Physical Review B*, 99(1):014104, January 2019. doi: 10.1103/PhysRevB.99.014104. URL https://link.aps.org/doi/10.1103/PhysRevB.99.014104.

Sathya Edamadaka, Soojung Yang, Ju Li, and Rafael Gómez-Bombarelli. Universally converging representations of matter across scientific foundation models, 2025. URL https://arxiv.org/abs/2512.03750.

Carl Edwards, Tuan Lai, Kevin Ros, Garrett Honke, Kyunghyun Cho, and Heng Ji. Translation between molecules and natural language. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 375–413, 2022. doi: 10.18653/v1/2022.emnlp-main.26. URL https://doi.org/10.18653/v1/2022.emnlp-main.26. arXiv:2204.11817.

Alex M. Ganose and Anubhav Jain. Robocrystallographer: automated crystal structure text descriptions and analysis. *MRS Communications*, 9(3):874–881, September 2019. ISSN 2159-6859. doi: 10.1557/mrc.2019.94. URL https://doi.org/10.1557/mrc.2019.94.

Aldo Glielmo, Claudio Zeni, Bingqing Cheng, Gábor Csányi, and Alessandro Laio. Ranking the information content of distance measures. *PNAS Nexus*, 1(2):pgac039, 05 2022. ISSN 2752-6542. doi: 10.1093/pnasnexus/pgac039. URL https://doi.org/10.1093/pnasnexus/pgac039.

Nate Gruver, Anuroop Sriram, Andrea Madotto, Andrew Gordon Wilson, C. Lawrence Zitnick, and Zachary Ulissi. Fine-tuned language models generate stable inorganic materials as text. In *International Conference on Learning Representations (ICLR)*, 2024. arXiv:2402.04379.

Tanishq Gupta, Mohd Zaki, N. M. Anoop Krishnan, and Mausam. Matscibert: A materials domain language model for text mining and information extraction. *npj Computational Materials*, 8 (1):102, 2022. doi: 10.1038/s41524-022-00784-w. URL https://doi.org/10.1038/s41524-022-00784-w. arXiv:2109.15290.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance, 2022. URL https://arxiv.org/abs/2207.12598.

Philipp Höllmer, Thomas Egg, Maya M. Martirossyan, Eric Fuemmeler, Zeren Shui, Amit Gupta, Pawan Prakash, Adrian Roitberg, Mingjie Liu, George Karypis, Mark Transtrum, Richard G. Hennig, Ellad B. Tadmor, and Stefano Martiniani. Open materials generation with stochastic interpolants. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, pp. 23417–23450, 2025. arXiv:2502.02582.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9. arXiv:2106.09685.

Minyoung Huh, Brian Cheung, Tongzhou Wang, and Phillip Isola. Position: The platonic representation hypothesis. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*, volume 235 of *Proceedings of Machine Learning Research*, pp. 20617–20642. PMLR, 2024. URL https://proceedings.mlr.press/v235/huh24a.html. arXiv:2405.07987.

Andrew Jaegle, Sebastian Borgeaud, Jean-Baptiste Alayrac, Carl Doersch, Catalin Ionescu, David Ding, Skanda Koppula, Daniel Zoran, Andrew Brock, Evan Shelhamer, Olivier J. Hénaff, Matthew M. Botvinick, Andrew Zisserman, Oriol Vinyals, and João Carreira. Perceiver io: A general architecture for structured inputs & outputs. In *International Conference on Learning Representations (ICLR)*, 2022. arXiv:2107.14795.

Anubhav Jain, Shyue Ping Ong, Geoffroy Hautier, Wei Chen, William Davidson Richards, Stephen Dacek, Shreyas Cholia, Dan Gunter, David Skinner, Gerbrand Ceder, and Kristin A. Persson. Commentary: The materials project: A materials genome approach to accelerating materials innovation. *APL Materials*, 1(1):011002, 2013. ISSN 2166-532X. doi: 10.1063/1.4812323. URL https://doi.org/10.1063/1.4812323.

Rui Jiao, Wenbing Huang, Peijia Lin, Jiaqi Han, Pin Chen, Yutong Lu, and Yang Liu. Crystal structure prediction by joint equivariant diffusion. In *Advances in Neural Information Processing Systems 36 (NeurIPS 2023)*, 2023. arXiv:2309.04475.

Nikita Kazeev, Wei Nong, Ignat Romanov, Ruiming Zhu, Andrey Ustyuzhanin, Shuya Yamazaki, and Kedar Hippalgaonkar. Wyckoff transformer: Generation of symmetric crystals. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, pp. 29495–29526, 2025. arXiv:2503.02407.

Hyunseob Kim, Jeongcheol Lee, Sunil Ahn, and Jongsuk Ruth Lee. A merged molecular representation learning for molecular properties prediction with a web-based service. *Scientific Reports*, 11(1):11028, May 2021. ISSN 2045-2322. doi: 10.1038/s41598-021-90259-7. URL https://www.nature.com/articles/s41598-021-90259-7.

Jing Yu Koh, Daniel Fried, and Ruslan Salakhutdinov. Generating images with multimodal language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, 2023. arXiv:2305.17216.

Tobias Kreiman, Yutong Bai, Fadi Atieh, Elizabeth Weaver, Eric Qu, and Aditi S. Krishnapriyan. Transformers discover molecular structure without graph priors, 2025. URL https://arxiv.org/abs/2510.02259.

Mario Krenn, Florian Häse, AkshatKumar Nigam, Pascal Friederich, and Alán Aspuru-Guzik. Self-referencing embedded strings (selfies): A 100% robust molecular string representation. *Machine Learning: Science and Technology*, 1(4):045024, December 2020. ISSN 2632-2153. doi: 10.1088/2632-2153/aba947. URL https://doi.org/10.1088/2632-2153/aba947. arXiv:1905.13741.

Guohao Li, Hasan Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. Camel: Communicative agents for "mind" exploration of large language model society. In *Advances in Neural Information Processing Systems 36 (NeurIPS 2023)*, 2023a. URL http://papers.nips.cc/paper_files/paper/2023/hash/a3621ee907def47c1b952ade25c67698-Abstract-Conference.html. arXiv:2303.17760.

Junnan Li, Dongxu Li, Silvio Savarese, and Steven C. H. Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pp. 19730–19742. PMLR, 2023b. URL https://proceedings.mlr.press/v202/li23q.html. arXiv:2301.12597.

Sihang Li, Zhiyuan Liu, Yanchen Luo, Xiang Wang, Xiangnan He, Kenji Kawaguchi, Tat-Seng Chua, and Qi Tian. Towards 3d molecule-text interpretation in language models. In *International Conference on Learning Representations (ICLR)*, 2024. URL https://openreview.net/forum?id=xI4yNlkaqh. arXiv:2401.13923.

Alexander H. Liu, Andy Ehrenberg, Andy Lo, Clément Denoix, Corentin Barreau, Guillaume Lample, Jean-Malo Delignon, Khyathi Raghavi Chandu, Patrick von Platen, Pavankumar Reddy Muddireddy, Sanchit Gandhi, Soham Ghosh, Srijan Mishra, Thomas Foubert, Abhinav Rastogi, Adam Yang, Albert Q. Jiang, Alexandre Sablayrolles, Amélie Héliou, Amélie Martin, Anmol Agarwal, Antoine Roux, Arthur Darcet, Arthur Mensch, Baptiste Bout, Baptiste Rozière, Baudouin De Monicault, Chris Bamford, Christian Wallenwein, Christophe Renaudin, Clémence Lanfranchi, Darius Dabert, Devendra Singh Chaplot, Devon Mizelle, Diego de las Casas, Elliot Chane-Sane, Emilien Fugier, Emma Bou Hanna, Gabrielle Berrada, Gauthier Delerce, Gauthier Guinet, Georgii Novikov, Guillaume Martin, Himanshu Jaju, Jan Ludziejewski, Jason Rute, Jean-Hadrien Chabran, Jessica Chudnovsky, Joachim Studnia, Joep Barmentlo, Jonas Amar, Josselin Somerville Roberts, Julien Denize, Karan Saxena, Karmesh Yadav, Kartik Khandelwal, Kush Jain, Lélio Renard Lavaud, Léonard Blier, Lingxiao Zhao, Louis Martin, Lucile Saulnier, Luyu Gao, Marie Pellat, Mathilde Guillaumin, Mathis Felardos, Matthieu Dinot, Maxime Darrin, Maximilian Augustin, Mickaël Seznec, Neha Gupta, Nikhil Raghuraman, Olivier Duchenne, Patricia Wang, Patryk Saffer, Paul Jacob, Paul Wambergue, Paula Kurylowicz, Philomène Chagniot, Pierre Stock, Pravesh Agrawal, Rémi Delacourt, Romain Sauvestre, Roman Soletskyi, Sagar Vaze, Sandeep Subramanian, Saurabh Garg, Shashwat Dalal, Siddharth Gandhi, Sumukh Aithal, Szymon Antoniak, Teven Le Scao, Thibault Schueller, Thibaut Lavril, Thomas Robert, Thomas Wang, Timothée Lacroix, Tom Bewley, Valeriia Nemychnikova, Victor Paltz, Virgile Richard, Wen-Ding Li, William Marshall, Xuanyu Zhang, Yihan Wan, and Yunhao Tang. Voxtral, 2025. URL https://arxiv.org/abs/2507.13264.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *Advances in Neural Information Processing Systems 36 (NeurIPS 2023)*, 2023a. arXiv:2304.08485.

Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 26296–26306, 2024. arXiv:2310.03744.

Nian Liu, Nikita Kazeev, Stephen Gregory Dale, Artem Maevskiy, Yuwei Zeng, Ryoji Kubo, Pengru Huang, Thomas Laurent, Yann LeCun, Kostya S. Novoselov, and Xavier Bresson. Crys-jepa: Accelerating crystal discovery via embedding screening and generative refinement, 2026. URL https://arxiv.org/abs/2605.14759.

Shengchao Liu, Weili Nie, Chengpeng Wang, Jiarui Lu, Zhuoran Qiao, Ling Liu, Jian Tang, Chaowei Xiao, and Animashree Anandkumar. Multi-modal molecule structure–text model for text-based retrieval and editing. *Nature Machine Intelligence*, 5(12):1447–1457, 2023b. doi: 10.1038/s42256-023-00759-6. URL https://doi.org/10.1038/s42256-023-00759-6. arXiv:2212.10789.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*, 2019. arXiv:1711.05101.

Chris Lu, Cong Lu, Robert Tjarko Lange, Yutaro Yamada, Shengran Hu, Jakob Foerster, David Ha, and Jeff Clune. Towards end-to-end automation of ai research. *Nature*, 651(8107):914–919, March 2026. ISSN 1476-4687. doi: 10.1038/s41586-026-10265-5. URL https://www.nature.com/articles/s41586-026-10265-5.

Xiaoshan Luo, Zhenyu Wang, Qingchang Wang, Xuechen Shao, Jian Lv, Lei Wang, Yanchao Wang, and Yanming Ma. Crystalflow: a flow-based generative model for crystalline materials. *Nature Communications*, 16(1):9267, 2025. doi: 10.1038/s41467-025-64364-4. URL https://doi.org/10.1038/s41467-025-64364-4. arXiv:2412.11693.

Maya M. Martirossyan, Thomas Egg, Philipp Hoellmer, George Karypis, Mark Transtrum, Adrian Roitberg, Mingjie Liu, Richard G. Hennig, Ellad B. Tadmor, and Stefano Martiniani. All that structure matches does not glitter. In *Advances in Neural Information Processing Systems 39 (NeurIPS 2025) Datasets and Benchmarks Track*, 2025. arXiv:2509.12178.

Fabian Mentzer, David Minnen, Eirikur Agustsson, and Michael Tschannen. Finite scalar quantization: Vq-vae made simple. In *International Conference on Learning Representations (ICLR)*, 2024. arXiv:2309.15505.

Amil Merchant, Simon Batzner, Samuel S. Schoenholz, Muratahan Aykol, Gowoon Cheon, and Ekin Dogus Cubuk. Scaling deep learning for materials discovery. *Nature*, 624(7990):80–85, December 2023. ISSN 1476-4687. doi: 10.1038/s41586-023-06735-9. URL https://www.nature.com/articles/s41586-023-06735-9.

Benjamin Kurt Miller, Ricky T. Q. Chen, Anuroop Sriram, and Brandon M. Wood. Flowmm: Generating materials with riemannian flow matching. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, pp. 35664–35686, 2024. arXiv:2406.04713.

Viggo Moro, Charlotte Loh, Rumen Dangovski, Ali Ghorashi, Andrew Ma, Zhuo Chen, Samuel Kim, Peter Y. Lu, Thomas Christensen, and Marin Soljačić. Multimodal foundation models for material property prediction and discovery. *Newton*, 1(1):100016, March 2025. ISSN 2950-6360. doi: 10.1016/j.newton.2025.100016. URL https://www.cell.com/newton/abstract/S2950-6360(25)00008-8.

Andre Niyongabo Rubungo, Craig Arnold, Barry P. Rand, and Adji Bousso Dieng. Llm-prop: predicting the properties of crystalline materials using large language models. *npj Computational Materials*, 11(1):186, June 2025. ISSN 2057-3960. doi: 10.1038/s41524-025-01536-2. URL https://www.nature.com/articles/s41524-025-01536-2.

Keisuke Ozawa, Teppei Suzuki, Shunsuke Tonogai, and Tomoya Itakura. Graph-text contrastive learning of inorganic crystal structure toward a foundation model of inorganic materials. *Science and Technology of Advanced Materials: Methods*, 4(1):2406219, December 2024. doi: 10.1080/27660400.2024.2406219. URL https://doi.org/10.1080/27660400.2024.2406219.

Yang Jeong Park, Sung Eun Jerng, Sungroh Yoon, and Ju Li. 1.5 million materials narratives generated by chatbots. *Scientific Data*, 11(1):1060, September 2024. ISSN 2052-4463. doi: 10.1038/s41597-024-03886-w. URL https://www.nature.com/articles/s41597-024-03886-w.

Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. Film: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

Chen Qian, Huayi Tang, Zhirui Yang, Hong Liang, and Yong Liu. Can large language models empower molecular property prediction?, 2023. URL https://arxiv.org/abs/2307.07443.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. Gpqa: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling (COLM)*, 2024. arXiv:2311.12022.

Benjamin Rhodes, Sander Vandenhaute, Vaidotas Šimkus, James Gin, Jonathan Godwin, Tim Duignan, and Mark Neumann. Orb-v3: atomistic simulation at scale, 2025. URL https://arxiv.org/abs/2504.06231.

Andre Niyongabo Rubungo, Kangming Li, Jason Hattrick-Simpers, and Adji Bousso Dieng. Llm4mat-bench: benchmarking large language models for materials property prediction. *Machine Learning: Science and Technology*, 6(2):020501, 2025. doi: 10.1088/2632-2153/add3bb. URL https://doi.org/10.1088/2632-2153/add3bb. arXiv:2411.00177.

Kiyoung Seong, Sungsoo Ahn, Sehui Han, and Changyoung Park. Multimodal crystal flow: Any-to-any modality generation for unified crystal modeling, 2026. URL https://arxiv.org/abs/2602.20210.

Raghav Singhal, Zachary Horvitz, Ryan Teehan, Mengye Ren, Zhou Yu, Kathleen McKeown, and Rajesh Ranganath. A general framework for inference-time scaling and steering of diffusion models. In *Proceedings of the 42nd International Conference on Machine Learning (ICML 2025), PMLR 267*, pp. 55810–55827, 2025. arXiv:2501.06848.

Yuta Suzuki, Tatsunori Taniai, Ryo Igarashi, Kotaro Saito, Naoya Chiba, Yoshitaka Ushiku, and Kanta Ono. Bridging text and crystal structures: Literature-driven contrastive learning for materials science. *Machine Learning: Science and Technology*, 6(3):035006, September 2025. ISSN 2632-2153. doi: 10.1088/2632-2153/ade58c. URL https://doi.org/10.1088/2632-2153/ade58c. arXiv:2501.12919.

Yingheng Tang, Wenbin Xu, Jie Cao, Weilu Gao, Steven Farrell, Benjamin Erichson, Michael W. Mahoney, Andy Nonaka, and Zhi Jackie Yao. A multimodal large language model for materials science. *Nature Machine Intelligence*, 8(4):588–601, April 2026. ISSN 2522-5839. doi: 10.1038/s42256-026-01214-y. URL https://www.nature.com/articles/s42256-026-01214-y.

Aäron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural discrete representation learning. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pp. 6306–6315, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/7a98af17e63a0ac09ce2e96d03992fbc-Abstract.html. arXiv:1711.00937.

Xierui Wang, Siming Fu, Qihan Huang, Wanggui He, and Hao Jiang. Ms-diffusion: Multi-subject zero-shot image personalization with layout guidance. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025. URL https://openreview.net/forum?id=PJqP0wyQek. arXiv:2406.07209.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. Emergent abilities of large language models, 2022. URL https://arxiv.org/abs/2206.07682.

David Weininger. Smiles, a chemical language and information system. 1. introduction to methodology and encoding rules. *Journal of Chemical Information and Computer Sciences*, 28(1):31–36, February 1988. ISSN 0095-2338. doi: 10.1021/ci00057a005. URL https://pubs.acs.org/doi/abs/10.1021/ci00057a005.

Brandon M. Wood, Misko Dzamba, Xiang Fu, Meng Gao, Muhammed Shuaibi, Luis Barroso-Luque, Kareem Abdelmaqsoud, Vahe Gharakhanyan, John R. Kitchin, Daniel S. Levine, Kyle Michel, Anuroop Sriram, Taco Cohen, Abhishek Das, Ammar Rizvi, Sushree Jagriti Sahoo, Zachary W. Ulissi, and C. Lawrence Zitnick. Uma: A family of universal models for atoms, 2026. URL https://arxiv.org/abs/2506.23971.

Luhuan Wu, Brian L. Trippe, Christian A. Naesseth, David M. Blei, and John P. Cunningham. Practical and asymptotically exact conditional sampling in diffusion models. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10-16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/63e8bc7bbf1cfea36d1d1b6538aecce5-Abstract-Conference.html. arXiv:2306.17775.

Yuyang Wu, Stefano Falletta, Delia McGrath, and Sherry Yang. Crystalreasoner: Reasoning and rl for property-conditioned crystal structure generation, 2026. URL https://arxiv.org/abs/2605.14344.

Tian Xie, Xiang Fu, Octavian-Eugen Ganea, Regina Barzilay, and Tommi Jaakkola. Crystal diffusion variational autoencoder for periodic material generation. In *International Conference on Learning Representations (ICLR)*, 2022. URL https://openreview.net/forum?id=03RLpj-tc_. arXiv:2110.06197.

Andy Xu, Rohan Desai, Larry Wang, Ethan Ritz, and Gabriel Hope. Plaid++: A preference aligned language model for targeted inorganic materials design, 2026. URL https://arxiv.org/abs/2509.07150.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Sherry Yang, Simon Batzner, Ruiqi Gao, Muratahan Aykol, Alexander L. Gaunt, Brendan McMorrow, Danilo J. Rezende, Dale Schuurmans, Igor Mordatch, and Ekin D. Cubuk. Generative hierarchical materials search. In *Advances in Neural Information Processing Systems 37 (NeurIPS 2024)*, 2024. arXiv:2409.06762.

Hu Ye, Jun Zhang, Sibo Liu, Xiao Han, and Wei Yang. Ip-adapter: Text compatible image prompt adapter for text-to-image diffusion models, 2023. URL https://arxiv.org/abs/2308.06721.

Claudio Zeni, Robert Pinsler, Daniel Zügner, Andrew Fowler, Matthew Horton, Xiang Fu, Zilong Wang, Aliaksandra Shysheya, Jonathan Crabbé, Shoko Ueda, Roberto Sordillo, Lixin Sun, Jake Smith, Bichlien Nguyen, Hannes Schulz, Sarah Lewis, Chin-Wei Huang, Ziheng Lu, Yichi Zhou, Han Yang, Hongxia Hao, Jielan Li, Chunlei Yang, Wenjie Li, Ryota Tomioka, and Tian Xie. A generative model for inorganic materials design. *Nature*, 639(8055):624–632, March 2025. ISSN 1476-4687. doi: 10.1038/s41586-025-08628-5. URL https://www.nature.com/articles/s41586-025-08628-5.

Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 3813–3824, 2023. doi: 10.1109/ICCV51070.2023.00355. arXiv:2302.05543.

Zhengyang Zhou, Yunrui Li, Pengyu Hong, and Hao Xu. Multimodal fusion with relational learning for molecular property prediction. *Communications Chemistry*, 8(1):200, July 2025. ISSN 2399-3669. doi: 10.1038/s42004-025-01586-z. URL https://www.nature.com/articles/s42004-025-01586-z.

