FlowLLM: Flow Matching for Material Generation with LLMs as Base Distributions

Anuroop Sriram, Benjamin Kurt Miller, Ricky T. Q. Chen, Brandon M. Wood

November 15, 2024

Generative Modeling for Material Discovery

- Discovering new, stable materials is a key challenge in material science.
- Prior generative methods used denoising methods (diffusion, flow matching), or large language models (LLMs), which have complementary strengths.
  - LLMs excel at generating discrete variables (atom types).
  - Denoising methods excel at continue values.
- *Question: How do we get the best of both worlds?*

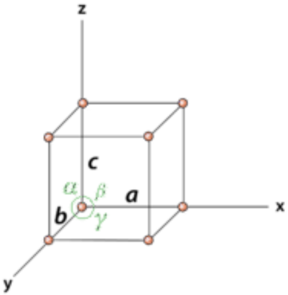
Generative Modeling for Material Discovery

- Discovering new, stable materials is a key challenge in material science.
- Prior generative methods used denoising methods (diffusion, flow matching), or large language models (LLMs), which have complementary strengths.
  - LLMs excel at generating discrete variables (atom types).
  - Denoising methods excel at continue values.
- *Question: How do we get the best of both worlds?*
  - We introduce **FlowLLM**, a simple yet effective method to combine LLMs and Riemannian Flow Matching (RFM).

Crystal Representation

A crystals with $n \in \mathbb{N}$ atoms can be represented as: $\boldsymbol{c} := (\boldsymbol{a}, \boldsymbol{f}, \boldsymbol{l}) \in \mathcal{C}$, consisting of:

- Lattice, $\boldsymbol{l}$, defined using three side lengths $(a, b, c) \in \mathbb{R}^+$ in Å, and three internal angles $(\alpha, \beta, \gamma) \in [60°, 120°]$.
- Atom types are categorical vectors: $\boldsymbol{a} := \left[a^1, \ldots, a^n\right]$, where $a^i \in \mathcal{A}$.
- Atom positions represented using fractional coordinates on a flat torus: $\boldsymbol{f} := \left[f^1, \ldots, f^n\right]$, $f^i \in \mathcal{F} = \mathbb{T}^3$. The positions *"wrap around"* the unit cell.



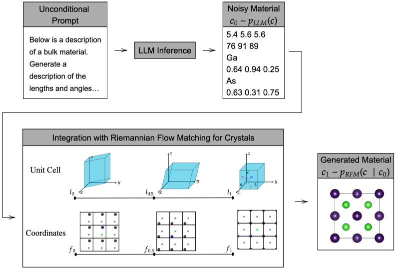
FlowLLM Model

FlowLLM generates materials via a two step process – it first samples an initial, noisy sample from an LLM, followed by an iterative refinement process using an RFM model:

$$\boldsymbol{c}_0 \sim p_{\mathrm{LLM}}(\boldsymbol{c};\theta_0), \qquad (1)$$

$$\boldsymbol{c}_1 \sim p_{\mathrm{RFM}}(\boldsymbol{c}|\boldsymbol{c}_0;\theta_1) \qquad (2)$$

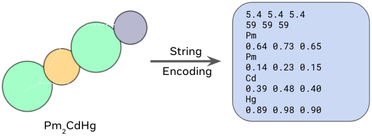
The LLM serves as the learned prior distribution for the RFM.



Large Language Model for Base Distribution

To train the LLM part of the model, we represent crystals using a text representation, and fine-tune a LLAMA-2 model on this.

This closely follows Crystal-LLM[1].

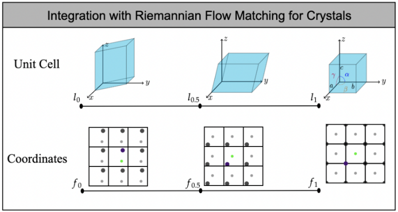


[1] Gruver et al. "Fine-Tuned Language Models Generate Stable Inorganic Materials as Text", ICLR 2024

Denoising with Riemannian Flow Matching

The output of the LLM is refined using Riemmannian Flow Matching with a suitable product manifold to represent crystals following FlowMM[2].

- Atom positions are represented on a flat torus, and lattice parameters in euclidean space. Atom types are kept fixed.
- We use an equivariant GNN for the velocity function.



[1] Miller et al. "FlowMM: Generating Materials with Riemannian Flow Matching", ICML 2024

FlowLLM Model Training

We train the our model in 3 steps:

1. The LLM is first trained independently to generate a text representation of the material, with suitable prompting by fine-tuning a 70B parameter LLaMA-2 model.
2. Next, we create a paired dataset of $\{(\boldsymbol{c}_0, \boldsymbol{c}_1)\}$ samples, where each base distribution sample, $\boldsymbol{c}_0$ is sample from the LLM with a prompt conditioned on the chemical formula of the corresponding target sample, $\boldsymbol{c}_1$.
3. Finally, the RFM is trained using a flow matching objective on this paired distribution.

Experiments

- We train our models on the MP-20 dataset ($\sim 40K$ materials).
- Key Metrics are **Stability Rate** (percentage of generated structures that are stable) and **SUN rate** (percentage that are stable, unique and nove).

Method	Type	Stability Rate(%)↑	SUN Rate(%) ↑
CDVAE	Diffusion	1.57	–
DiffCSP	Diffusion	5.06	3.34
FlowMM	Flow Matching	4.65	2.34
CrystalLLM (70B)	LLM	5.28	–
FlowLLM(Ours)			
$\tau = 1.0, P = 0.9$	LLM + Flow Matching	10.07	4.89
$\tau = 0.7, P = 1.0$	LLM + Flow Matching	13.03	4.88
$\tau = 0.7, P = 0.9$	LLM + Flow Matching	**17.82**	**4.92**

FlowLLM significantly outperforms prior methods!

Thank you

Check out our poster, paper, and code!

Paper

Code