# MSP-LLM: A UNIFIED LARGE LANGUAGE MODEL FRAMEWORK FOR COMPLETE MATERIAL SYNTHESIS PLANNING

**Heewoong Noh**[1,✉], **Gyoung S. Na**[1,2,✉], **Namkyeong Lee**[1,✉], **Chanyoung Park**[1,✉*]

[1]KAIST, [2]KRICT

## ABSTRACT

Material synthesis planning (MSP) remains a fundamental and underexplored bottleneck in AI-driven materials discovery, as it requires not only identifying suitable precursor materials but also designing coherent sequences of synthesis operations to realize a target material. Although several AI-based approaches have been proposed to address isolated subtasks of MSP, a unified methodology for solving the entire MSP task has yet to be established. We propose MSP-LLM, a unified LLM-based framework that formulates MSP as a structured process composed of two constituent subproblems: precursor prediction (PP) and synthesis operation prediction (SOP). Our approach introduces a discrete material class as an intermediate decision variable that organizes both tasks into a chemically consistent decision chain. For SOP, we further incorporate hierarchical precursor types as synthesis-relevant inductive biases and employ an explicit conditioning strategy that preserves precursor-related information in the autoregressive decoding state. Extensive experiments show that MSP-LLM consistently outperforms existing methods on both PP and SOP, as well as on the complete MSP task, demonstrating an effective and scalable framework for MSP that can accelerate real-world materials discovery.

## 1 INTRODUCTION

Recently, Artificial Intelligence (AI) has achieved notable successes in condensed matter physics and materials science, primarily by enabling a high-throughput screening at the scales of real-world materials spaces Wang et al. (2023); Kim et al. (2025); Du et al. (2023). After exploring the materials space, we are then confronted with the next scientific challenge of designing a synthesis plan to realize the discovered materials Szymanski et al. (2023a). However, most existing machine learning methods have primarily focused on exploring the materials space while overlooking the next challenge in **material synthesis planning (MSP)** Jansen (2002); Szymkuć et al. (2016), and this oversight remains a bottleneck of AI for science in real-world materials discovery. Although numerous synthesis planning methods have been developed for organic compounds, benefiting from well-specified molecular structures and well-defined reaction rules, MSP remains a long-standing challenge in AI for science due to two reasons: (1) The lack of atomic structures for the product materials causes multiple synthesis paths on polymorphic structures, thereby substantially increasing uncertainty in MSP Na (2023); Okabe et al. (2024). Moreover, (2) fully specifying the synthesis paths based solely on a target material's composition remains a challenge, as compositional data alone fails to provide a comprehensive blueprint for the necessary chemical reactions Noh et al. (2024); Prein et al. (2025b).

In real-world applications, MSP is explored within a materials space defined at the level of chemical compositions Na (2023); Okabe et al. (2024); Choi et al. (2025), and requires predicting a set of precursor materials $P$ and a sequence of synthesis operations $O$ for a given desired product material. Here, $P \in 2^{\mathcal{M}} \setminus \{\emptyset\}$ denotes a set of precursor materials, where $\mathcal{M}$ denotes the set of materials, and $O \in \mathcal{O}^*$ denotes a finite-length synthesis operation sequence. Formally, the MSP task can be defined as $f : \mathcal{M} \to 2^{\mathcal{M}} \times \mathcal{O}^*$, which maps a given product material to a set of precursor materials and a corresponding synthesis operation sequence. However, due to the inherent challenges of MSP, most existing studies have addressed only partial aspects of MSP, rather than providing a framework

*Corresponding author (cy.park@kaist.ac.kr)

that solves MSP in its entirety. In particular, prior works typically focus on one of the following two subproblems: (1) **precursor prediction (PP)**, which predicts a set of precursor materials from a given product material, $f_p : \mathcal{M} \rightarrow 2^{\mathcal{M}}$, or (2) **synthesis operation prediction (SOP)**, which aims to predict a sequence of synthesis operations given a product material and a set of precursor materials, $f_o : \mathcal{M} \times 2^{\mathcal{M}} \rightarrow \mathcal{O}^*$.

Among these subproblems, SOP remains largely underexplored and has been studied only in narrowly scoped settings, typically restricted to specific material systems rather than general MSP Na (2023); Pan et al. (2025). By contrast, most existing efforts have focused on PP, yet these approaches often rely on complex retrieval pipelines He et al. (2023); Noh et al. (2024) or access to proprietary LLM APIs Kim et al. (2024); Prein et al. (2025b), which limits their scalability and reproducibility. These limitations underscore the need for the scalable and unified modeling framework that can address both SOP and PP within a general MSP setting.