APPENDIX

A Related Work . . . 18

A.1 Molecules . . . 18

A.2 Single-modality materials models . . . 18

A.3 Multimodal materials via system-level composition . . . 18

A.4 Inference-time control of diffusion . . . 19

B Architecture design choices and ablations . . . 19

B.1 Teaching language models to natively understand materials through soft tokens . . . 19

B.1.1 Training objectives . . . 20

B.1.2 Atomistic encoder ablations . . . 21

B.1.3 Continuous atomistic-to-language adapter . . . 22

B.2 Guiding crystal denoising with language model embeddings . . . 23

B.2.1 Denoising diffusion model training and objectives . . . 23

B.2.2 Separating crystal structure prediction from de-novo generation . . . 25

B.2.3 The producer–consumer bridge . . . 28

B.2.4 Language model backbone finetuning for steering generation . . . 30

B.2.5 Architectural ablations over generative variants . . . 31

B.3 Text-to-Crystal Feynman-Kac algorithm details . . . 33

B.3.1 Bootstrap-SMC sampler . . . 34

B.3.2 Stoichiometry reward and Hungarian Z-override . . . 35

B.3.3 Additional hyperparameter sweeps and ablations . . . 36

B.3.4 Porting T2C-FK to enable ALM Edit . . . 36

B.4 Scaling laws . . . 36

C Training data . . . 36

C.1 Data used to teach Atomistic Language Models to understand atoms . . . 37

C.1.1 Five-bucket training mixture . . . 38

C.1.2 The URL leak failure mode and mitigations . . . 38

C.2 Data used to teach Atomistic Language Models to generate atoms . . . 39

C.2.1 Seven-bucket training mixture . . . 39

C.2.2 Bucket-mixture ablations . . . 40

D ALM Bench . . . 41

D.1 Task 1: directional editing (atom+text→atom) . . . 43

D.2 Task 2: application-consistency (text→atom) . . . 43

D.3 Task 3: polymorph (atom+text→atom) . . . 43

D.4 Task 4: doping substitution (atom+text→atom) . . . 44

D.5 Task 5: strain (atom+text→atom) . . . 44

D.6 Task 6: text→structure recovery (text→atom) 44
D.7 Evaluating newer frontier models and improving ALM performance 44

E Metrics 45

E.1 Property prediction metric definitions 45
E.2 Crystal structure prediction metric details 47
E.3 *De novo* generation metric details 50
E.4 Representational alignment metrics 50

A Related Work

Unimodal atomistic models, multimodal materials models built by system-level composition, and inference-time control of diffusion denoising serve as the foundation for Atomistic Language Modeling.

A.1 Molecules

Although ALMs are trained on inorganic crystals, multimodal models of molecules supply some working recipes, but lean on near-lossless string encodings (SMILES) that are faithful only to equilibrium conformations and for which crystalline materials have few analogs. SMILES (Weininger, 1988) and SELFIES (Krenn et al., 2020) encode molecular graphs as strings with essentially no information loss for an equilibrium geometry, making molecule-to-text conversion nearly trivial and yielding a long line of working systems. MoleculeSTM (Liu et al., 2023b) contrastively aligns descriptions and atomistic structures for retrieval; LLM-Fusion (Boyar et al., 2025) fuses SMILES, SELFIES, text, and learned embeddings for property prediction; 3D-MoLM (Li et al., 2024) fine-tunes a language model on 3D molecular encodings, showing that atomistic embeddings meaningfully enhance a language model, an approach similar to our atomistic encoder-soft token architecture. SMILES-based translators (Edwards et al., 2022; Kim et al., 2021; Qian et al., 2023) move between language and molecular structure, and MMFRL (Zhou et al., 2025) aligns SMILES with property data through spectra. Crystalline materials have no identical SMILES analog (e.g., CIF files preserve atomic coordinates but are not natural language).

A.2 Single-modality materials models

On the language side, LLM-Prop (Niyongabo Rubungo et al., 2025) predicts properties from Robocrystallographer text but discards numerical structural information; CrystalLLM (Antunes et al., 2024) trains autoregressively on CIF files for *de novo* generation; MatSciBERT (Gupta et al., 2022) captures domain language without structural input; and Crystal-Text-LLM (Gruver et al., 2024) fine-tunes a pretrained LM on CIF-style generation for crystal structure prediction, reportedly with greater diversity and stability than contemporary specialized atomistic models, evidence that language model priors carry useful inductive biases for crystal generation. On the structure side, atomistic foundation models (machine learning interatomic potentials) (Wood et al., 2026; Batatia et al., 2025) can be finetuned to predict properties from periodic graphs at state-of-the-art accuracy, and diffusion-based generators (Xie et al., 2022; Jiao et al., 2023; Zeni et al., 2025) sample structures from learned distributions over crystal lattices. Neither side alone couples text-conditioned structure generation with property reasoning; ALM combines them in one backbone, breaking the "GNN–LLM" accuracy wall on property prediction while generating crystal structures competitively with natural language steering.

A.3 Multimodal materials via system-level composition

Multimodal materials models to date compose pretrained components at the system level. MultiMat (Moro et al., 2025) aligns atomistic embeddings, numerical property data, and Robocrystallographer text with a CLIP-style contrastive loss, but supports only property prediction.

CLaSP (Suzuki et al., 2025) aligns language with atomistic structures from paper titles and abstracts and tests only retrieval. CLICS (Ozawa et al., 2024) contrastively learns over atomistic embeddings and Robocrystallographer text but cannot consume free-form language. MatterChat (Tang et al., 2026) and L^2M^3OF (Cui et al., 2025) consume free-form language and atomistic structure, but cannot generate atomistic structures. Against the failure modes of §1, none closes the full text-in, structure-out, structure-in, text-out loop; none places generation under cross-modal latent control to enable text-in, structure-in, structure-out generation; and their contrastive embedding spaces are aligned for retrieval, not steerable sampling. ALMs close that loop in a single model. In vision and language, analogous steps toward integrated multimodal understanding (LLaVA (Liu et al., 2023a)) and generation (Janus (Chen et al., 2025)) exist, yet do not transfer to atomistic understanding or scientific reasoning (Rubungo et al., 2025), nor to crystal generation. Their 2D image encoders carry no periodicity or atom-permutation symmetry, and the generator in Janus (Chen et al., 2025) discretizes outputs into a learned codebook that the continuous geometry and exact stoichiometry of crystals resist. LLaVA projects features for understanding only, while Janus generates within its own token space rather than bridging a language model to an external, physics-aware decoder, backpropagating the generative loss into the language model, or emitting a structured compositional target.

A.4 Inference-time control of diffusion

Conditional diffusion typically injects guidance during sampling. Classifier guidance (Dhariwal & Nichol, 2021) and classifier-free guidance (Ho & Salimans, 2022) steer the denoising trajectory toward conditioning information, but require differentiable score signals and do not support discrete compositional constraints. Particle-based Feynman–Kac methods (Singhal et al., 2025) reweight partial trajectories by a reward, softly enforcing arbitrary objectives without touching the diffusion model. None exploits a shared latent space between a language model and the diffusion decoder; T2C-FK does, scoring trajectories with a per-atom Hungarian assignment between the LM-conditioned predicted atomic distribution and the target stoichiometric multiset to enforce natural-language compositional targets at inference time.

B Architecture design choices and ablations

Atomistic Language Modeling (ALM) is designed to be the first paradigm that covers every direction of the materials structure–property–text map. Table 6 makes this concrete, contrasting ALM against the three model families that currently dominate materials science. Machine-learned interatomic potentials (MLIPs) regress properties from a fixed structure; atomistic generative models sample structures from a given composition, property, or an empty prompt; and large language models that serialize crystals as strings can read and write structures as text but carry no learned 3D geometric prior. Atomistic Language Modeling is the only paradigm that supports all seven.

Two conditional roles from one architecture. The full coverage in Table 6 is realized by two checkpoints that instantiate the same language model-to-diffusion bridge on different decoder backbones. **ALM Edit** is the model with the full language-to-atomistic bridge architecture and composition-observing diffusion model, accomplishing both $\mathbf{T} \to \mathbf{S}$ and $(\mathbf{S}, \mathbf{T}) \to \mathbf{S}$, while **ALM Gen** covers the quasi-unconditional, de-novo generation direction $(\varnothing, \mathbf{T} \to \mathbf{S})$, with a diffusion model that also denoises over element types. The structure-to-property and structure-to-text directions are served by the shared atomistic encoder, projector, and LLM forming **ALM Core**. The remainder of this section documents the architecture and the design choices that make this coverage possible: the atomistic encoder (§B.1.2), the input-side adapter (§B.1.3), the language model adaptation (§B.2.4), the generator and language-to-diffusion model bridge (§B.2), and the test-time T2C-FK steering method (§B.3).

B.1 Teaching language models to natively understand materials through soft tokens

ALM Core reads each material's 3D structure as continuous soft tokens (one per atom) and answers in natural language. It is trained with a causal language modeling loss over the assistant turn (below); the generation objectives for **ALM Edit** and **ALM Gen**, together with their auxiliary terms, are

Table 6: **Positioning.** Capability coverage of materials science model families. Columns are seven task directions over the structure, property, and natural-language modalities; a ✓ marks a direction each family supports natively with nontrivial performance.

Model family	Property → Structure	Text Desc. → Structure	Structure → Property	*De novo* (∅ → Structure)	Structure + Text → Structure	Structure → Text	Text → Text
MLIPs (Rhodes et al., 2025; Wood et al., 2026; Batatia et al., 2025)	×	×	✓	×	×	×	×
Atomistic generative models (Zeni et al., 2025; Jiao et al., 2023)	✓	×	×	✓	×	×	×
LLMs emitting string crystal reps. (Antunes et al., 2024; Gruver et al., 2024)	×	✓	✓	✓	×	✓	×
ALM (this work)	✓	✓	✓	✓	✓	✓	✓

Table 7: **Training hyperparameters for the three models.** Encoder $\mathcal{E}$ (OrbV3) is frozen throughout; ALM Edit/ALM Gen initialize the LM from the ALM Core understanding checkpoint. Data-mixture buckets and sampler are in Appendix C (ALM Core: Table 14; generation fine-tuning: Section C.2). "—" marks an inapplicable entry.

	ALM Core	**ALM Edit**	**ALM Gen**
Objective	$\mathcal{L}_{\text{LM}}$ (Eq. 1)	$\mathcal{L}_{\text{CSP}}$ (Eq. 7)	$\mathcal{L}_{\text{DNG}}$ (Eq. 8)
LM adaptation	LoRA r=128, α=256	full fine-tune	LoRA r=8, α=16
Diffusion backbone	—	CSP-mode MatterGen	DNG-mode MatterGen-Base
Bridge	P_{in} (2-layer MLP)	Q-Former (M=16) + IP-Adapter	per-token MLP + IP-Adapter
LM learning rate	2e−4 (LoRA)	5e−7	2e−4 (LoRA)
Projector/bridge lr	2e−5 (P_{in})	3e−4	3e−4
Optimizer	AdamW	AdamW	AdamW
Batch / GPU	4	2	4
Max tokens	2048	1536	1536
Optimizer steps	12,000	30,000	10,000
Grad clip	1.0	1.0	off
CFG dropout p_{drop}	—	0.2	0.2
CFG Guidance strength g	—	0.5	1.0
Diffusion steps T	—	100	1000
Guidance g (op. pt.)	—	0.5	0.5
Data mixture	5-bucket (Tab. 14)	7-bucket (Tab. 16)	7-bucket (Tab. 16)

stated with the generator formalism in Appendix B.2. Notation for the whole appendix can be found above, in Table 8.

B.1.1 Training objectives

ALM Core. Each training example is a ChatML-formatted token sequence $\mathbf{w} = (w_1, \ldots, w_L)$ with a supervised (assistant turn) index set $\mathcal{S} \subseteq \{1, \ldots, L\}$. System and user prompt tokens, as well as the input atomistic soft tokens, are label-masked out. Precisely, these soft tokens are node-wise embeddings of crystals outputted by the atomistic encoder, $\mathbf{H} \in \mathbb{R}^{N_p \times d_{\mathcal{E}}}$, projected into the LLM's input token space $P_{\text{in}}(\mathbf{H})$. The objective is the causal cross-entropy over the supervised positions,

$$\mathcal{L}_{\text{LM}} = -\mathbb{E}_{\mathbf{w}\sim\mathcal{D}_{\text{S}}}\left[\frac{1}{|\mathcal{S}|}\sum_{i\in\mathcal{S}}\log p_\phi\big(w_i \mid w_{<i}\big)\right], \tag{1}$$

where $p_\phi(\cdot \mid w_{<i})$ is the language model's (LM's) next-token distribution evaluated on the P_{in}-spliced input embeddings and $\mathcal{D}_{\text{S}}$ is the **ALM Core** training dataset (Appendix C.1). **Core**

Table 8: Symbol glossary for the generator formalism (Appendix B.2).

Symbol	Definition	Shape
$\mathbf{L}$,	Lattice vectors	$\mathbb{R}^{3\times3}$
$\mathbf{X}$	3D fractional coordinates	$[0,1)^{N_p\times3}$
$\mathbf{A}$	Atomic numbers	$\{1,...,100\}^{N_p}$
$\mathbf{u}_t = (\mathbf{L}_t, \mathbf{X}_t)$	Continuous diffusion state at time t	—
$\hat{x}_0$	Tweedie clean-structure estimate	—
$\sigma(t)$	SDE diffusion coefficient	scalar function of t
$s_\theta(\mathbf{u}_t, \mathbf{A}_t, t \mid \mathbf{C})$	Score network (GemNet-T)	—
N_p	Atoms per unit cell	integer
T	Diffusion / PC iterations	1000
$g \in \mathbb{R}_{\geq0}$	Classifier-free guidance scale	scalar
p_{drop}	CFG conditioning-dropout probability	0.2
K	Output-side atomistic tokens	K=8
N	Producer context window	128
d_{LM}	LM hidden dim (Qwen3-8B)	4096
M	Learnable producer queries	M=16
d_{cond}	MatterGen conditioning dim	512
d_h	Per-atom hidden dim inside GemNet-T	512
$\mathbf{Z}$	LM hidden states at the K atomistic-token positions	$K \times d_{\text{LM}}$
$\mathbf{S} = [\mathbf{Z}_{\text{ctx}}; \mathbf{Z}]$	Producer source (N context + K atomistic states)	$(N{+}K) \times d_{\text{LM}}$
$\mathbf{Q}_{\text{LQ}}$	Learnable producer queries	$M \times d_{\text{cond}}$
$\mathbf{C} = f_{\text{QF}}(\mathbf{Q}_{\text{LQ}}; \mathbf{S})$	Producer output (conditioning sequence)	$M \times d_{\text{cond}}$
$\widetilde{\mathbf{C}}(t)$	Timestep-fused conditioning	$M \times d_{\text{cond}}$
$\mathbf{h}_b$	Per-atom hidden state at GemNet block b	$N_p \times d_h$
$\gamma_b \in \mathbb{R}$	Learnable per-block bridge gate (init 1.0)	scalar

is trained with a warm start, in which the LLM is frozen, allowing only the projector P_{in} to train, for 5 epochs.

B.1.2 Atomistic encoder ablations

The choice of OrbV3 as the atomistic encoder $\mathcal{E}$ is justified here. Four architecturally distinct backbones are swept across: OrbV3 (Direct force prediction, 20 neighbor cutoff, trained on OMat, with 256-d per-atom features), UMA-S (Version 1.1 from `fairchem-core` 2.20.0 with `task=omat`, 128-d), PET-MAD XS (v1.5.0, 640-d), and PET-MAD S (v1.5.0, 1280-d). They were ablated over by holding the Qwen3-8B base, LoRA r=64 α=128, data mixture, and 4,608,000 total samples seen during training. Each arm had its own atomistic-to-language MLP projector (warm started with a similar number of steps) sized to the encoder feature dimension. Three benchmarks supply the downstream metrics: LLM4Mat-Bench MAD/MAE (adjusted to penalize for samples in which the LLM would not output a parseable property), in which a "good model" that is useful to scientists achieves at least a 5; test set Mat2Props raw MAE on narratives, i.e. property prediction given free-form text with structure, property, and application descriptions; and GNoME formation-energy MAE.

OrbV3 is the only encoder that clears the LLM4Mat-Bench "good model" threshold on a non-trivial number of configs ($4/9$ at leak-adjusted MAD/MAE ≥ 5; PET-S $2/9$, the others $0/9$), and the only arm whose checkpoint reaches the $\geq 98\%$ Mat2Props validity target at effective batch 256. UMA-S has the lowest raw GNoME formation-energy MAE (0.015 eV/atom on the surviving $\sim 47\%$), but its 52% URL-leak rate collapses the leak-adjusted comparison; PET-XS sits below both axes. PET-S is the only non-OrbV3 arm competitive on Mat2Props (winning formation energy and density at raw MAE), but below the validity $\geq 95\%$ floor. Richer feature dimensions ($d \geq 640$) do not translate into understanding alignment within the 12k-step budget, because the chat-template training has to relearn a projector geometry that OrbV3's 256-d features happen to provide nearly for free. The validity problem, in which Qwen3-8B, when finetuned, starts to regress to its pretraining priors, outputting IMGUR URLs or mangled JSON instead of the valid JSON they are prompted to write (discussed further in Appendix C.1.2).

Table 9: **Atomistic encoder ablation.** LLM4Mat-Bench MAD/MAE is leak-adjusted as RAW $\times$ (1 $-$ number of unparseable failures) (↑ better; ≥ 5 is the paper's "good model" threshold); the Mean is over the MP slice's four properties (formation energy, band gap, E_{hull}, density). Mat2Props MAEs are RAW on the MP held-out split (↓ better); Validity (Valid.) is the fraction of outputs that are JSON-parseable to a number. GNoME-FE columns are RAW MAE and the rate at which the LLM outputted unparseable values on the GNoME slice.

		LLM4Mat (↑)		Mat2Props MAE (↓)				GNoME-FE		Mat2Props
Encoder	dim	Mean	$\geq 5/9$	bg	E_f	E_h	ρ	MAE (↓)	Leak (↓)	Valid. (↑)
ORB	256	**6.42**	**4/9**	**0.244**	0.085	**0.070**	0.157	0.026	**2.6%**	**98.3%**
UMA-S	128	2.34	0/9	0.319	0.101	0.128	0.541	**0.015**	52%	47%
PET-XS	640	0.70	0/9	0.503	0.774	0.100	0.295	0.125	32%	20%
PET-S	1280	4.48	2/9	0.371	**0.083**	0.087	**0.160**	0.030	35%	71.5%

On the other hand, there is recent evidence that machine-learned interatomic potentials *converge in representation space* as they improve. Edamadaka et al. (2025) show across nearly sixty scientific foundation models that, on inputs similar to those seen during training, high-performing models align closely in latent space, with weaker models diverging into local sub-optima. Our four encoders are all competent MLIPs trained on overlapping bulk-crystal distributions, so on the in-distribution LLM4Mat-Bench / GPT-Narratives materials we evaluate, they expose nearly the same structural signal to the projector; the gaps in Table 9 are dominated not by representational content but by feature dimensionality (richer d slows projector alignment within a fixed step budget) and by each arm's downstream calibration (the URL-leak and validity floors). OrbV3 was chosen, then, doubly because of its inference speed. In practice, node-wise embeddings of all train, validation, and test set structures were cached and retrieved during each phase.

B.1.3 Continuous atomistic-to-language adapter

On the input side, the frozen OrbV3 encoder emits a variable-length per-atom representation (N tokens of 256 dimensions for an N-atom cell) that must be mapped into the language model's $d_{\text{LM}} = 4096$ embedding space before being prepended to the prompt tokens. ALMs use the simplest possible *adapter*: a two-layer GELU MLP, $\text{Linear}(256 \to 4096) \to \text{GELU} \to \text{Linear}(4096 \to 4096)$ ($\approx$21M parameters, with full parameter counts in Appendix B.2.4). This follows the LLaVA projector convention (Liu et al., 2023a; 2024), which uses the identical two-linear-with-GELU for vision language modeling. Crucially, the adapter is *token-preserving*: it emits one LM token per atom and leaves the encoder's variable-resolution output intact, so the LM's expanded sequence length scales *with* the structure rather than being compressed against a fixed budget.

Three families of cross-modal interface that have been successful in vision-language modeling each fail here because crystalline matter does not concentrate on a small set of recurring modes the way natural images do.

Learned-query bottlenecks (Q-Former, Perceiver IO). Q-Former-style adapters (Li et al., 2023b) and Perceiver IO (Jaegle et al., 2022) compress the foreign modality through a small fixed set of cross-attended learned queries (typically, 32 to 64). The construction assumes the encoder's output is well summarized by $L \ll N$ tokens. For a crystal of N atoms this assumption fails in both directions: when N is small ($N \leq L$) the bottleneck wastes capacity on null queries; when N is large it forces a fixed-budget compression on a representation whose information density grows with atom count. A 1000-atom defect supercell carries information that does not exist in any smaller subset of its atoms, and a single fixed L could struggle to represent both regimes at once. The two-layer MLP sidesteps the trade-off entirely by keeping the per-atom representation variable-length on the encoder side; a fixed-length learned-query producer is reserved for the **ALM Edit** and **Gen** *decoder* side, where the diffusion sampler accepts a fixed-size conditioning tensor by construction (detailed in Appendix B.2.3).

Codebook quantization (VQ-VAE, FSQ, JANUS). VQ-VAE (van den Oord et al., 2017), FSQ (Mentzer et al., 2024), and JANUS (Chen et al., 2025) round the latent representation to one of V discrete entries. The resolution requirement here is severe. The TiO_2 polymorph case is

illustrative, as mentioned in the main text. A codebook small enough to be trainable may struggle to resolve both regimes, and growing the codebook to that resolution simply converges to leaving the representation continuous in the limit. ALMs therefore keep the latent continuous and let the LM's own attention scores, rather than a discrete code, decide atomic detail.

Overall, a richer encoding would add more training burden and computational cost than benefit, as the real representational learning is done elsewhere: the frozen OrbV3 encoder already produces a physically grounded per-atom embedding (Appendix B.1.2), and the LM attends over those tokens. The adapter's only job is a per-atom linear lift into the LM embedding space, for which two layers with a GELU nonlinearity are sufficient capacity. Ablations that pooled all embeddings into a single token or 3 tokens unnecessarily gated the amount of information flowing to the LM, decreasing its property prediction performance. Empirically, warm start (projector-only training) alignment loss drops sharply within the first few hundred steps and plateaus near 0.10 (Appendix C.1), producing nearly perfect structural descriptions, confirming the dimensional lift is learned cleanly with this light, low-parameter map.

B.2 Guiding crystal denoising with language model embeddings

The ALM generator pairs Qwen3-8B with a pretrained crystal *diffusion decoder*, MatterGen, and steers that decoder through a learned conditioning channel via CFG. This section formalizes the bridge architecture proposed in the main text, as well as validates the choice of a diffusion decoder and of MatterGen specifically.

Choice of diffusion models as decoders. Conventional materials generation takes the form of *de novo* discovery (**ALM Gen**) and crystal structure prediction (**ALM Edit**). Score-based diffusion over the periodic-crystal manifold held the state of the art on both at the start of this work (Jiao et al., 2023; Zeni et al., 2025), jointly denoising lattice, fractional coordinates, and atomic numbers with periodic-translation and point-group symmetries often built in. Two properties make it the natural bridge target. First, the score network factors into an unconditional backbone and a conditioning branch, so an external producer attaches without retraining the backbone. Second, classifier-free guidance (CFG) exposes a single scalar g that interpolates from the unconditional backbone ($g{=}0$) to the fully conditioned model ($g{>}1$ extrapolates) — an inference-time dial on conditioning strength, decoupled from training, that underlies the stability tension analyzed below. Autoregressive string decoders (CrystaLLM (Antunes et al., 2024), Crystal-Text-LLM (Gruver et al., 2024)) expose no such dial, and VQ/codebook decoders quantize the latent we want kept continuous.

Choice of MatterGen as the decoder backbone. We built on MatterGen (Zeni et al., 2025) because of several reasons. Firstly, its GemNet-T score network accepts an external conditioning sequence through an adapter interface exposed in the released code, so a cross-attention bridge (Ye et al., 2023) needs only new key/value/gate parameters and no backbone surgery. Second, its stability-filtered `mattergen_base` checkpoint (Alex-MP-20) carries the metastability prior our training mixture lacks (quantified below), and serves as the **ALM Gen** backbone. Lastly, its from-scratch CSP-mode configuration, which observes atom types rather than denoising over them, was precisely what was retrained from scratch to serve as the base denoising model which **ALM Edit** used CFG to guide. Table 8 fixes the symbols used across all of Appendix B.2. A periodic crystal is factored into a continuous lattice $\mathbf{L}$, continuous fractional coordinates $\mathbf{X}$, and discrete atomic numbers $\mathbf{A}$; the continuous components are bundled into the diffusion state $\mathbf{u}_t = (\mathbf{L}_t, \mathbf{X}_t)$. We write the SDE diffusion coefficient as $\sigma(t)$ and the GemNet consumer block index is b.

B.2.1 Denoising diffusion model training and objectives

MatterGen factorizes a periodic crystal into a continuous lattice $\mathbf{L} \in \mathbb{R}^{3\times3}$, continuous fractional coordinates $\mathbf{X} \in [0,1)^{N_p\times3}$, and a discrete atomic-number assignment $\mathbf{A} \in \{1,\dots,100\}^{N_p}$. The continuous components $\mathbf{u} = (\mathbf{L}, \mathbf{X})$ evolve under a score-based forward SDE,

$$\mathrm{d}\mathbf{u}_t \;=\; \mathbf{f}(\mathbf{u}_t, t)\,\mathrm{d}t \;+\; \sigma(t)\,\mathrm{d}\mathbf{w}_t, \qquad \mathbf{u} = (\mathbf{L}, \mathbf{X}), \tag{2}$$

while the discrete atomic numbers diffuse under an absorbing-state D3PM (Austin et al., 2021) whose forward kernel mixes each clean type toward an absorbing MASK state with cumulative probability $\bar{\beta}_t$,

$$q(\mathbf{A}_t \mid \mathbf{A}_0) \;=\; \mathrm{Cat}\big(\mathbf{A}_t\,;\,(1-\bar{\beta}_t)\,\delta_{\mathbf{A}_0} \;+\; \bar{\beta}_t\,\delta_{\mathrm{MASK}}\big). \tag{3}$$

The score network $s_\theta(\mathbf{u}_t, \mathbf{A}_t, t \mid \mathbf{C})$ is parameterized by GemNet-T and depends on the conditioning sequence $\mathbf{C}$ exactly through the consumer cross-attention branch of Eq. (22). Reverse-time generation integrates

$$\mathrm{d}\mathbf{u}_t \;=\; \big[\mathbf{f}(\mathbf{u}_t, t) - \sigma(t)^2\, s_\theta(\mathbf{u}_t, \mathbf{A}_t, t \mid \mathbf{C})\big]\,\mathrm{d}t \;+\; \sigma(t)\,\mathrm{d}\bar{\mathbf{w}}_t, \tag{4}$$

paired with the D3PM reverse step for $\mathbf{A}_t$. The number of predictor-corrector iterations, or diffusion timesteps, is 1000 for all models; **ALM Edit** CSP match-rate on MP-20 is nearly flat in this count, with M@K (K=64) of 0.828 at 100 steps and 0.844 at 1000 steps.

The bridge is trained with classifier-free-guidance dropout: with probability p_{drop}=0.2 per step the `alm_embedding` conditioning is replaced by a learned zeros vector, so the network jointly learns the conditional and unconditional scores. At inference we apply the standard CFG extrapolation,

$$\widetilde{s}_\theta(\mathbf{u}_t, \mathbf{A}_t, t \mid \mathbf{C}, g) \;=\; s_\theta(\cdot \mid \varnothing) + g \cdot \big(s_\theta(\cdot \mid \mathbf{C}) - s_\theta(\cdot \mid \varnothing)\big), \tag{5}$$

where g=0 recovers the unconditional MatterGen distribution exactly.

Shared diffusion objective. Both generative models train the GemNet-T score network s_θ against the MatterGen denoising loss. For a conditioning signal C, with the corrupted state $(\mathbf{u}_t, \mathbf{A}_t)$ from the forward SDE (Eq. 2) and the absorbing-state D3PM (Eq. 3),

$$\mathcal{L}_{\mathrm{diff}}(C) = \mathbb{E}_{\substack{(\mathbf{u}_0,\mathbf{A}_0)\sim\mathcal{D}_3 \\ t\sim\mathcal{U}\{1,\dots,T\}}}\Bigg[\underbrace{\omega(t)\,\big\|\, s_\theta(\mathbf{u}_t, \mathbf{A}_t, t \mid C) - \nabla_{\mathbf{u}_t} \log p_t(\mathbf{u}_t \mid \mathbf{u}_0)\,\big\|^2}_{\text{lattice } \mathbf{L}+ \text{ coords } \mathbf{X} \text{ (score matching)}}$$
$$- \underbrace{\log p_\theta\big(\mathbf{A}_0 \mid \mathbf{A}_t, t, C\big)}_{\text{atom types } \mathbf{A} \text{ (absorbing D3PM)}}\Bigg], \tag{6}$$

with $p_t(\mathbf{u}_t \mid \mathbf{u}_0)$ the per-field forward kernel ($\mathbf{X}$ uses the periodic wrapped-normal kernel) and $\omega(t)$ the standard denoising weight (we reserve λ for the Feynman-Kac log-weight and g for the CFG scale). The conditioning C is produced by LLM encoder ϕ from the text prompt paired with $(\mathbf{u}_0, \mathbf{A}_0)$.

ALM Edit. **Edit** conditions on the Q-Former producer output $\mathbf{C} = f_{\mathrm{QF}}(\mathbf{Q}_{\mathrm{LQ}}; \mathbf{S})$, the M=16-token conditioning sequence (timestep-fused to $\widetilde{\mathbf{C}}(t)$ inside s_θ). It is trained with CFG conditioning dropout at rate p_{drop}:

$$\mathcal{L}_{\mathrm{CSP}} = \mathbb{E}_{\xi\sim\mathrm{Bern}(p_{\mathrm{drop}})}\Big[\,\mathcal{L}_{\mathrm{diff}}\big((1-\xi)\,\mathbf{C} + \xi\,\varnothing\big)\Big], \tag{7}$$

i.e. with probability p_{drop}=0.2, the producer sequence $\mathbf{C}$ is replaced by the learned null $\varnothing$, so s_θ learns the conditional and unconditional scores that the CFG mixing of Eq. 5 extrapolates at g=0.5. The score network is trained from scratch and LLM ϕ is fully fine-tuned.

ALM Gen. **Gen** replaces the Q-Former producer with a lightweight per-token projector P_{out}, which maps each of the K atomistic-token hidden states *independently* into the conditioning sequence $\mathbf{C} = P_{\mathrm{out}}(\mathbf{Z}) \in \mathbb{R}^{K\times d_{\mathrm{cond}}}$ with $[\,P_{\mathrm{out}}(\mathbf{Z})\,]_k = \mathrm{MLP}(\mathbf{Z}_k)$ — no learned queries, no context window, and no pooling — and feeds the resulting K-token sequence to the *same* IP-Adapter cross-attention consumer as **Edit**. Training uses an identical CFG-dropout objective:

$$\mathcal{L}_{\mathrm{DNG}} = \mathbb{E}_{\xi\sim\mathrm{Bern}(p_{\mathrm{drop}})}\Big[\,\mathcal{L}_{\mathrm{diff}}\big((1-\xi)\,\mathbf{C} + \xi\,\varnothing\big)\Big]. \tag{8}$$

The objective is identical to Eq. 7. The differences from **Edit** are the producer (a per-token MLP emitting the K-token sequence $\mathbf{C}$, versus the M=16-query Q-Former — both consumed by the *same* per-block IP-Adapter cross-attention), the LLM adaptation (LoRA r=8, versus full fine-tuning), and the backbone (MatterGen-Base in DNG mode, versus the from-scratch CSP-mode score network).

On top of the headline objectives (Eqs. 1–8), both **ALM Edit** and **Core** generation models add a per-element composition-count term (λ_{aux}=1.0) and a contrastive term (λ_{contr}=0.02); **ALM Edit**

adds a third, directional term (λ_{dir}=0.1, with λ_{dir}=0 for the de-novo **Core** model). The full generation loss is

$$\begin{aligned}\mathcal{L}_{\text{gen}} = \mathcal{L}_{\text{diff}}\big(s_\theta(\mathbf{u}_t, \mathbf{A}_t, t \mid \mathbf{C}),\, \mathbf{u}_0, \mathbf{A}_0\big) \\ + \lambda_{\text{aux}}\, \mathcal{L}_{\text{count}}\big(g_{\text{aux}}(\mathbf{C}),\, c(x)\big) + \lambda_{\text{contr}}\, \mathcal{L}_{\text{contr}} + \lambda_{\text{dir}}\, \mathcal{L}_{\text{dir}},\end{aligned} \tag{9}$$

with the per-element composition presence loss a class-balanced binary cross-entropy —

$$\mathcal{L}_{\text{count}} = -\frac{1}{N_Z}\sum_{z=1}^{N_Z}\Big[\, w_+\, c_z(x)\, \log\sigma(s_z) + \big(1 - c_z(x)\big)\log\big(1-\sigma(s_z)\big)\Big], \tag{10}$$

the contrastive (decorrelation) term

$$\mathcal{L}_{\text{contr}} = \frac{1}{B(B-1)}\sum_{i\neq j}\left(\frac{\bar{\mathbf{c}}_i^{\top}\bar{\mathbf{c}}_j}{\|\bar{\mathbf{c}}_i\|\,\|\bar{\mathbf{c}}_j\|}\right)^2, \tag{11}$$

and the directional term (**ALM Edit** only)

$$\mathcal{L}_{\text{dir}} = \frac{1}{|\mathcal{F}|}\sum_{i\in\mathcal{F}}\text{CE}\big(W_{\text{dir}}\,\bar{\mathbf{c}}_i,\; y_i\big), \qquad y_i = \mathbb{1}[\text{prompt } i \text{ raises the target property}]\,. \tag{12}$$

Here, $\mathbf{Z} \in \mathbb{R}^{K\times d_{\text{LM}}}$ are the atomistic-token hidden states extracted from the language model, and $\mathbf{C}$ is the producer output that conditions the score network s_θ (Section B.2.3). Auxiliary heads like g_{aux} (a per-element atom-count regression head trained with a BCE presence objective) operate on $\mathbf{C}$ directly, allowing gradients to reach the producer without passing through the score network. In $\mathcal{L}_{\text{count}}$, $s = g_{\text{aux}}(\mathbf{C}) \in \mathbb{R}^{N_Z}$ are per-element presence logits, $c(x) \in \{0,1\}^{N_Z}$ is the multi-hot composition of the target structure ($c_z(x)$=1 iff element z is present), N_Z=100 spans $Z \in \{1,\ldots,100\}$, and w_+=32 up-weights the rare present-element class. In $\mathcal{L}_{\text{contr}}$, $\bar{\mathbf{c}}_i$ is the producer output for prompt i averaged over its M conditioning tokens and B is the batch size, so $\mathcal{L}_{\text{contr}}$ is the mean squared off-diagonal cosine similarity across the batch. Intuitively, this is a decorrelation penalty pushing distinct prompts toward distinct conditioning vectors. In $\mathcal{L}_{\text{dir}}$, $W_{\text{dir}} \in \mathbb{R}^{2\times d_{\text{cond}}}$ is a learned linear head, $\bar{\mathbf{c}}_i$ is the same M-token-pooled producer output, and $\mathcal{F}$ is the subset of *directional* prompts, rows carrying an explicit raise/lower instruction (non-directional rows are masked out). The direction label is *not* a MatterGen cond_field: it never reaches the diffusion decoder and trains only the producer, forcing $\mathbf{C}$ to be linearly separable by direction, the targeted fix for the near-collinear "raise" and "lower" conditioning vectors of Appendix B.2.3. Finally, $\mathcal{L}_{\text{diff}}$ is the standard SDE/D3PM loss on the lattice, fractional coordinates, and atomic numbers $(\mathbf{L}, \mathbf{X}, \mathbf{A})$ (Section B.2.1). Equation (9) is the loss actually optimized during generation training, but only $\mathcal{L}_{\text{diff}}$ flows through the score network via CFG. The auxiliary losses are purely regularizers added to shape the latent steering vectors (producer output $\mathbf{C}$, and thus the atomistic-token states). *We present systematic ablations to validate the presence of each term below.*

Ablation results for auxiliary losses. To start, the composition auxiliary head is essential. Lowering the λ_{aux}=1.0 to 0.0 degrades de-novo metastable-SUN (MSUN) by 2%, but increasing it to 3.0 decreased MSUN by 49%. This effect is not only present in latent space, but also in observable statistics on generated samples. Figure 7 shows how turning the compositional auxiliary loss off loses prompt conditioning on difficult tasks, like asking **ALM Edit** to "generate a perovskite" and checking if any of $K = 20$ generated samples are perovskites (middle bar in each group). However, as **ALM Gen** is weakly conditioned on the atomistic tokens, the MSUN of its generated structures doesn't change drastically. With the atomistic token embeddings collapsing to a single, low-information vector, the model does not need to shift away from its language model priors, allowing it to stay performant on LLM judge evaluations of materials science knowledge.

The contrastive loss over atomistic token hidden states $\mathbf{Z}$ is also essential; without it, the $\mathbf{Z}$ collapses to a cosine distance of 0.12 between different prompts. With λ_{aux}=1, the average cosine distance between $\mathbf{Z}$ across all prompts in the evaluation dataset is 0.85.

B.2.2 Separating crystal structure prediction from de-novo generation

The architectural tension that leads to **ALM Edit** and **ALM Gen** being separate models is solely dependent on the denoising diffusion model $\mathcal{D}$. A fundamental limitation of MatterGen is that

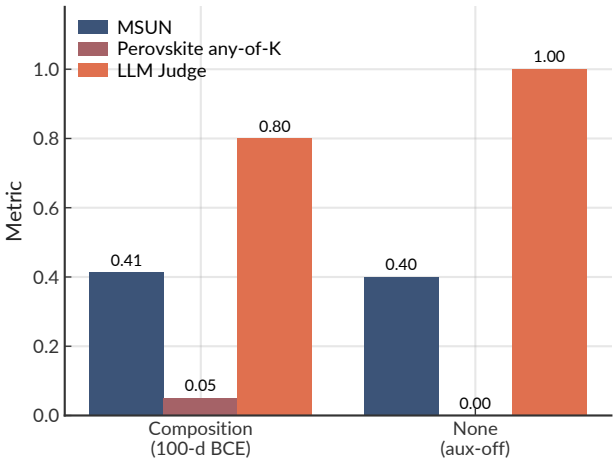

Figure 7: **Auxiliary supervision target comparison.** Composition BCE (the optimum) vs aux-off across MSUN, perovskite any-of-K, and the LM judge; aux-off keeps MSUN but loses prompt-following entirely.

Table 10: **MatterGen architectures trade off between *de novo* generation and crystal structure prediction performance.** Metastability of generated samples versus crystal structure prediction (CSP) performance. The two denoising diffusion models are architecturally identical at $g = 0$, but differ only in that MatterGen CSP does not denoise over element types, instead taking them in as input.

Backbone	Metastability ($E_{\text{hull}} \leq 0.1$)	CSP M@K=128 (MP-20)
ALM Gen @ $g = 0$ (MatterGen Base)	**0.750**	0.370
ALM Edit @ $g = 0$ (MatterGen CSP)	0.167	**0.777**

it trades composition and stoichiometry obeyance with stability of generated structures. When provided with the exact element count of a desired structure through CFG, Mattergen Base tends to generate crystals with similar compositions, but that are more stable (**T2C-FK** leverages the fact that denoised compositions don't stray too far to enforce composition and stoichiometry following). Mattergen CSP, which has the same architecture as Base but does not denoise over element types (instead taking them as input to directly initialize node embeddings), produces structures with far lower energy. There are two additional factors beyond architecture that help explain the disparity between each architecture in metastability and CSP: the underlying training data and guidance scale g, which also have strong interplay, as discussed below.

Underlying data carry stability and validity biases. The polymorphs in MP-20 don't always have the lowest energy out of other geometries and are not guaranteed to be metastable. MatterGen Base, which generates structures with stabilities at similar rates to its training data, would suffer from lower stability and SUN performance after training on MP-20 CSP. MatterGen CSP, which closely learns how to predict polymorphs from given compositions in MP-20, thus also learns to produce the labeled polymorphs, thus generating structures with lower stability than a *de novo* model that only trains on metastable structures. Another example of this is SMACT validity, a common metric for realistic structures including charge neutrality, electronegativity, and mixed-valence checks. **ALM Gen** and **ALM Edit** don't produce a high proportion of SMACT-valid structures. A large, contributing factor to this is our training data, which, as shown in Fig. 8 to the right, are only 39% SMACT-valid on average. In addition, many SMACT-valid structures are high energy (Fig. 8 left and middle), while many example materials with performant band gaps and properties of interest are SMACT-invalid (OQMD Fig. 8 right).

In addition, the aggregate distribution of materials that we post-train both models on has a large amount of volume away from the energetic hull (E_{h} or $E_{\text{above hull}} \leq 0$), as seen in Fig. 9.

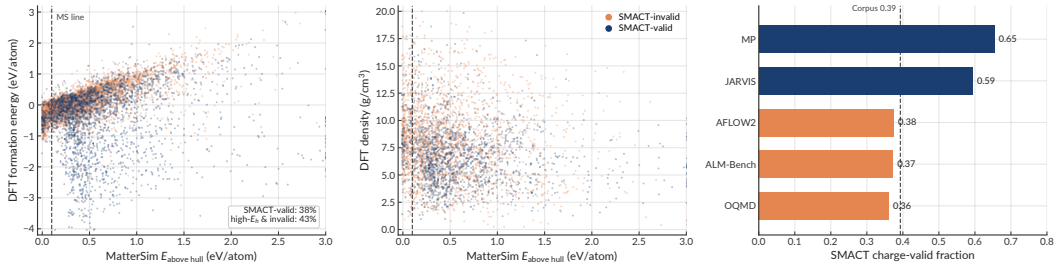

Figure 8: **SMACT charge-validity of the training compositions.** *Left, middle:* MatterSim E_h versus DFT formation energy and density, coloured by SMACT charge-validity. Most training compositions are charge-invalid and high-E_h (lower-right quadrant). *Right:* SMACT charge-valid fraction by training source.

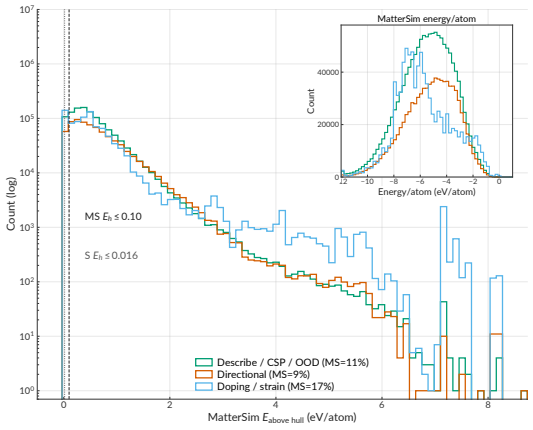

Figure 9: **Right-tailed energy distribution of the generation training corpus.** Main shows per-bucket MatterSim energy above hull (E_h, eV/atom) on a log-count axis over the full range. The mass of the distribution sits far above the hull with a tail extending to $\sim$8 eV/atom (the 92 percentile are OQMD/AFLOW structures). The metastable ($E_h \leq 0.10$, dashed) and stable ($E_h \leq 0.016$, dotted) thresholds are marked, and the inset shows MatterSim-evaluated energy-per-atom for the same buckets. A conditioned generator would reproduce this unstable distribution.

Further, when the MatterGen CSP mode model was pretrained on our dataset, its Match@$K = 20$ performance for MP-20 was 4% higher than the same architecture trained on MP-20-only, the same scale of difference as between **ALM Edit** and the second best model at CSP. Therefore, the ceiling for the stability is bounded by the pre-trained MatterGen diffusion model for *de novo* generation and the data that it was trained on.

The effect of guidance scale g on generation stability and crystal structure prediction. The guidance parameter g controls how much base denoising diffusion models obey the conditional priors instilled by the additional data they were finetuned on, by CFG's design. A model trained to reproduce an unstable distribution will, when conditioned, reproduce its instability. However, g offers the possibility of a tradeoff between instruction-following and generation quality (as measured by metrics like stability). Fig. 10 shows how g has different effects for a variant of the backbone used for **ALM Edit**. Raising g hurts CSP performance while helping editing tasks, as it controls the strength of the task- and input composition-encoding conditional signal from the language-to-atomistic bridge. This information is crucial for improving performance on **ALM Bench** tasks, but as **Edit** autoregressively generates the composition for the MatterGen CSP backbone when doing crystal structure prediction, any additional conditioning may pull the model away from its already strong performance.

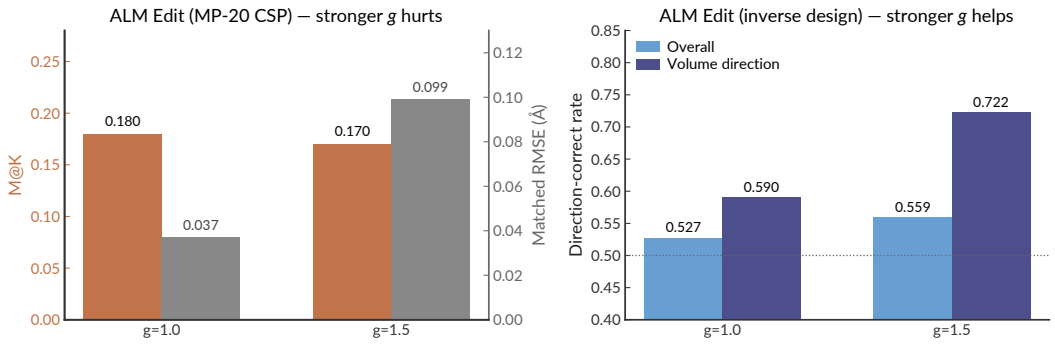

Figure 10: **ALM Edit CFG guidance scale tradeoff between CSP and inverse design.** Stronger g hurts crystal structure prediction (left, Match@K and RMSE for matches) but helps **ALM Bench** inverse design performance.

The guidance score produces very different behavior for **ALM Gen**. Here, g controls how much a global conditioning vector steers Mattergen Base away from its strongly performing frozen base. There is a positive operating range for g, leading to the choice of $g = 1.0$, as shown in Fig. 11.

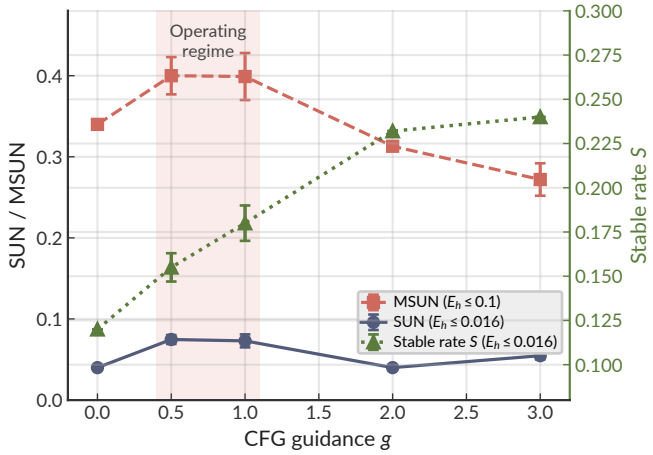

Figure 11: **CFG guidance scale sweep for ALM Gen.** SUN ($E_h \leq 0.016$, left axis, circles) and MSUN ($E_h \leq 0.1$, right axis, squares) depend heavily on g.

B.2.3 The producer–consumer bridge

Our language-to-atomistic bridge architecture enables the language model ϕ to guide the crystal diffusion decoder $\mathcal{D}$'s generation. Let $\mathbf{Z} \in \mathbb{R}^{K \times d_{\text{LM}}}$ be the final-layer hidden states of the K=8 atomistic-token positions emitted by ϕ (Table 8). These K tokens are randomly initialized and added to the model's vocabulary before training (we ablated over K values of 4, 8, and 16, and found that $K = 8$ balanced performance and computational cost). The producer reads these K states together with the window of N=128 language-model hidden states immediately preceding them, forming the source sequence $\mathbf{S} = [\mathbf{Z}_{\text{ctx}}; \mathbf{Z}] \in \mathbb{R}^{(N+K) \times d_{\text{LM}}}$ (length $N{+}K$=136), with a learned type embedding marking the N context states apart from the K atomistic states; the selected window width N=128 is ablated in Figure 15.

The *producer* is a shallow learnable-query transformer (Koh et al., 2023; Li et al., 2023b) that compresses $\mathbf{S}$ into a fixed-shape conditioning sequence

$$\mathbf{C} = f_{\text{P}}(\mathbf{Q}_{\text{LQ}}; \mathbf{S}) \in \mathbb{R}^{M \times d_{\text{cond}}}, \qquad M{=}16,\ d_{\text{cond}}{=}512, \tag{13}$$

where $\mathbf{Q}_{\mathrm{LQ}} \in \mathbb{R}^{M \times d_{\mathrm{cond}}}$ is a set of continuous, learnable queries that cross-attend into the LM-side source $\mathbf{S}$.

The *consumer* injects $\mathbf{C}$ into every block of $\mathcal{D}$'s GemNet-T score network as an IP-Adapter–style cross-attention head (Ye et al., 2023; Wang et al., 2025) — the only cross-attention in the network. MatterGen's native property conditioning does not enter through cross-attention: it is concatenated into a per-crystal latent broadcast into the atom embedding, and is therefore already carried by the frozen backbone block. Writing Ψ_b for that frozen block update and $\mathbf{h}_b$ for the per-atom hidden state at GemNet block b, the bridge adds a single gated read-out:

$$\mathbf{h}_b \leftarrow \underbrace{\Psi_b(\mathbf{h}_b)}_{\text{frozen backbone (native conditioning)}} + \gamma_b \cdot \underbrace{W_b^{\mathrm{mix}}\,\mathrm{Attn}\big(\Psi_b(\mathbf{h}_b)\,W_b^{Q},\ \widetilde{\mathbf{C}}(t)W_b^{K,\mathrm{alm}},\ \widetilde{\mathbf{C}}(t)W_b^{V,\mathrm{alm}}\big)}_{\text{new (trained): the network's only cross-attention}}. \tag{14}$$

The bridge contributes only $\{W_b^{Q}, W_b^{K,\mathrm{alm}}, W_b^{V,\mathrm{alm}}, W_b^{\mathrm{mix}}, \gamma_b\}$: the query reads the block's atom features and the keys and values are linear projections of the timestep-fused conditioning $\widetilde{\mathbf{C}}(t)$ (the producer output $\mathbf{C}$ fused with the noise level; Section B.2.3), with the mixin $W_b^{\mathrm{mix}} \in \mathbb{R}^{d_h \times d_h}$ projecting the read-out back into the GemNet stream. $\gamma_b \in \mathbb{R}$ is a learnable per-block scale (init 1.0); $g \in \mathbb{R}_{\geq 0}$ is the classifier-free guidance scale applied at sampler time. Two ControlNet-style zero-initializations (Zhang et al., 2023) make the bridge a no-op at training step zero: $W_b^{\mathrm{mix}} \equiv \mathbf{0}$ and the final layer of the timestep-fusion MLP (Section B.2.3) is zero-initialized in weight and bias.

Producer: cross-modal block stack. The producer maps the source sequence $\mathbf{S} \in \mathbb{R}^{(N+K)\times d_{\mathrm{LM}}}$ — the N=128 context states followed by the K=8 atomistic-token states (extracted as in Appendix B.2.5) — to the conditioning sequence $\mathbf{C} = f_{\mathrm{QF}}(\mathbf{Q}_{\mathrm{LQ}}; \mathbf{S}) \in \mathbb{R}^{M \times d_{\mathrm{cond}}}$ (M=16, d_{cond}=512), inspired by Q-Former (Li et al., 2023b). Expanded, f_{QF} is a stack of L_{QF}=2 transformer blocks (8 attention heads, M=16 learned queries), indexed by j (the consumer block index b is reserved for the GemNet stack below):

$$\mathbf{Q}^{(0)} = \mathbf{Q}_{\mathrm{LQ}}, \qquad \mathbf{S}^{\mathrm{cond}} = W^{\mathrm{down}}\mathbf{S} + \mathbf{E}_{\mathrm{type}} \quad (W^{\mathrm{down}} \in \mathbb{R}^{d_{\mathrm{cond}}\times d_{\mathrm{LM}}}), \tag{15}$$

$$\widetilde{\mathbf{Q}}^{(j)} = \mathbf{Q}^{(j)} + \mathrm{MHA}\big(\mathbf{Q}^{(j)}, \mathbf{Q}^{(j)}, \mathbf{Q}^{(j)}\big), \quad j = 0, \ldots, L_{\mathrm{QF}}{-}1, \tag{16}$$

$$\widehat{\mathbf{Q}}^{(j)} = \widetilde{\mathbf{Q}}^{(j)} + \mathrm{MHA}\big(\widetilde{\mathbf{Q}}^{(j)}, \mathbf{S}^{\mathrm{cond}}, \mathbf{S}^{\mathrm{cond}}\big), \tag{17}$$

$$\mathbf{Q}^{(j+1)} = \widehat{\mathbf{Q}}^{(j)} + \mathrm{FFN}\big(\widehat{\mathbf{Q}}^{(j)}\big), \tag{18}$$

$$\mathbf{C} = \mathrm{LayerNorm}\big(\mathbf{Q}^{(L_{\mathrm{QF}})}\big), \tag{19}$$

with $\mathrm{MHA}(\mathbf{Q},\mathbf{K},\mathbf{V}) = \mathrm{Softmax}(\mathbf{Q}W^{Q}(\mathbf{K}W^{K})^{\top}/\sqrt{d})\,\mathbf{V}W^{V}$. Equation (15) is the only place dimensionality changes from d_{LM}=4096 to d_{cond}=512; the rest of the Q-Former operates in the diffusion decoder's native conditioning space. The additive type embedding $\mathbf{E}_{\mathrm{type}}$ (one learned vector for the N context rows, another for the K atomistic rows), W^{down}, the per-block attention and FFN weights, and the learnable queries $\mathbf{Q}_{\mathrm{LQ}}$ are the only trainable parameters in the producer. This Q-Former-style encoder generalizes a Perceiver-style resampler (Jaegle et al., 2022): it compresses the variable-content $(N{+}K)$-token source into a fixed-length M-token conditioning set decoupled from the source length.

Timestep-aware conditioning fusion. The producer output $\mathbf{C}$ is timestep-independent, but the diffusion trajectory passes through wildly different noise regimes governed by $\sigma(t)$ (Appendix B.2.1). We fuse a noise-level encoding $\boldsymbol{\tau}(t) \in \mathbb{R}^{d_{\mathrm{cond}}}$ (the same NoiseLevelEncoding used by MatterGen) into $\mathbf{C}$ via a zero-initialized residual MLP:

$$\widetilde{\mathbf{C}}(t) \;=\; \mathrm{LayerNorm}\Big(\mathbf{C} \;+\; \mathrm{MLP}\big([\mathbf{C}\,;\,\boldsymbol{\tau}(t)]\big)\Big), \tag{20}$$

whose final linear is zero-initialized, so at training step zero $\widetilde{\mathbf{C}}(t) = \mathrm{LayerNorm}(\mathbf{C})$ and the MLP is the only path through which the bridge becomes noise-aware. $\widetilde{\mathbf{C}}(t)$ replaces $\mathbf{C}$ in Eq. (21).

Consumer: cross-attention injection. The bridge amplifies the LLM's conditioning signal with the Q-Former producer and injects it, at every denoising block, through a single decoupled cross-attention head. Let $\Psi_b(\mathbf{h}_b) \in \mathbb{R}^{N_p \times d_h}$ be the per-atom hidden state at the output of GemNet-T consumer block $b \in \{1, \ldots, L_{\mathrm{D}}\}$'s message passing ($N_p$ atoms per cell, d_h=512). Expanding Section 2.2's boxed Eq. (14), the injected read-out and the resulting block update are:

$$\Delta\mathbf{h}_b^{\mathrm{alm}} = W_b^{\mathrm{mix}} \cdot \mathrm{Softmax}\big(\Psi_b(\mathbf{h}_b)\, W_b^Q\, (\widetilde{\mathbf{C}}(t) W_b^{K,\mathrm{alm}})^\top / \sqrt{d_h}\big)\, \widetilde{\mathbf{C}}(t) W_b^{V,\mathrm{alm}}, \tag{21}$$

$$\mathbf{h}_b \leftarrow \Psi_b(\mathbf{h}_b) \;+\; \gamma_b \cdot \Delta\mathbf{h}_b^{\mathrm{alm}}, \tag{22}$$

where Ψ_b is the frozen GemNet block (message passing plus MatterGen's additive native-property conditioning) and $\widetilde{\mathbf{C}}(t)$ is the timestep-fused conditioning of Appendix B.2.3. The bridge introduces a single cross-attention head — the only cross-attention in the score network — whose query reads the block's neighborhood-aggregated atom features and whose keys and values are independent linear projections of the M=16 producer tokens $\widetilde{\mathbf{C}}(t)$; the mixin $W_b^{\mathrm{mix}} \in \mathbb{R}^{d_h \times d_h}$ projects the read-out back into the GemNet stream. Native property conditioning is *not* a parallel cross-attention — it is folded into Ψ_b via the concatenated per-crystal latent — so the score s_θ of Appendix B.2.1 depends on the producer output $\mathbf{C}$ *only* through this single gated branch. The per-block gate γ_b is learned; classifier-free guidance g is applied at sampler time through the CFG extrapolation of Eq. (5). The entire language-to-atomistic bridge has roughly 19M parameters.

Equation (22) shares MS-Diffusion's decoupled-K/V cross-attention topology (Wang et al., 2025) but differs in three deliberate ways — the source signal is LoRA-adapted Qwen3-8B hidden states (not CLIP image features), the queries $\mathbf{Q}_{\mathrm{LQ}}$ are purely learnable (not grounding-token-initialized), and the consumer keys/values derive only from $\widetilde{\mathbf{C}}(t)$ (not concatenated with the text stream). In addition, unlike canonical IP-Adapter (Ye et al., 2023), it keeps random K/V initialization, achieving zero contribution at the beginning of training from zero-initialized W_b^{mix} and timestep-fusion MLP.

B.2.4 Language model backbone finetuning for steering generation

LoRA and Full finetuning ablations. Full finetuning was necessary for the language model ϕ (Qwen3-8B) to produce rich enough atomistic token embeddings to steer generation in **ALM Edit**. LoRA, although at different ranks and α values, sufficed for **ALM Core** and **Gen**. We present systematic ablations to validate these choices, as well as to explain why only **Edit** performs well on the LLM-judged materials-knowledge retention task in **ALM Bench**.

We found that different LoRA parameters worked best for **ALM Core** versus **Gen**. **Core** uses rank r=128, α=256, dropout 0.05, at effective batch size 256, with a sweep shown in Table 11 over r.

Table 11: **LoRA rank ablation on the LLM4Mat-Bench MP slice.**

Config	LLM4Mat MP MAD/MAE	Leak Rate
r=8	2.5	$< 1\%$
r=32	4.0	$< 1\%$
r=64	5.23	$< 1\%$
r=128	**6.42**	$< 1\%$

To train **ALM Gen**, the 224 LoRA matrices from training **Core** were merged into the base weights at the start of finetuning, and a new LoRA was attached at r=8, α=16, and a learning rate of 1e-5, decoupling generation adaptation from **Core** materials understanding. Raising learning rate to 2e-4 collapses the cross-prompt hidden-state geometry (Fig. 12). Counter to the intuition that more adapter capacity helps, the final configuration outperforms an unmerged understanding LoRA at r=64, α=128, which regressed the mixed de-novo MSUN to 0.094. The inductive bias of the merged understanding LoRA already carries the LM-side adaptation, so the small fresh LoRA only has to translate the K=8 atomistic-token hidden states into useful conditioning.

An ablation over full- versus LoRA-finetuning **ALM Core** to develop **Gen** is shown in Table 12. Full finetuning achieves better metrics across all measured evaluations than LoRA $r = 8$. Crucially, it also prevents the degradation of instruction following observed with LoRA, in which the model starts to output IMGUR links or loop continuously, described further in Appendix C.1.2.

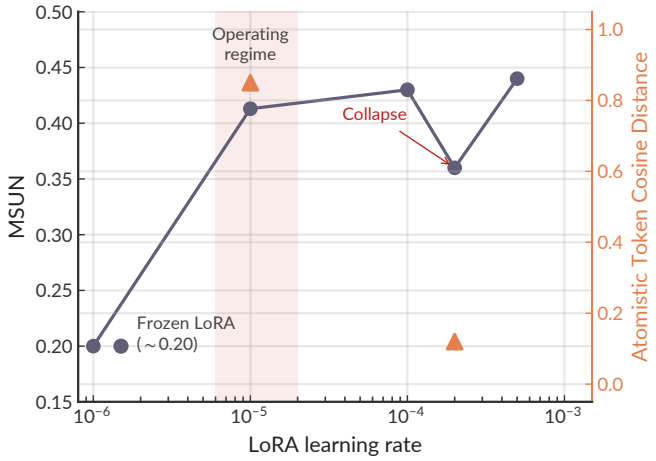

Figure 12: **generation LoRA learning-rate sweep.** MSUN as a function of fresh-rank-8 LoRA learning rate (log scale). lr=0 leaves the atomistic tokens out-of-vocabulary; lr=2e-4 collapses cross-prompt cosine distance to 0.12 (triangles on the right axis). lr=1e-5 is the selected setting.

Table 12: **Full finetuning versus LoRA on ALM Edit.** Full finetuning is done with PEFT. Retention judge is the LLM-judge score (0–2).

Metric	LoRA r=8	Full Finetuning
Retention judge (0–2)	1.053	**1.842**
loop-rate	0.316	**0.0**
keyword-pass	0.462	**0.923**
App-consistency	0.0275	**0.326**
Direction-correct	0.608	**0.621**
CSP Match@1	0.428	**0.472**
CSP Match@$K = 64$	0.881	**0.896**

As a result of the further training on instruction tuning, **ALM Edit** and **Gen** are better at predicting certain properties than **ALM Core**. The URL leak and loop rates are also near-zero for **ALM Gen**, as shown in Table 13.

Table 13: **ALM Variants evaluated on atomistic understanding and materials science knowledge tasks**.

	LLM4Mat MAD:MAE ↑				Mat2Props MAE ↓		Accuracy ↑			
Model	MP E_f	MP gap	MP ρ	OQMD E_f	E_f	gap	MaScQA	GPQA-chem	Mat2MCQ	GSM8K
ALM Core	14.12	3.88	10.10	13.23	0.087	0.275	0.643	0.247	0.417	0.775
ALM Gen	3.94	2.24	3.07	0.64	0.397	0.584	0.381	0.333	0.564	0.720
ALM Edit	15.79	4.37	12.33	15.65	0.074	0.244	1.000	0.228	0.508	0.790

B.2.5 Architectural ablations over generative variants

Language-to-atomistic bridge architecture ablations. Several ablations over simpler architectures of the language-to-atomistic bridge were conducted, validating the final design, as shown in Fig. 13. These architectures were a simple MLP across the $K = 8$ token embeddings concatenated together; IP-Adapter (Ye et al., 2023); a hand-constructed guidance vector consisting of a multi-hot vector to encode composition and additional bits to account for directional tasks in **ALM Bench** (e.g., 1 for raise, 0 for lower); FiLM (Perez et al., 2018); and lastly, one MLP per atomistic token, preserving the *Seq*uence information without flattening all of the embeddings upon input.

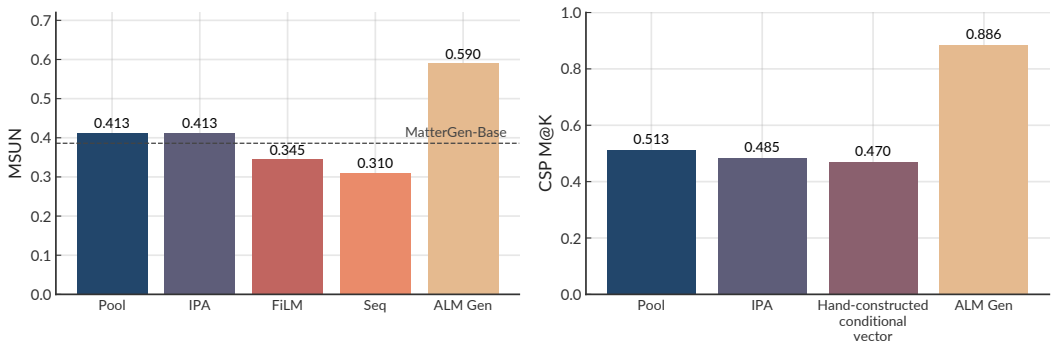

Figure 13: **Alternate bridge architectures on DNG SUN and CSP M@K.** Left: MSUN ($E_{\text{hull}} < 0.1$) at each bridge's optimal g. Right: CSP M@K at $K = 64$ on the CSP-mode backbone. $N = 500$ rows were drawn for each task from the test set, explaining the higher-than-reported CSP and MSUN than ALMs achieve in the main text.

Several of the architectures did not produce any valid structures for certain tasks, outputting generations that exploded upon relaxation or degenerate crystals with a single atom. However, no architecture performed as well as the consumer–producer bridge formulation used to build **ALM Edit** and **Gen**. One reason for this is that several of these architectures don't support growing the bridge contribution from a literal zero at step 0 (the ControlNet-style zero-init of Appendix B.2.3, (Zhang et al., 2023)). In practice, this stabilizes training across a wide range of CFG guidance scale g values. FiLM, Seq, and the hand-constructed conditional vector all did not support this training dynamic; as an example, FiLM-style feature-wise linear modulation (Perez et al., 2018) is multiplicative ($\gamma(\mathbf{C}) \odot \mathbf{h}_b + \beta(\mathbf{C})$) and initializes $\gamma \approx 1$, so it already perturbs every block at step 0, never establishes a conditioning gradient, and every learning-rate, warmup, and data-mix intervention drives it further toward a no-op. Seq also crashes under strong conditioning, e.g. producing degenerate cells on $20/20$ when prompted for perovskites.

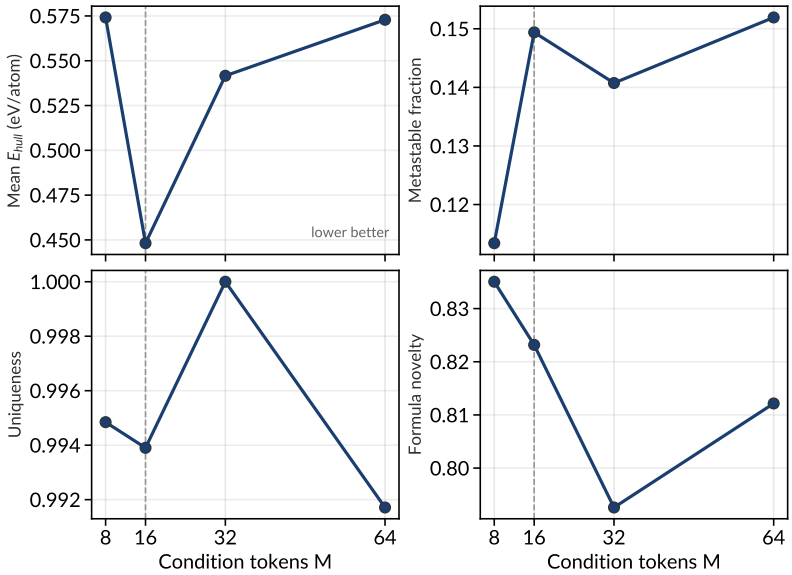

Figure 14: **De-novo generative metrics across condition-token count $M \in \{8, 16, 32, 64\}$.**

Fixed-length conditioning output from the producer. The producer emits a length-M condition sequence $\mathbf{C} \in \mathbb{R}^{M \times d_{\text{cond}}}$. M therefore sets the bandwidth of the signal the consumer can cross-attend to at each denoising step. The effect of sweeping over values of M on *de novo*

generation metrics is shown in Fig. 14. $M = 16$ was chosen as the operating point due to strong performance across each metric.

Cross-attention context window for the producer. The second producer-bandwidth hyperparameter is the length S, the total number of token embeddings cross-attended to by the M=16 queries. Here, $S = N + K$ is chosen such that N=128 recent LLM context states and K=8 atomistic token states. Widening N=128 to 512 does not help directional editing, although it marginally improves the rate at which **ALM Edit** generates structures that obey the prompted application area, as shown in Figure 15. We observe the collapse of direction following when the $E_f \uparrow$ rate decreases, as the model is trained on several tasks to output polymorphs with E_h lower than the inputted material, and thus is regressing to the distribution of structures marginalized over stability. This collapse is also observed when removing the composition auxiliary head, as well as swapping the order of the atomistic token teacher forcing to before the composition JSON is outputted (Fig. 16).

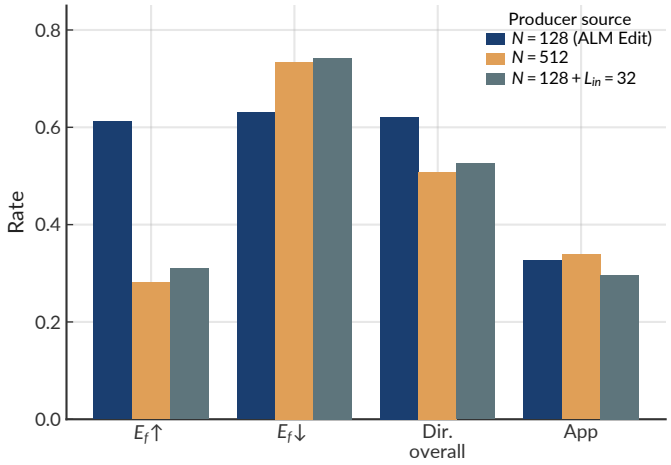

Figure 15: **Q-Former producer source-length (context-window) ablation** (g=0.5, honest denominator over all 768 atomtxt attempts; rest of the final recipe held fixed). *Both* ways of feeding the producer more source — widening the window (N=512) or prepending an explicit L_in=32 input-<atoms> segment — collapse raise-E_f direction-correctness below chance (the dashed line at 0.5) and lift lower-E_f, landing at overall direction-correct $\sim$ 0.51–0.53 versus the final N=128 recipe's symmetric 0.62. App-consistency is roughly unchanged across all three. The small directional residual the consumer reads is diluted by any extra source content, but coarse text→structure conditioning is not.

We also find that the conditional signal in **ALM Edit** is small (Fig. 17), learning a conditional head through CFG that is very close in cosine distance to the unconditional head output. However, the signal does produce slight g-dependence in the overall performance on directional tasks in **ALM Bench**.

B.3 Text-to-Crystal Feynman-Kac algorithm details

Text-to-Crystal Feynman-Kac steering (T2C-FK; Section 2.4) is an inference-time mechanism that makes **ALM Gen** generate crystals with a requested element set and stoichiometry. **ALM Gen** is a strong but deliberately weakly-conditioned de-novo sampler: its backbone produces stable cells, but the language prompt only biases the composition towards CFG-provided element counts. **T2C-FK** closes that gap by wrapping the unconditional reverse diffusion in an N-particle sequential Monte Carlo (SMC) sampler that reweights toward a stoichiometry reward, with no retraining or change to the score network. The same machinery ports to **ALM Edit** for composition-exact CSP decoding and for directional editing; those uses are collected as a side application in Appendix B.3.4. All diffusion notation follows the glossary of Appendix B.2.

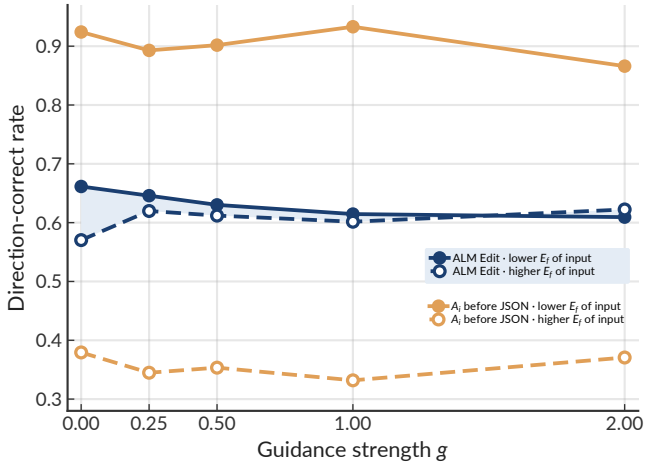

Figure 16: **Output-token ordering determines whether ALM Edit follows directional instructions.** Direction-correct rate (fraction of generations that moved formation energy E_f the requested way relative to the input) versus classifier-free-guidance strength g, for two output-token orderings: the **ALM Edit** ordering (composition JSON before the A_i atom tokens, blue, highlighted) and an ablation that teacher-forces the A_i tokens before the composition (A_i before JSON, orange).

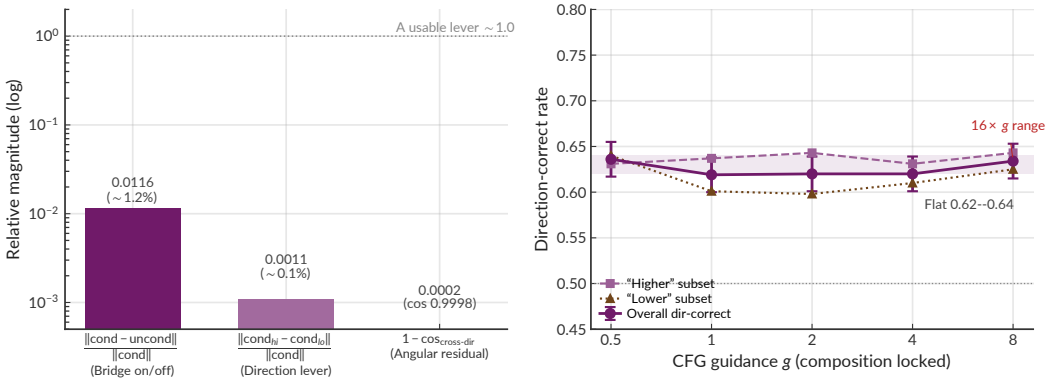

Figure 17: **Unconditional and conditional outputs are similar for ALM Edit.** *Left:* relative magnitudes (log scale) of setting g to be 0.5 (conditional) or 0 (unconditional). *Right:* direction-correctness is flat at 0.62–0.64 over a 16× range of g, indicating that the ceiling is a representation-quality limit, not a magnitude limit.

B.3.1 BOOTSTRAP-SMC SAMPLER

T2C-FK replaces MatterGen's single denoising trajectory with an N-particle bootstrap SMC sampler over the reverse diffusion (Wu et al., 2023; Singhal et al., 2025). All N particles are propagated in lockstep by the unconditional predictor-corrector step (shared score network s_θ, condition $\mathbf{C}$); every S steps the population is reweighted by the reward on the Tweedie clean estimate $\hat{x}_0$ and resampled when its effective sample size falls below ρN. The per-particle log-weight updates as

$$\ell_t^{(i)} \leftarrow \mathrm{clip}\big(\ell_t^{(i)} + \lambda\, r(\hat{x}_0(x_t^{(i)}, t)),\ \pm L_{\mathrm{clip}}\big), \tag{23}$$

where $\hat{x}_0(x_t^{(i)}, t)$ is the Tweedie estimator and λ the steering scale. Because FK reweights and resamples whole particles outside the score evaluation, it composes additively with the language-to-atomistic bridge architecture through CFG. In addition, the absorbing-state D3PM over $\mathbf{A}_t$ is MASK-dominated for $t > T/2$, where the reward on $\hat{x}_0$ is uninformative, so scoring begins

only once $t < T(1 - \tau_{\text{start}})$; with τ_{start}=0.5 this halves the scoring compute at no measurable cost in match rate. Algorithm 1 gives the full procedure.

Algorithm 1 T2C-FK: Feynman-Kac steered structure generation.

Require: text prompt u, target multiset $\mathcal{T}$, particles N, steps T, scoring period S, deferred start τ_{start}, scale λ, ESS threshold ρ, clip L_{clip}.

$\mathbf{S} \leftarrow$ context + [atoms_$*$] hidden states from $\phi(\text{tokenize}(u))$; $\mathbf{C} \leftarrow f_{\text{QF}}(\mathbf{Q}_{\text{LQ}}; \mathbf{S})$.
$\{x_T^{(i)}\}_{i=1}^N \sim p_{\text{prior}}$; $\ell^{(i)} \leftarrow 0$.
for $t = T-1, \ldots, 0$ **do**
  $\{x_t^{(i)}\} \leftarrow \text{PCStep}(\{x_{t+1}^{(i)}\}, s_\theta, \mathbf{C})$.
  if $t < T(1 - \tau_{\text{start}})$ **then**
    $\hat{x}_0^{(i)} \leftarrow \text{Tweedie}(x_t^{(i)}, t)$; $\ell^{(i)} \leftarrow \text{clip}(\ell^{(i)} + \lambda r(\hat{x}_0^{(i)}), \pm L_{\text{clip}})$.
    if $t \bmod S = 0$ **and** $\text{ESS}(\ell) < \rho N$ **then**
      $\{x_t^{(i)}\} \leftarrow \text{Multinomial}(\{x_t^{(i)}\}, \text{softmax}(\ell), N)$; $\ell^{(i)} \leftarrow 0$.
    end if
  end if
end for
Apply Hungarian Z-override on $\{x_0^{(i)}\}_{i=1}^N$.
return $\{x_0^{(i)}\}_{i=1}^N$.

Posterior-correction guarantee. **T2C-FK** accumulates the Feynman-Kac weight with the `sum` rule, $G_t^{(i)} = \exp\big(\lambda\, r(\hat{x}_0(x_t^{(i)}, t))\big)$ (alternate accumulation rules are ablated in Appendix B.3.3). Let $p^*(x_0) \propto p(x_0)\exp(r(x_0)/\tau)$ be the target posterior at t=0, with $p(x_0)$ the unconditional MatterGen marginal and τ an effective temperature set by λ. Bootstrap-SMC with proposal p and potential G_t recovers p^* as $N \to \infty$ under bounded reward and standard SMC regularity (Wu et al., 2023; Singhal et al., 2025); the three conditions hold here: the reward is bounded above (after clipping), the proposal is the unconditional MatterGen sampler at every step (FK never alters s_θ), and multinomial resampling on `softmax`-normalized log-weights is a valid SMC move. Clipping at $\pm L_{\text{clip}}$ introduces a controlled bias ($<5\%$ clip-saturation at ε=10^{-4}).

B.3.2 Stoichiometry reward and Hungarian Z-override

The reward is needed because providing element counts through CFG effectively enforces the element set, but MatterGen's score network loosens the stoichiometric ratios during denoising. With the auxiliary composition loss, decoding the atomistic-token hidden states $\mathbf{Z}$ through the generation auxiliary head recovers the target composition at $>95\%$ top-1, yet the unsteered diffusion trajectory drifts off the exact elemental ratio. The per-particle reward scores that ratio as a sum of three components,

$$r(\hat{x}_0) = r_{\text{stoich}}(\hat{x}_0) + r_{\text{count_L1}}(\hat{x}_0) + r_{\text{ratio_JS}}(\hat{x}_0). \tag{24}$$

For predicted per-atom element distributions $\{\mathbf{p}_a\}_{a=1}^{N_p}$ (N_p atoms in the cell) and target multiset $\mathcal{T}$:

$$r_{\text{stoich}} = -\frac{1}{N_p}\sum_a -\log\big(\mathbf{p}_a[z_{\pi^*(a)}] + \varepsilon\big), \tag{25}$$

$$r_{\text{count_L1}} = -\frac{1}{|\mathcal{S}|}\sum_{e\in\mathcal{S}} \big|\, n_e^{\text{argmax}} - n_e^{\text{target}} \,\big|, \tag{26}$$

$$r_{\text{ratio_JS}} = -\text{JS}\big(\hat{\mathbf{q}}^{\text{argmax}} \,\big\|\, \mathbf{q}^{\text{target}}\big)\,, \tag{27}$$

where π^* is the Hungarian (linear-sum) assignment of atoms to target elements, $\mathcal{S}$ the set of distinct target elements, n_e^{argmax} the count of element e under per-atom argmax, and JS the symmetric, $\log 2$-normalized Jensen-Shannon divergence. r_{stoich} is the negated mean Hungarian-assigned per-atom NLL (cost $C_{aj} = -\log(\mathbf{p}_a[z_j] + \varepsilon)$, solved at $\sim 10\,\mu$s per particle, capped at $\log(1/\varepsilon) \approx 9.2$); $r_{\text{count_L1}}$ and $r_{\text{ratio_JS}}$ are each bounded in $[-1, 0]$. The two hard-argmax components are necessary: a uniform soft distribution that averages to the target stoichiometry earns a perfect soft score even though no individual particle is valid, so $r_{\text{count_L1}}$ and $r_{\text{ratio_JS}}$ commit each atom to a single element *before* scoring, breaking it.

Hungarian Z-override. After the final denoising step, the linear-assignment problem is solved on the terminal atomic-number probabilities and each atom is set to its assigned element $z_{\pi^*(a)}$, leaving the lattice $\mathbf{L}$ and fractional coordinates $\mathbf{X}$ untouched. SMC has already concentrated mass on target-consistent particles, so this almost always agrees with the argmax it replaces; its role is to *guarantee* the composition match on the residual.

The per-atom reward is informative only when N_p is a multiple of the number of atoms in the requested material: otherwise, every Hungarian assignment is forced to place wrong-element atoms, the soft NLL of Eq. 25 is fixed at its $\log(1/\varepsilon)$ cap, and the cumulative log-weight saturates L_{clip} on every particle, causing SMC to degenerate to uniform resampling. **T2C-FK** therefore makes the number of particles N_p a multiple of the total number of atoms. This fixed aspect of particles is what makes the `sum` potential well-behaved: with the right atom count, each per-atom NLL stays bounded away from its cap, so every step contributes a finite, discriminative increment.

B.3.3 Additional hyperparameter sweeps and ablations

ALM Gen's T2C-FK configuration is N=8, T=1000, S=10, τ_{start}=0.5, λ=0.5, ρ=0.5, L_{clip}=50, ε=10^{-4}, the `sum` potential, and the equal-weight three-component reward over the last half of a T=1000 trajectory. The DNG-FK row of Table 5 uses these settings. A value of λ=0.5 is selected for **ALM Gen** stoichiometry and λ=3 for directional editing (Appendix B.3.4), where the stronger signal is needed to overcome MatterSim's relaxation bias toward lower energy. **T2C-FK** with 8 particles costs $\sim 8.1\times$ unsteered MatterGen sampling. As for reward components: r_{stoich} gives a calibrated continuous signal early in the second half of denoising while the element distributions are still soft; $r_{\text{count_L1}}$ supplies discrete-count alignment as the absorbing-state D3PM commits to specific elements; $r_{\text{ratio_JS}}$ is the bounded shape regularizer that keeps the SMC from collapsing onto a single dominant particle.

Tolerance ε was swept over. Values of ε=10^{-8} (per-atom NLL cap $\sim$ 18.4) saturated L_{clip}=50 on every wrong-element atom and degenerated the SMC into uniform resampling because every particle's cumulative log-weight hit the clip at once. The final version for **T2C-FK** uses ε=10^{-4} (cap $\sim$9.2), the smallest value keeping clip-saturation below 5% in our diagnostic logs. Lastly, the `sum` rule (Appendix B.3.1) accumulates $\lambda\, r(\hat{x}_0)$ each step. Other formulations of the rule, including a difference rule $G_t^{(i)} = \lambda\big(r(\hat{x}_0(x_t^{(i)}, t)) - r(\hat{x}_0(x_{t+1}^{(i)}, t{+}1))\big)$ do not reach the same performance as the sum formulation.

B.3.4 Porting T2C-FK to enable ALM Edit

The same T2C-FK serves **ALM Edit** by allowing for rewards prescribed by **ALM Bench** directional editing tasks to steer structure generation during denoising. Specifically **ALM Bench** task direction is parsed from the prompt into the reward, a MatterSim energy reward (one MatterSim forward pass yielding potential energy E/atom). On **ALM Edit** at g=0.5, this formulation of FK pushed the **ALM Bench** direction-correctness rise to 0.719. Inference-time FK reward-steering recovers both directions whenever the requested direction can be parsed from the prompt into an explicit reward.

B.4 Scaling laws

Across several regression metrics, strong scaling laws emerge for property prediction tasks, as shown in Fig. 18; as the underlying language model, Qwen3, grows in parameter count, property prediction performance improves. The cleanest series is JARVIS-QETB energy, where both the MAD/MAE skill ratio and the raw MAE follow a near-linear trend in log–log space. The trend is not universal: indirect gap and MatText perovskites improve with scale but retain task-specific curvature and noise. Cantor-HEA shows emergent scaling, where additional size does not help until a sudden jump from 4B to 8B parameters. We hypothesize that further scaling of the ALM could produce similar emergent effects.

C Training data

This appendix documents the data ALM is trained on, in the same order the model acquires its capabilities: first *understanding* (structure → text and structure+instruction → text/value, Section C

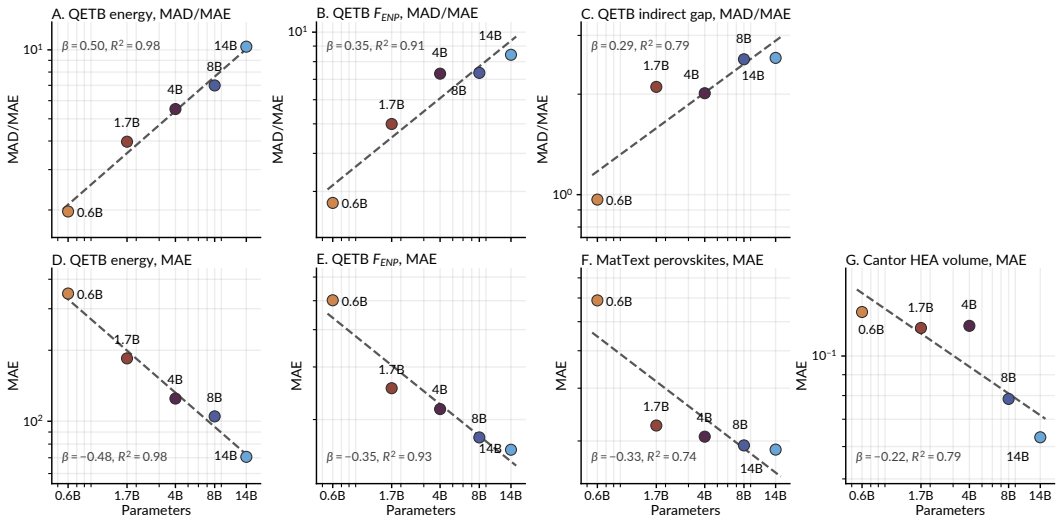

Figure 18: **Model-size scaling across seven property-prediction metrics.** Each panel uses the available Qwen3 **ALM Core** evaluation suite for 0.6B, 1.7B, 4B, 8B, and 14B-sized LMs. The top row reports MAD/MAE skill ratios on LLM4Mat-Bench (Rubungo et al., 2025) JARVIS-QETB properties (higher is better). The bottom reports raw MAE on JARVIS-QETB, MatText, and Cantor-HEA properties (lower is better).

below), then *generation and editing* (Section C.2). For each phase we give the exact bucket mixture, the per-bucket source datasets and sample counts, the pairing procedure, and the balance ablations that fix the mixture weights. Generation/editing buckets are deferred to Appendix C.2; the precise definitions of the evaluation metrics quoted here live in Appendix E.1.

C.1 Data used to teach Atomistic Language Models to understand atoms

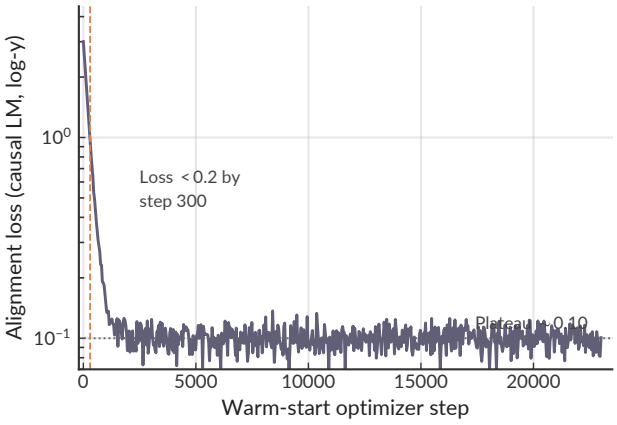

Figure 19: **Warm-start alignment loss trajectory** (K=8)**.** Causal LM loss on $\sim$ 1.35M structure-description pairs drops below 0.2 in $\sim$300 optimizer steps and plateaus near 0.10.

ALM is taught to understand atoms in two phases. *Alignment* fits the input-side projector P_{in} so the frozen LM can attend to atomistic features at all; *instruction tuning* then leverages this warm start to attach a LoRA adapter and finetune the language model on the full distribution of structure-conditioned tasks it must answer. Both phases keep the OrbV3 encoder frozen.

C.1.1 FIVE-BUCKET TRAINING MIXTURE

Stage 1: Alignment data and optimizer. Only P_{in} is trained, under the standard causal language modeling loss on nearly 1.35M structure–description pairs drawn from LLM4Mat-Bench (Rubungo et al., 2025) and the four GPT-Narratives parquets (`dft_3d`, `mp_3d_2020`, `aflow2`, `oqmd`). AdamW (Loshchilov & Hutter, 2019) is used at learning rate $1e{-3}$, as well as weight decay 0, betas $(0.9, 0.999)$, a 3% cosine warmup capped at 2000 steps then cosine decay, gradient clip 1.0, and an effective batch of 32. The alignment loss falls below 0.2 within $\sim$ 300 steps and plateaus near 0.10 across the full 23k-step run (Figure 19). The two-linear-with-GELU projector was chosen over a single-linear map (insufficient capacity) and a three-linear map (marginal gain at extra parameters); LLaVA-1.5 (Liu et al., 2024) uses the same two-linear projector at smaller scale. The encoder-adapter design space is analyzed in Appendix B.1.3.

Stage 2: Instruction-tuning mixture. With P_{in} aligned, a LoRA adapter is attached to the LM, then instruction-tune on a fixed budget of 12k optimizer steps (following the LLaVA-1.5 convention (Liu et al., 2024) of total optimizer steps, *not* epochs over any single bucket). Training draws from a **five-bucket** mixture (Table 14) spanning two structure-conditioned tasks and three text-only tasks; the per-step bucket selection is a categorical draw with probability vector π. The two structural buckets (`describe`, consisting of tasks prompting the model to describe the structure of a material or to generate a free-form narrative about its device applications, `property_apps`, consisting of property prediction tasks which request a JSON with only the given property as output) share the $\sim$ 0.80 structural budget at a $0.40/0.40$ split, while the `arxiv` (given a title and keywords of materials science preprints, generate a realistic abstract) supplies scientific fluency, and `camel`/`mascqa` (both scientific QA datasets) text buckets supply scientific fluency and Q&A instruction following to prevent catastrophic forgetting. Over the full run, this mixes a total of 2.82M training samples. Figures 21 and 22 visualize representative rows of the dataset and how **ALM Core** responds compared to the ground truth labels.

Table 14: **Understanding-phase five-bucket mixture.** π is the categorical probability used by the per-rank bucket sampler (Appendix C.1.2)

Bucket	Task type	Source	Rows	π
`describe`	captioning (structure $\rightarrow$ prose)	LLM4Mat `description` + GPT-Narratives	$\sim$700k	0.40
`property_apps`	VQA (structure + instruction $\rightarrow$ value/text)	LLM4Mat property + GPT-Narratives EXPLAIN	1.45M	0.40
`arxiv`	ChatML IT (title + categories $\rightarrow$ abstract)	JARVIS arXiv abstracts	375,571	0.14
`camel`	text Q&A	CAMEL-AI chemistry + physics role-plays	37k	0.04
`mascqa`	MCQ benchmark	MaScQA (80% train split)	519	0.02

Figure 20 visualizes the distributions of properties for inputted materials across the data buckets.

During this stage of training, AdamW is used to optimize two groups of parameters with different learning rates: LoRA at learning rate $2e{-4}$, projector at $2e{-5}$, and both with weight decay 0.01, betas $(0.9, 0.95)$, and a cosine schedule with warmup $\min(2000, 0.03\,T_{\max})$. The effective batch size is 256. Validation buckets are partitioned with the same seed via `split_seed=42`: arXiv and CAMEL hold out 500 rows each, MaScQA holds out 20% stratified by topic ($131/650$).

C.1.2 THE URL LEAK FAILURE MODE AND MITIGATIONS

The effect of batch size on URL leak rate. The understanding phase (Stage 2 of training) requires an effective batch size of at least 64. Below this threshold (Table 15) the LoRA receives too few gradient updates to suppress Qwen3-8B's base-language model web prior at the given LoRA learning rate, which re-surfaces on uncertain tasks: at an effective batch size of 16, 98.3% of Mat2Props outputs return URL placeholders (`i.imgur.com` / `materialsproject.org` links inside a markdown image embed) rather than structured numeric answers in JSON. Learning rate is not the cause (matching the same learning rate at low effective batch reproduces the failure), token-wise suppression at inference time did not help, and a smaller Qwen3-0.6B for an effective batch size of 16 fails identically. At an effective batch size of 256, the URL leak rate falls below 1% and Mat2Props validity rises to 98.3%.

The leak is a base model prior, not data contamination. Auditing the supervised text confirms it is essentially URL-free: the `describe`, `property_apps`, `arxiv`, and `mascqa` buckets contain

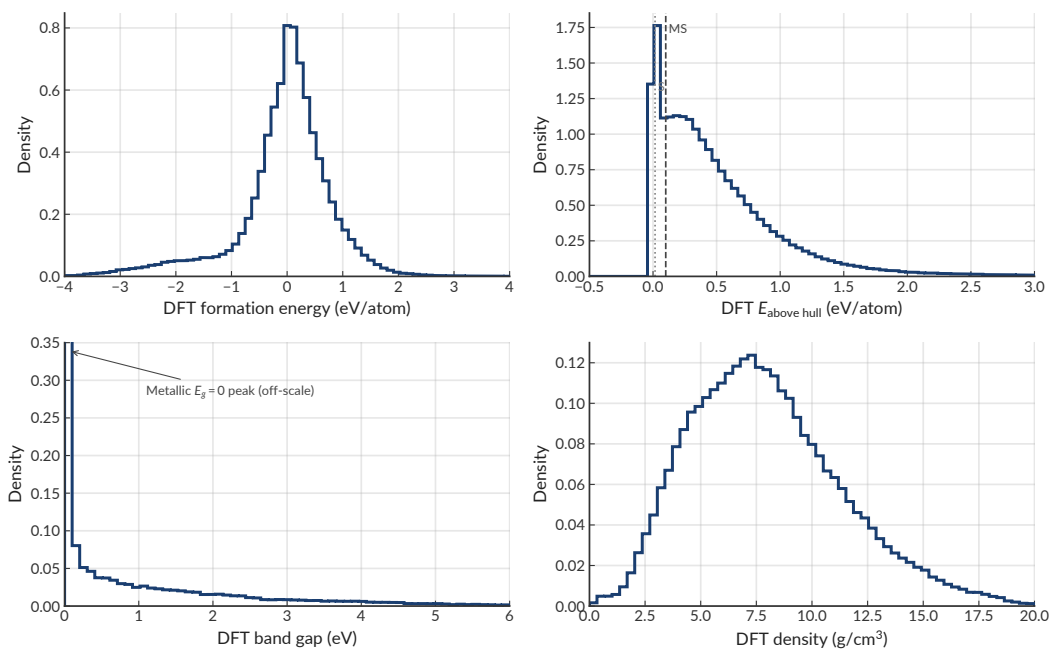

Figure 20: **Target-property distributions** over the training data (formation energy, energy above hull, band gap, density). These set the support of the values ALM is asked to read off and, in the editing phase, to move.

Table 15: **Effective batch size ablation.** Mat2Props validity and URL-leak rate (fraction of outputs that are URL placeholders or contain a leaked URL prior to property recovery). Leak rate is defined in Appendix E.1.

Config	Mat2Props validity	URL leak
ALM Core (B_{eff}=256, r=128)	**98.3%**	$< 1\%$
B_{eff}=16, lr=1e-4/1e-5	4–7%	95%
B_{eff}=16, lr=2e-4/2e-5	4%	96%
Qwen3-0.6B at B_{eff}=16	3.4%	97%

zero `imgur` occurrences and `camel` contains only two, across millions of rows. The leak is therefore a Qwen3-base pretraining prior that surfaces under fine-tuning-induced uncertainty, not a property of the training. Filtering 28k LLM4Mat predictions per checkpoint for `i.imgur.com` confirms this (Figure 23): the base model never leaks (0.0%), including the arXiv text bucket in training data but without ChatML tokens leaks 27.7%, and dropping the arXiv bucket entirely makes the leak strictly worse (73.2%). This is because removing the scientific-text exposure removes the text-only task contributions during training that keep the prior suppressed when learning on soft tokens. By including the dataset but formatting it with ChatML tokens, the leak rate issue is fixed.

C.2 Data used to teach Atomistic Language Models to generate atoms

The final training phase seeks to train the language-to-atomistic bridge to enable **ALM Edit** and **Gen** to steer the denoising of crystal structures.

C.2.1 Seven-bucket training mixture

The generation mixture is drawn from seven task buckets at the weights of Table 16: three unpaired text→structure buckets (`describe`, a long GPT-Narrative→structure caption; `csp`, a (formula, space-group) crystal-structure-prediction prompt; `ood`, an LLM-rephrased noisy-template

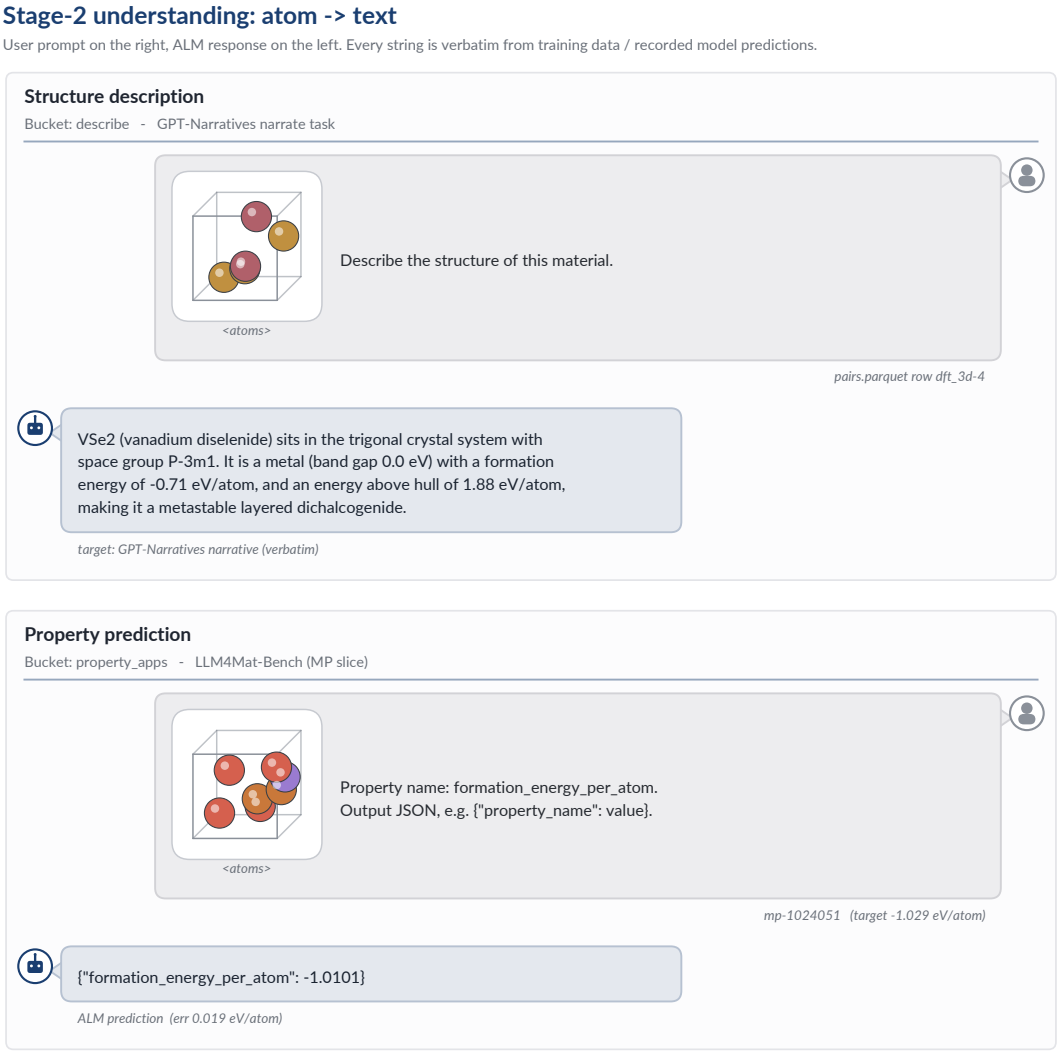

Figure 21: **Representative understanding-phase interactions**, one per bucket type (user prompt on the left, ALM response on the right): captioning (`describe`), property and applications VQA (`property_apps`), and the text-only science Q&A buckets (`arxiv`/`camel`/`mascqa`). The `<atoms>` placeholder is expanded inline to OrbV3 node features for the structure-conditioned turns.

paraphrase of the same structures) plus an application-class bucket (`app`), and three paired structure+text→structure editing buckets (`atomtxt`, directional property edits; `polymorph`, polymorph→lower-hull edits; `doping`, single-element substitution and strain). The `describe`, `csp`, and `ood` buckets share the same underlying GPT-Narrative (Park et al., 2024) structures ($\leq$ 20 atoms, K=8 tokens) and differ only in how the prompt is templated; the editing buckets are constructed as explicit input/output structure pairs. The `atomtxt` or "directional" bucket consists of **ALM Bench** tasks and was built as described in Appendix D.1.

C.2.2 Bucket-mixture ablations

Multi-bucket versus single-bucket. Spreading weight across the seven buckets does slightly reduce the performance of certain individual tasks. In particular, adding the directional **ALM Bench** editing buckets at the cost of `csp` weight drops CSP match-rate by 44%. We nonetheless use the multi-bucket mixture throughout, because it is the mixture that enables the directional-editing capability. On the other hand, increasing the weight on the **ALM Bench** (`atomtxt`) bucket from

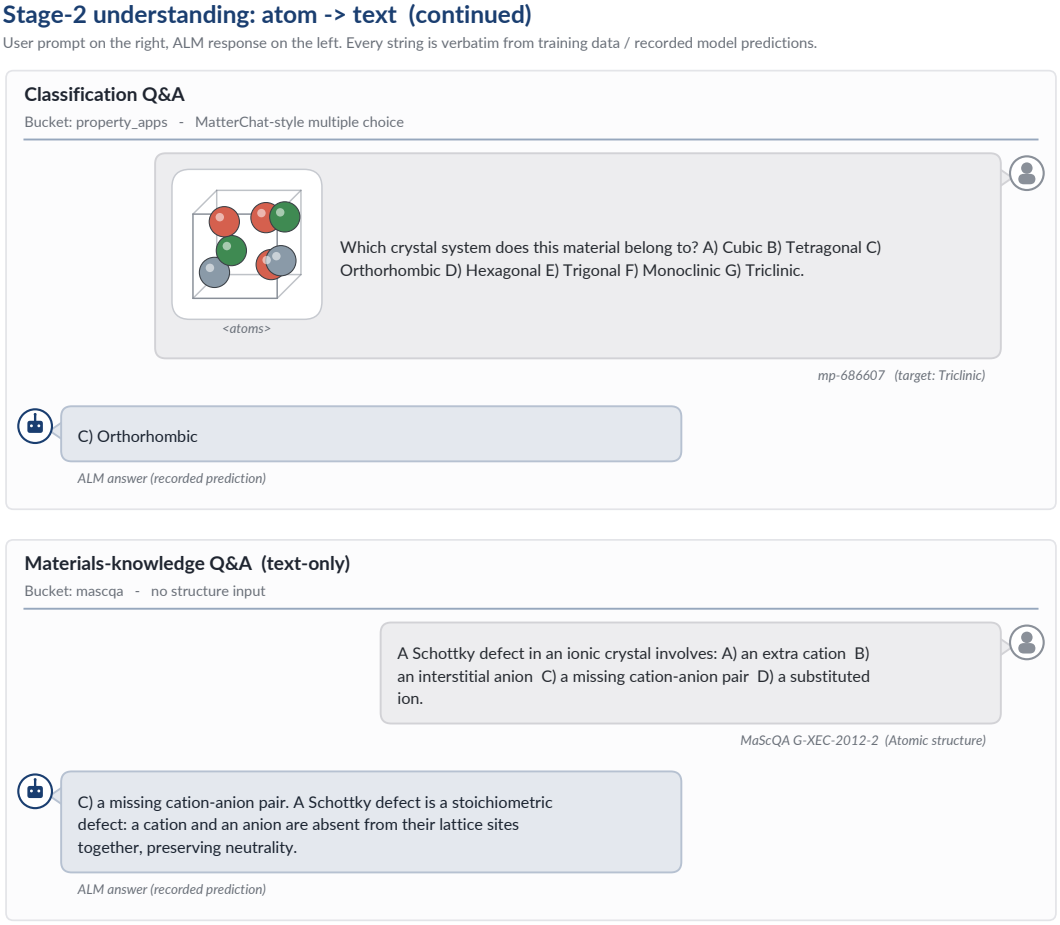

Figure 22: **Representative understanding-phase interactions (continued).**

Table 16: **Seven-bucket generation training mixture.** Different bucket weightings π were found to yield the best performance for **ALM Edit** and **Gen**. The weights indicate the likelihood for a row from each bucket to be drawn during training.

Bucket	Task	Rows	π (ALM Edit)	π (ALM Gen)
`describe`	long narrative $\rightarrow$ structure	1,352,176	0.08	0.10
`csp`	(formula, sg) $\rightarrow$ structure	1,352,176	0.15	0.20
`ood`	noisy template $\rightarrow$ structure	1,350,711	0.08	0.10
`app`	application class $\rightarrow$ structure	20,000	0.04	0.05
`atomtxt`	(struct., prop. dir.) $\rightarrow$ struct.	882,499	0.40	$0.05^{\ddagger}$
`polymorph`	(struct., polymorph) $\rightarrow$ struct.	545,145	0.15	0.25
`doping`	(struct., edit) $\rightarrow$ structure	1,000,000	0.10	0.25

ALM Gen's 5% to 40% lifts directional-editing performance significantly. The directional ceiling on the pooled bridge is therefore *structural* rather than data-scale-bound; a different bridge architecture and finetuning method would lead to higher performance.

D ALM BENCH

We introduce **ALM Bench**, a benchmark for *conditional* crystal generation in which the conditioning carries both a structure and a natural-language instruction. It is the first benchmark

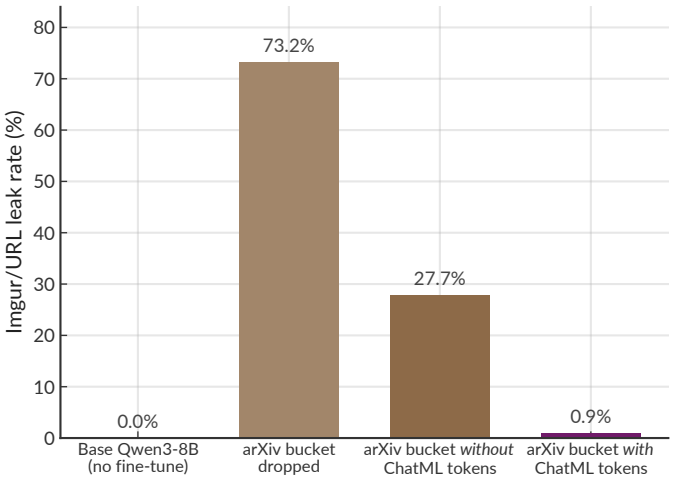

Figure 23: **URL-leak rate is set by how the arXiv bucket is formatted.**

to score the **atom+text→atom** and **atom+text→text** tasks. Here, **input(s)→output(s)** indicates a model capable of taking in a prompt of **input** modalities and generating responses in **output** modalities, a notation that will be used frequently in this section. In particular, **ALM Bench** asks a model to generate a new polymorph of the given crystal that *satisfies a described intent*, scoring the result against a physical predictor or an LLM judge instead of against a single reference. **ALM Edit** is evaluated on each task here. The benchmark comprises 11 scored tasks: six *directional editing* tasks (E_f, ρ, and V, each raised or lowered), *polymorph*, *doping/substitution*, *strain*, *application-consistency*, and *text→structure recovery* (the Describe and OOD prompts, similar to the *describe* bucket in **ALM Core**'s instruction tuning dataset) whose metric formulas are given in §E.1. Every scored rate is reported as the **mean ± 95% CI over** $5 \times N{=}200$ **held-out prompts**. Fig. 24 visualizes the distributions of structure sizes across the 7 buckets. Generated structures for all tasks (besides the structure recovery task) are briefly relaxed, as detailed in Appendix D.1.

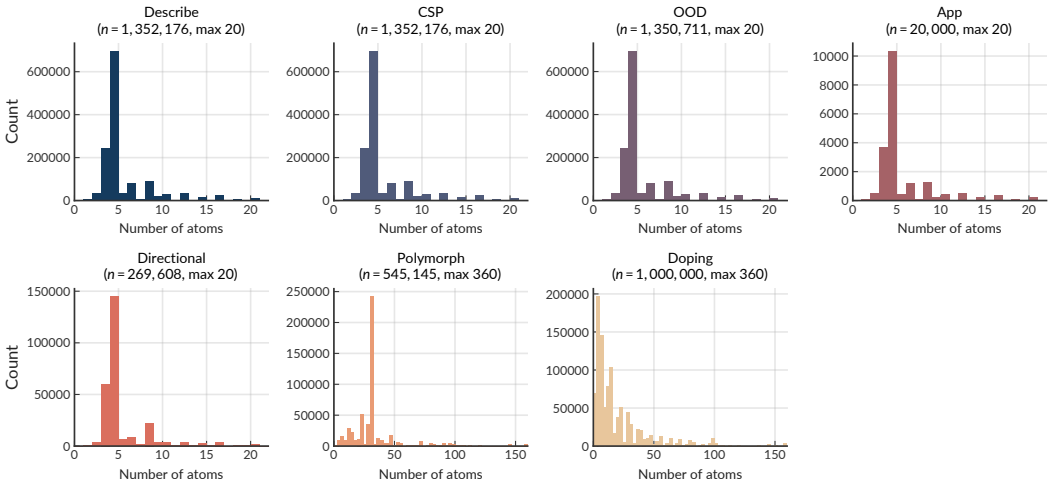

Figure 24: **Per-dataset atom-count distributions** for the structural training data. The structural buckets follow the LLM4Mat/GPT-Narratives distribution; the generation/editing pairs of Appendix C.2 are capped at ≤ 20 atoms per cell.

D.1 TASK 1: DIRECTIONAL EDITING (ATOM+TEXT→ATOM)

The directional editing task requires *paired* structures (A, B) that share a composition but differ along one property in a known direction. Our final pairing technique builds these pairs entirely *within a single parent dataset* so that DFT-calculated property labels have consistent levels of theory: it (i) clusters candidate structures by reduced formula, deduplicating near-identical polymorphs; (ii) attaches a MatterSim single-point energy-per-atom (and density / volume) label to every member of a cluster; and (iii) for each source structure emits *both* a higher-E and a lower-E target drawn from the same cluster. Emitting both directions per source auto-balances the dataset across `higher`/`lower` prompts. The text prompt ends with "... with a higher / lower ⟨property⟩," and the input and target materials are the paired structures, accordingly.

Scoring. We MatterSim-relax both the input and the generated structure (full-cell relaxation, $f_{\max}$=0.05, 500 steps, as it de-games a pure lattice rescale, which would otherwise relax straight back to the input volume), then score whether or not the MatterSim-calculated properties have the direction prompted for in language. For density and volume requests, a correct edit must *additionally* be valid, composition-preserving, and structurally distinct from the input under `StructureMatcher` (`ltol`=0.3, `stol`=0.5, `angle_tol`=10). The headline metric is `direction_correct_rate` = (#moved the requested way)/(all scored candidates), with degenerate / NaN-property gens kept in the denominator.

Sub-categories. There are three types of directional editing tasks (within which are "higher" and "lower" subtasks):

- **formation energy** E_f: MatterSim total energy per atom, valid at fixed composition;
- **density** ρ: relaxed mass / volume;
- **volume** V: relaxed cell volume per atom;

The E_f "higher" prompts, in particular, push *against* the energy-lowering relaxation prior of the diffusion decoder, whereas a "lower" request takes advantage of the lower-energy distribution of unconditional generations.

D.2 TASK 2: APPLICATION-CONSISTENCY (TEXT→ATOM)

This task evaluates whether models can take in a generic application prompt that contains *no* formula and *no* space group (e.g. "a porous metal hydride for lightweight hydrogen storage") and generate crystals matching the requested application class. This is a text-conditional crystal generation task.

Scoring. A GPT-4o-mini or GPT-4o judge (which were ablated across to reveal roughly similar scoring preferences) reads a (prompt, formula, space group, N_p, elements, density, volume per atom, formation energy) tuple of the MatterSim-relaxed generation and returns a verdict on the $\{0, 1, 2\}$ scale. An invalid raw structure is forced to score 0. The headline metric is the per-prompt mean consistency $\in [0, 2]$, reported alongside the fraction scoring 2 and the fraction judged inconsistent. The judge was independently calibrated against 150 ground-truth (positive, negative-control) pairs across 8 application classes, producing TP $90/90$ and TN $60/60$, confirming that a high inconsistent-verdict rate reflects the model, not a judge artifact. The dominant failure mode across application classes is that the model omits the required element entirely (a metal-hydride prompt returns a hydrogen-free structure, or a Li-ion-cathode prompt returns a lithium-free one).

D.3 TASK 3: POLYMORPH (ATOM+TEXT→ATOM)

This task consists purely of inputting crystals and instructing the model to "Generate a lower-energy polymorph." The generated crystal must have the *same composition* as the input but a different geometry and a *lower* energy. The `StructureMatcher` tolerances used to verify that the input and output structures are not identical are (`ltol`=0.3, `stol`=0.5, `angle_tol`=10).

Scoring. The primary metric is the fraction of generations whose MatterSim-relaxed total energy per atom is below the relaxed input's, *gated* on the generation being valid,

composition-preserving (reduced formula equal to the input), and structurally distinct from the input (`StructureMatcher.fit(input, gen)` is false). The pairs are MP-derived polymorph→lower-E_{hull} mappings (the `polymorph` training bucket of Table 16), held out for evaluation.

D.4 Task 4: doping substitution (atom+text→atom)

This task consists of "Substitute element X with element Y" prompts, wherein the model must perform a clean single-species substitution in a known input crystal.

Scoring. A generation counts as successful only when the dopant is present, the donor is removed, the per-element ratio matches, *the structure is valid*, and it is distinct from a naive relabel (the same positions up to numerical precision with element types swapped). Here, the per-element ratio of the target and generated structures can differ by up to 10%.

D.5 Task 5: strain (atom+text→atom)

Extending the section above (Appendix D.4), another metric to determine if a generated, doped crystal is realistic is calculating its strain compared to the original crystal. The doping edit alone (Task 4) ignores this lattice-deformation axis, so a dedicated strain task is added that shares the same parquet and prompt ("replace X with Y") but additionally scores whether the generated cell adjusts its volume by the correct amount.

Scoring. In addition to the validity and element ratio metrics mentioned in the section above, the strain on the generated doped crystal should be similar to that of the labeled doped crystal. After an initial, short relaxation, the equilibrium strain of the generated crystal is compared to that of the reference, achieving success if they differ by less than 5%.

D.6 Task 6: text→structure recovery (text→atom)

This task is to generate crystals from their structure descriptions, the inverse of the understanding *describe* (structure→text) task. We evaluate two prompt styles over the same held-out materials: **describe** (the verbose generation narrative from Rubungo et al. (2025) and Park et al. (2024), e.g. "... has a tetragonal crystal system with space group P4/nmm") and **OOD** (a terse, out-of-training-style spec, e.g. "Show me As, Cu, Si, Ti in 119.56 Å^3 and 8 [atoms]").

Scoring. Each generation is scored against the ground-truth structure on how well the elemental compositions match and an ordered `StructureMatcher` with `ltol`=0.3, `stol`=0.5, `angle_tol`=10 to match the generation to the GT. Intuitively, this is a crystal structure prediction task with a much richer and more free-flowing language prompt. No relaxation is applied.

Qualitative examples. Figure 25 shows verbatim **ALM Edit** transcripts (**ALM Bench** prompt on the left, model response on the right) across four representative ALM-Bench task types, including one deliberate failure case (a "higher formation energy" request, the arm that pushes against the relaxation prior) to make the scoring concrete.

D.7 Evaluating newer frontier models and improving ALM performance

Several of **ALM Edit**'s failures at Describe (Struct.) and OOD tasks come from the underlying **Core** language model failing to recognize the material or application area described in the prompt. To illustrate this, we provided the same prompt to Qwen3.5-9B, a more recent release than the Qwen3-8B used throughout this work. In addition, because it had not been finetuned without reasoning traces as **Core** had, CoT thinking could be enabled, unlike **ALMs**. We then sampled the suggested composition of the material and teacher forced the outputted JSON into **Edit**'s generated response. Using another LLM to recommend materials compositions for **Edit** to generate mitigates deficiencies in the base language model and allows **ALMs** to beat 5.6-Sol on four of the five generation tasks and remain competitive with even GPT-6 Astra (Table 17). Table 18 reports the editing tasks with the same additional baselines.

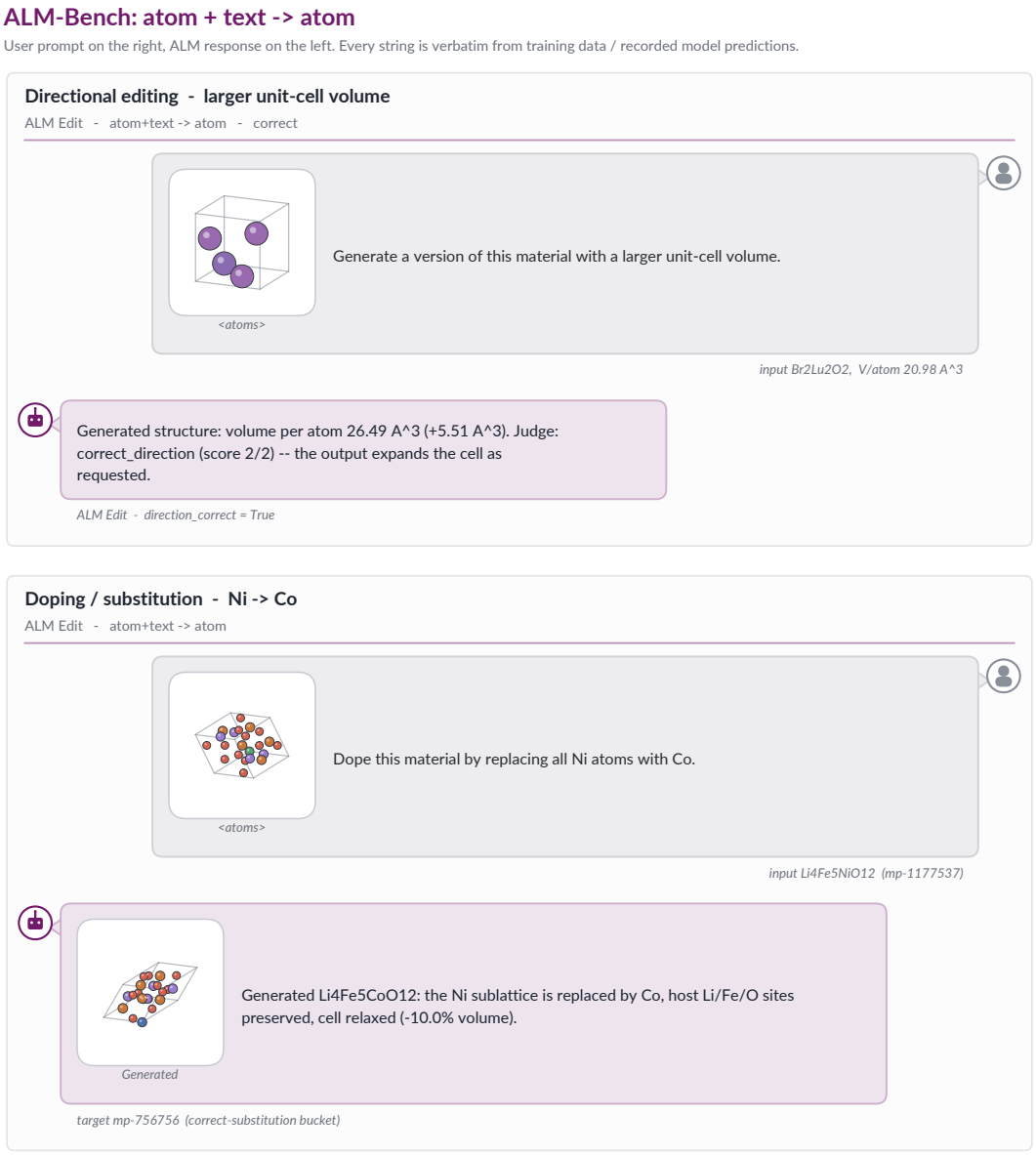

Figure 25: **ALM-Bench chat examples.**

E METRICS

We lay out exact descriptions of each metric used to quantify performance in our work and enable **ALM Core, Edit,** and **Gen** to serve as comparable baselines for future work.

E.1 PROPERTY PREDICTION METRIC DEFINITIONS

MAD / MAE. For a regression property we report the ratio of the test-set Mean Absolute Deviation to the model's Mean Absolute Error,

$$\mathrm{MAD/MAE} = \frac{\frac{1}{n}\sum_i |y_i - \bar{y}|}{\frac{1}{n}\sum_i |\hat{y}_i - y_i|}, \tag{28}$$

where $\bar{y}$ is the test-set mean. This is the scale-free skill score of LLM4Mat-Bench (Rubungo et al., 2025): *higher is better*, MAD/MAE=1 is no better than the mean predictor, and ≥ 5 is

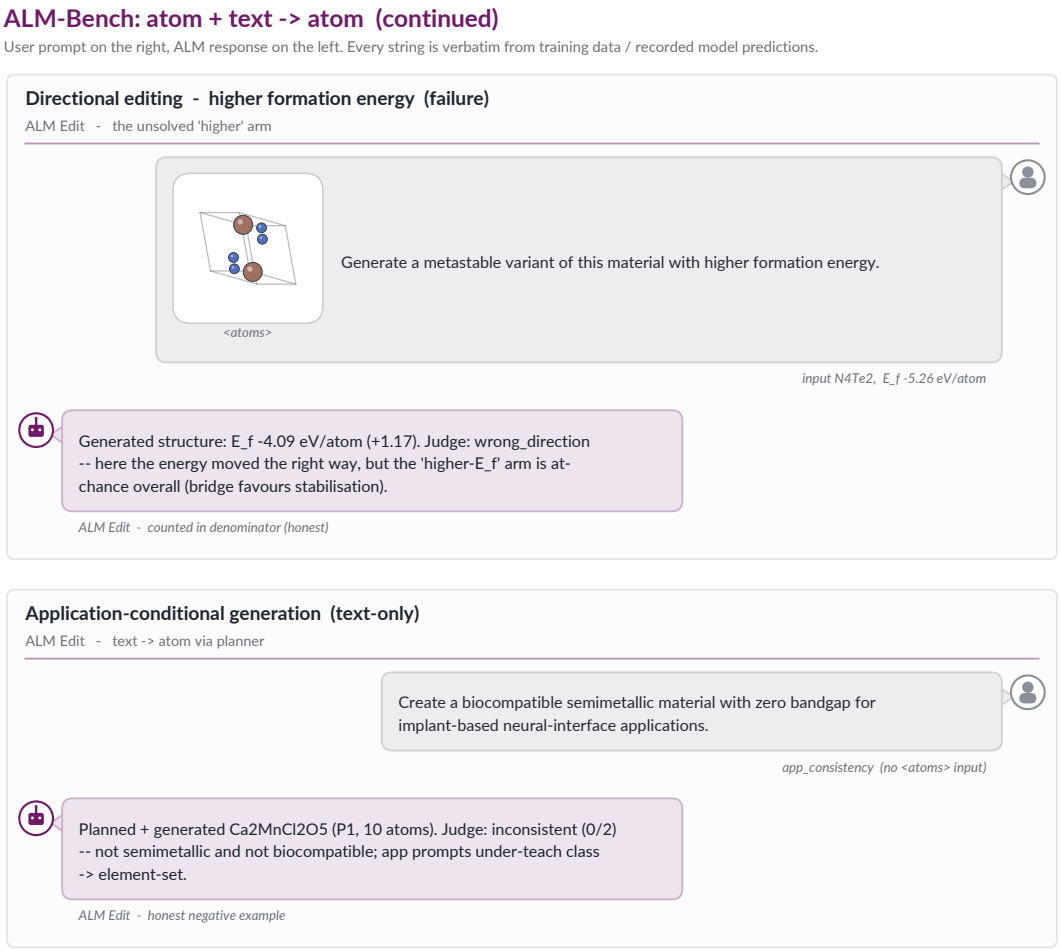

Figure 26: **ALM-Bench chat examples (continued).**

Table 17: **ALM Bench generation tasks with additional baselines and planner.**

Model	Application	Describe (Comp.)	Describe (Struct.)	OOD (Comp.)	OOD (Struct.)
ALM Edit	0.323 ±0.020	0.730 ±0.031	0.412 ±0.041	0.641 ±0.032	0.316 ±0.022
+ Qwen3.5-9B planner	**0.512** ±0.067	**0.864** ±0.019	0.329 ±0.016	0.780 ±0.026	0.371 ±0.030
GPT-4o	0.131 ±0.042	0.279 ±0.019	0.121 ±0.024	0.212 ±0.029	0.063 ±0.017
GPT-4.1	0.224 ±0.032	0.254 ±0.019	0.090 ±0.018	0.228 ±0.029	0.050 ±0.005
GPT-5.2	0.252 ±0.051	0.356 ±0.014	0.162 ±0.013	0.366 ±0.026	0.108 ±0.008
GPT-5.6-sol	0.263 ±0.067	0.728 ±0.022	0.479 ±0.004	0.696 ±0.030	0.357 ±0.026
GPT-6-astra	0.382 ±0.045	0.855 ±0.015	**0.612** ±0.011	**0.801** ±0.022	**0.455** ±0.015

the paper-defined "good model" threshold (ratios < 1 are reported only for completeness). Being scale-free, the single ≥ 5 threshold is comparable across all 28 property slices.

Mean Absolute Error (MAE). Raw $\frac{1}{n}\sum_i |\hat{y}_i - y_i|$ in the property's native units (eV, eV/atom, g/cm^3, Å, etc.); *lower is better*. MAE is the quantity reported against external regression baselines (Mat2Props, MatText, GNoME-FE, the MatterChat regression tasks) where those baselines publish MAE rather than a skill ratio. As a result, we also show MAEs for selected property prediction tasks in Table 19.

Table 18: **ALM Bench editing tasks with additional baselines.** Columns as in Table 3.

Model	$E_f \uparrow$	$E_f \downarrow$	$\rho \uparrow$	$\rho \downarrow$	$V \uparrow$	$V \downarrow$	Polymorph	Doping	Strain
ALM Edit	**0.586** ±0.024	0.626 ±0.026	**0.380** ±0.011	**0.370** ±0.040	**0.451** ±0.059	**0.355** ±0.033	0.224 ±0.022	**0.879** ±0.012	**0.151** ±0.028
GPT-4o	0.505 ±0.043	0.469 ±0.057	0.024 ±0.022	0.127 ±0.013	0.081 ±0.030	0.018 ±0.015	0.040 ±0.007	0.007 ±0.002	0.000 ±0.000
GPT-4.1	0.465 ±0.032	0.496 ±0.025	0.007 ±0.009	0.239 ±0.034	0.276 ±0.035	0.040 ±0.016	0.083 ±0.013	0.003 ±0.004	0.000 ±0.000
GPT-5.2	0.437 ±0.073	0.414 ±0.042	0.058 ±0.026	0.244 ±0.019	0.006 ±0.005	0.032 ±0.015	0.118 ±0.023	0.002 ±0.002	0.000 ±0.000
GPT-5.6-sol	0.513 ±0.046	0.636 ±0.032	0.220 ±0.019	0.238 ±0.019	0.050 ±0.021	0.028 ±0.005	0.444 ±0.043	0.004 ±0.002	0.000 ±0.000
GPT-6-astra	0.550 ±0.025	**0.723** ±0.056	0.233 ±0.028	0.256 ±0.021	0.105 ±0.015	0.130 ±0.015	**0.638** ±0.029	0.001 ±0.002	0.000 ±0.000

Table 19: **Raw MAE on LLM4Mat-Bench**: ALM Core (canonical understanding r=128+IT) vs CIF structure-input baselines, in the physical units of the unit row. Lower is better; **bold** marks columns ALM Core wins. These are the absolute errors behind the skill ratios of Table 19; generative-LLM baselines are omitted as there.

	MP				JARVIS		OQMD		GNoME		hMOF			
Model	E_f eV/at	gap eV	E_h eV/at	ρ g/cc	E_f eV/at	E_h eV/at	E_f eV/at	gap eV	E_f eV/at	gap eV	void –	LCD Å	PLD Å	CO_2 mol/kg
ALM Core	0.070	**0.300**	0.052	0.212	0.078	0.063	0.056	0.093	0.034	0.051	**0.057**	**1.25**	**1.43**	3.25
CGCNN (CIF)	0.123	0.366	0.058	0.246	0.063	0.170	0.033	0.062	0.014	0.045	0.062	1.76	2.04	1.31
MatBERT (CIF)	0.091	0.348	0.059	0.207	0.084	0.055	0.053	0.058	0.020	0.042	0.110	2.27	2.42	1.58
LLM-Prop (CIF)	0.070	0.317	0.103	0.156	0.066	0.101	0.039	0.129	0.017	0.098	0.095	2.00	2.50	1.44

Leak rate. The fraction of rows whose generation is flagged by the parser as a Markdown/image-URL emission (`![...]` or an imgur/`materialsproject.org` link), rather than a materials answer. Leaks count as property prediction failure (never excluded from the denominator).

Accuracy (multiple choice / classification). For MaScQA, Mat2MCQ, and the MatterChat classification tasks, fraction-correct accuracies are reported. For MatterChat's five classification tasks we use weighted-F1 (w-F1) to account for class imbalance (e.g. *is_metal*, *is_magnetic*). MaScQA's numerical-answer items are scored by MAE.

MatterChat 9-task scoring. The MatterChat benchmark (Tang et al., 2026) bundles five classification tasks (including crystal system, `is_magnetic`, and `is_metal`) scored by w-F1 (↑) and three regression tasks (formation energy and energy-above-hull in eV/atom, band gap in eV) scored by MAE/RMSE (↓), n=1000 test rows per task. ALM and all four MatterChat-paper variants are free-form text-generation models (cross-entropy on token probabilities, no per-task regression heads), so the comparison is output-mechanism-matched; the architectural difference is the input-side structural-encoder path and the dominant confound is per-task training exposure, not the head.

LLM judge. The LLM *Judge* column of Table 1 tests something the three accuracy benchmarks do not — whether crystal-specialization erodes the model's ability to *discuss* materials in natural language. We pose a fixed, in-house set of 190 materials-science questions: 130 short closed-form recall probes (e.g. the general perovskite formula ABX_3; magnetite = Fe_3O_4; the cubic-perovskite space group $Pm\bar{3}m$; the B-site location in ABO_3) and 60 open-ended free-text questions (e.g. describe the ABO_3 perovskite structure; why B-site doping shifts the electronic structure; what makes a good thermoelectric). Each answer is decoded without thinking enabled (64 tokens for closed-form responses, 128 for open-ended) and graded by `gpt-4o-mini` (temperature 0) on a 0–2 rubric: **2** = correct *and* coherent; **1** = partially correct, with a minor error; **0** = wrong, hallucinated, incoherent, empty, or stuck in a repetition loop. Degenerate or malformed responses are scored to 0. Alongside the mean score (normalized to $[0, 1]$ in Table 1) we track an independent degeneracy diagnostic, *loop-rate*: the fraction of answers whose most-frequent 4-gram repeats ≥ 4 times. Figures 27 and 28 demonstrate examples of the LLM judge.

E.2 Crystal structure prediction metric details

This section fixes the exact, reproducible definitions of the CSP metrics reported for **ALM Edit** throughout the paper.

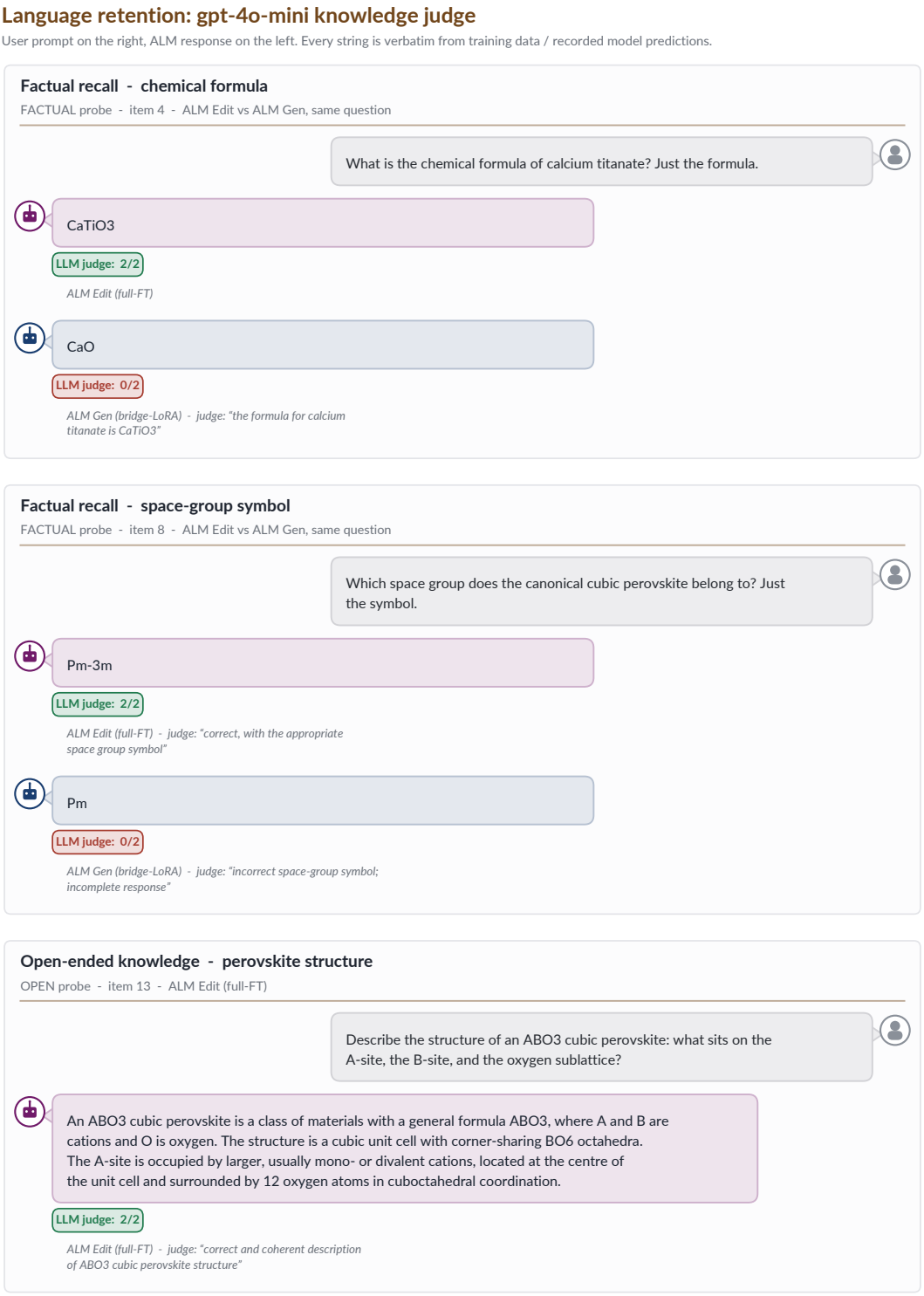

Figure 27: **Materials-knowledge retention judge: verbatim graded exchanges.**

Matcher. A generated structure is declared a match to its reference if pymatgen's `StructureMatcher` (wrapped by MatterGen's `OrderedStructureMatcher`) returns a fit at the CDVAE/CrystaLLM tolerances (Xie et al., 2022; Antunes et al., 2024), `ltol`=0.3,

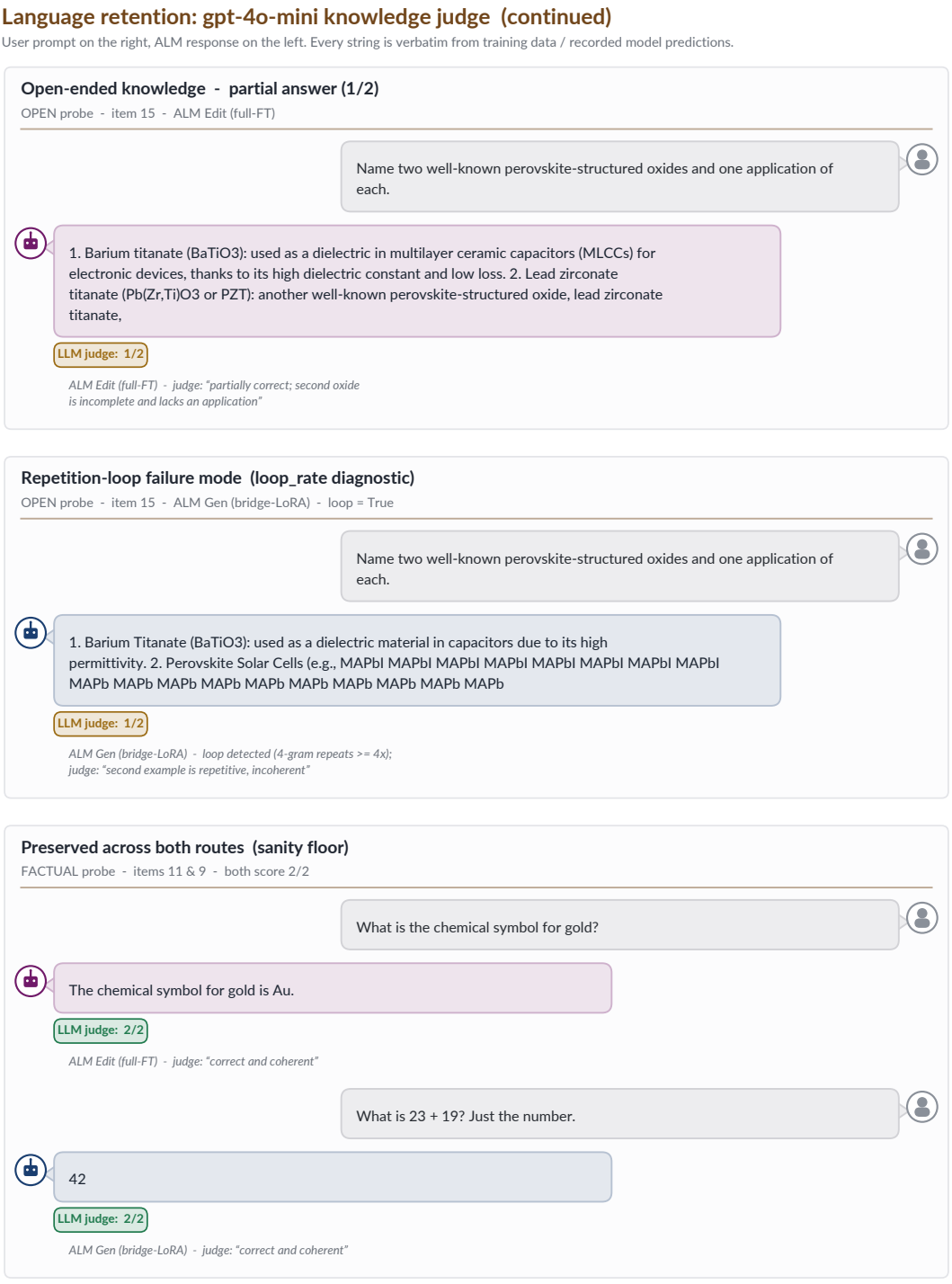

Figure 28: **Materials-knowledge retention judge (continued).**

`stol`=0.5, `angle_tol`=10. The match test is `matcher.fit(gen, ref)` and the matched-pair RMSD is read from `matcher.get_rms_dist(gen, ref)`, which returns a $(\mathrm{rms}, \mathrm{max_dist})$ pair. Following CDVAE, we report the rms component, in Å, normalized as pymatgen normalizes by $\left(V/N_p\right)^{1/3}$ (cell volume per atom).

Per-target aggregation: Match@1, Match@K, RMSE@1, RMSE@K. For one reference we draw K i.i.d. generations $\{g_1, \ldots, g_K\}$ (the "n=K generations per row" of Table 2) and compute:

- **Match@1** — indicator that the *first* generation g_1 matches the reference. **RMSE@1** is the matched RMSD of g_1 on a successful match.
- **Match@K** — indicator that *any* of the K generations matches the reference. **RMSE@K** is the *minimum* RMSD over the matched subset $\min_{i:\, g_i \text{ matches}} \mathrm{rms}(g_i, \mathrm{ref})$.

Reported Match@1/Match@K are the means of these indicators over the n_{rows} targets; reported RMSE@1/RMSE@K are means over matched rows only. CSP rows are reproduced verbatim from Table 1 of Seong et al. (2026).

E.3 *De novo* GENERATION METRIC DETAILS

ALM Gen is still a fundamentally conditional model, as it requires a prompt to the base language model to produce an output. However, its conditioning is deliberately weak, producing structures that are biased towards, but do not exactly follow, inputted prompts. To measure DNG performance, natural-language structural narratives are sampled from the GPT-Narratives (Park et al., 2024) dataset (`dft_3d`, `mp_3d_2020`, `aflow2`, and `oqmd` narratives).

Matcher. Uniqueness U and novelty N are computed by the matcher against the generated set (for U) and against the MP-2020 reference set (for N). To determine whether or not a generated structure matches a set of reference structures, we use MatterGen's `DefaultDisorderedStructureMatcher` with default tolerances ℓ_{tol}=0.2, s_{tol}=0.3, angle tolerance 5°, the same matcher used by `mattergen-evaluate` and by the LeMat-GenBench protocol.

Stability. The energy-above-hull E_{hull} used by each stability gate in Table 21 is the MP-2020-corrected hull energy from a MatterSim single-point relaxation. We report stability at three E_h cut-offs, applied to the same generated structures:

- $E_{\text{hull}} \leq 0.10$ **eV/atom: Metastable** (MS) $\rightarrow$ **MSUN.** This is the gate that MatterGen (Zeni et al., 2025), and Crys-JEPA (Liu et al., 2026) label "stable," which we show in Table 20.
- $E_{\text{hull}} \leq 0.016$ **eV/atom: Stable** (S) $\rightarrow$ **SUN.** This is the stricter CrystalReasoner (Wu et al., 2026) convention.
- $E_{\text{hull}} \leq 0$**: strict Stable** $\rightarrow$ **strict SUN.** This is the stability metric for LeMat-GenBench (Betala et al., 2026).

In addition, for LeMat-GenBench, E_{hull} is evaluated and averaged by a three-MLIP ensemble (MACE-MP + UMA + Orb-V3), each building a self-consistent convex hull from the broader-chemistry **LeMat-Bulk** reference. All generations are pre-relaxed before scoring.

E.4 REPRESENTATIONAL ALIGNMENT METRICS

The representational analysis of Figure 6B quantifies how information is shared across **ALM Edit**'s internal latent spaces. For N=2,000 **ALM Bench** prompts passed through **ALM Edit**, four representations f are extracted: the frozen OrbV3 node-wise embeddings, the output of the MLP that projects each token into the LLM embedding space (i.e., turning them into soft tokens), the language model's K=8 atomistic token embeddings, and the language-to-atomistic producer embedding output $\mathbf{C}$. Their pairwise alignment is measured with two complementary metrics, both bounded to $[0, 1]$: information imbalance (global, asymmetric) (Glielmo et al., 2022) and CKNNA (local) (Huh et al., 2024).

Information imbalance. This is an asymmetric measure of how much more information one representation holds than another (Glielmo et al., 2022), built on the premise that a representation's nearest-neighbor ranking is more informative than per-coordinate distances. Let r^f_{ij} be the nearest-neighbor rank of $f(x_j)$ with respect to $f(x_i)$ (rank 1 is the nearest), and let $c_f \approx r^f/N$

Table 20: **De-novo generation on MP-20, metastable-MS convention** ($E_{\text{hull}} \leq 0.10$ eV/atom).

Method	MS (%) ↑	N (%) ↑	**MSUN** (%) ↑
SymmCD†	34.7 ±1.4	85.1 ±1.5	19.0 ±0.9
SGEquiDiff†	46.5 ±1.9	74.8 ±1.2	23.5 ±0.9
DiffCSP++†	39.7 ±2.0	82.8 ±1.0	23.8 ±1.3
CrysLLMGen-7B†	35.1 ±1.9	87.4 ±1.1	22.9 ±0.9
FlowMM†	40.8 ±2.0	83.1 ±1.1	25.3 ±1.6
FlowLLM†	36.5 ±1.5	86.4 ±1.5	25.1 ±0.6
CDVAE (Xie et al., 2022)†	29.9 ±1.2	**96.5** ±0.6	27.0 ±1.3
DiffCSP (Jiao et al., 2023)†	45.9 ±1.8	83.6 ±0.6	30.9 ±1.8
ADiT†	69.5 ±1.1	58.9 ±0.9	30.3 ±0.8
MatterGen (Zeni et al., 2025) (*Crys-JEPA reprod.*)†	47.0 ±1.1	86.6 ±1.0	34.6 ±1.2
Crys-JEPA-full†	**76.6** ±1.1	83.3 ±1.1	**45.2** ±1.4
MatterGen-Base (our reprod., g=0 uncond)	71.9 ±1.2	62.5 ±0.8	36.8 ±1.1
ALM Gen (g=0.5)	68.3 ±2.6	67.8 ±1.9	39.0 ±3.5
ALM Gen (g=0.5) + **T2C-FK**	65.1 ±3.0	58.9 ±3.9	23.9 ±1.9

Table 21: ***De novo* generation on LeMat-GenBench** (Betala et al., 2026; Seong et al., 2026) ($N = 2500$), in which strict stability is defined by $\bar{E}_{\text{hull}} < 0$ eV/atom and structures are pre-relaxed. Validity is measured by charge neutrality, physical plausibility, and minimum distance checks, and metastability by $E_{\text{hull}} < 0.1$. E_f, E_{hull}, and RMSD are scored by 3 MLIPs (further details and conventions in Appendix E.3).

				Energy-based			Strict Stability		Metastab.	
Model	Valid↑	Unique↑	Novel↑	E_f ↓	$\bar{E}_{\text{hull}}$ ↓	RMSD↓	Stable↑	SUN↑	Meta↑	MSUN↑
MatterGen (Zeni et al., 2025)	95.7	95.1	**70.5**	−0.70 ±0.79	0.18 ±0.18	0.39 ±0.50	2.0	0.2	33.4	15.0
PLaID++ (Xu et al., 2026)	96.0	77.8	24.2	−0.50 ±0.44	0.09 ±0.16	0.13 ±0.29	12.4	**1.0**	60.7	7.6
WyFormer (Kazeev et al., 2025)	93.4	93.0	66.4	−0.43 ±0.95	0.50 ±0.51	0.81 ±0.98	0.5	0.1	15.7	1.9
WyFormer-DFT	95.2	95.0	66.4	−0.67 ±0.91	0.27 ±0.36	0.42 ±0.60	3.7	0.4	24.8	7.8
MCFlow-S (Seong et al., 2026)	97.2	**96.3**	52.2	−0.85 ±0.84	0.10 ±0.12	0.16 ±0.27	11.7	0.7	49.5	18.9
MCFlow-B (Seong et al., 2026)	97.7	95.5	25.4	−0.91 ±0.85	0.05 ±0.10	0.08 ±0.18	17.6	0.7	64.3	11.9
MCFlow-L (Seong et al., 2026)	**98.6**	95.2	18.6	**−0.93** ±0.87	**0.04** ±0.08	**0.06** ±0.15	**18.8**	0.5	**68.3**	9.3
ALM Gen	92.2	91.3	61.5	−0.44 ±0.06	0.09 ±0.00	0.20 ±0.01	3.6	0.8	58.7	**35.2**

be the associated copula (cumulative) variable. The information imbalance from representation f to g is

$$\Delta(f \to g) \;=\; 2 \lim_{\epsilon \to 0} \big\langle\, c_g \,\big|\, c_f = \epsilon \,\big\rangle, \tag{29}$$

the average g-rank of the points that are nearest neighbors in f. Because $r^f_{ij} \neq r^g_{ij}$, the pair $\big(\Delta(f \to g),\, \Delta(g \to f)\big)$ is asymmetric: both near 0 (bottom-left of an II plot) means the two spaces encode identical information; both near 1 (top-right) means orthogonal information; $\Delta(f \to g) \approx 0$ with $\Delta(g \to f) \approx 1$ (top-left) means g is contained within f; and $\Delta \approx 0.5$ on both axes means the spaces share information without subsuming one another. Ranks are computed by exact cosine k-nearest-neighbor ordering over all N points (full-rank, ii-k=50).

Centered kernel nearest-neighbor alignment (CKNNA). This is a local latent space similarity that is high when two representations agree on which points are mutual nearest neighbors (Huh et al., 2024). With centered inner-product kernels $\bar{K}_{ij} = \langle f(x_i), f(x_j)\rangle - \mathbb{E}[\langle f(x_i), f(x_j)\rangle]$ and $\bar{L}_{ij}$ defined analogously for g, alignment is restricted to mutual k-nearest-neighbor pairs,

$$\text{Align}(K, L) \;=\; \sum_{i,j} \alpha(i,j)\, \bar{K}_{ij}\, \bar{L}_{ij}, \tag{30}$$

where $\alpha(i,j)$=1 iff $f(x_j)$ is among the k nearest neighbors of $f(x_i)$ *and* $g(y_j)$ is among the k nearest neighbors of $g(y_i)$, and 0 otherwise; CKNNA is the normalized form $\text{Align}(K,L) \,/\, \sqrt{\text{Align}(K,K)\,\text{Align}(L,L)}$, bounded to $[0,1]$. We use k=25. II and CKNNA agree across all representation pairs (Figure 6B), so the global information-content reading is corroborated by local neighborhood structure.