Recently, large language models (LLMs) fine-tuned for domain-specific scientific tasks have demonstrated strong capabilities across scientific domains, particularly in materials science, including generation of candidate materials Gruver et al. (2024) and emulation of scientific discovery processes Ding et al. (2024). This modeling framework is particularly well-suited to MSP, where all relevant input information is naturally represented as text in the form of chemical compositions Prein et al. (2025b). By leveraging LLMs that have been pre-trained on large-scale and diverse textual corpora, LLM-based MSP models can potentially access and utilize extensive background knowledge, enabling more informed and physically grounded decisions Bran et al. (2025); Rueda et al. (2025) in both precursor selection and synthesis operation planning.

However, despite their promise, effective strategies for fine-tuning LLMs to address MSP remain largely unexplored. Specifically, no existing work has established a principled fine-tuning strategy that enables LLMs to perform PP and SOP in a unified MSP framework.

In this paper, we propose a unified LLM-based framework, called MSP-LLM, for solving complex MSP tasks. We develop novel task-specific LLM fine-tuning strategies for the two constituent subproblems of MSP, i.e., PP and SOP. Specifically, (i) we introduce a discrete material class variable that captures synthesis-relevant characteristics of materials and employ it in both PP and SOP as an intermediate decision variable. By incorporating this variable, each subtask is organized into a structured decision chain, in which the model conditions its task-specific generation on the material class variable, leading to more accurate predictions. Furthermore, for SOP, we introduce an explicit conditioning strategy that integrates hierarchical precursor types as synthesis-relevant inductive biases and enforces their effective utilization through precursor constraint factorization, promoting precursor-conditioned operation generation and more reliable retention of precursor constraints in the decoder state. By integrating LLMs fine-tuned for each subproblem within a single coherent framework, MSP-LLM enables complete MSP from target material compositions. Extensive experiments demonstrate that MSP-LLM significantly outperforms existing methods on both individual subproblems and the overall MSP task, highlighting its effectiveness and practical applicability to real-world materials discovery.

To the best of our knowledge, MSP-LLM is the first work to formulate and tackle the complete MSP task, and to show that it can be effectively addressed by fine-tuning LLMs in a unified framework.

In this study, we make the following contributions:

- We propose MSP-LLM , a unified framework for general Material Synthesis Planning (MSP) that formulates the task as a two-stage process consisting of (1) precursor prediction (PP) and (2) synthesis operation prediction (SOP), each implemented using fine-tuned LLMs tailored to the respective subtask.
- By incorporating domain-specific inductive biases via a material group decision chain in both PP and SOP, together with an explicit conditioning strategy, our framework effectively leverages precursor information in SOP. We further analyze the mechanisms underlying the resulting performance gains.
- Through extensive experiments, we demonstrate that the proposed framework significantly outperforms all baselines on the complete MSP task as well as its subtasks, highlighting the superiority of our approach.

The source code for MSP-LLM is available at https://github.com/HeewoongNoh/MSP-LLM-Official.

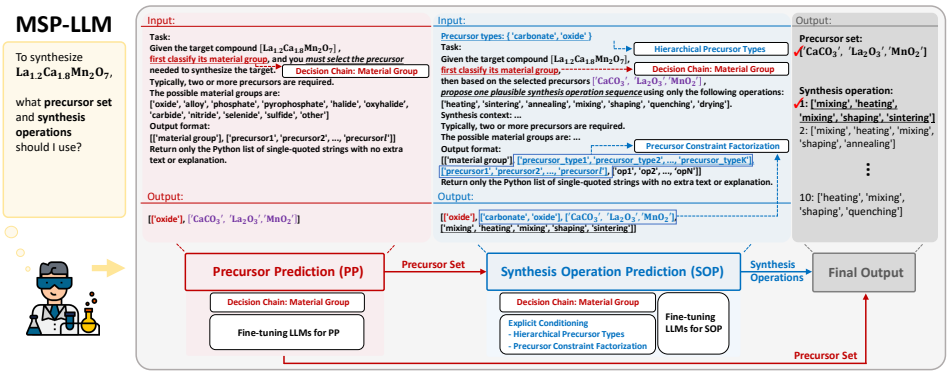


Figure 1: Overall framework of MSP-LLM. Given a target material, MSP-LLM first predicts the precursor set (PP), and then, conditioned on the predicted precursors, generates the synthesis operation sequence (SOP), resulting in a complete MSP. The input–output texts above the PP and SOP LLMs illustrate example prompts used during LLM fine-tuning for each task.

## 2 Preliminaries

**Material Synthesis Planning (MSP).** As defined in Eq. (1), MSP associates a target material with a precursor set and a synthesis operation sequence. We adopt a two-stage formulation that emulates the experimental workflow of synthesis chemists who first identify feasible precursors and then design operations conditioned on them. While a one-stage approach can, in principle, learn to output both precursors and operations simultaneously, such predictions collapse material selection and process design into a single, opaque step that is difficult to verify, adapt, or safely execute in practice Tang et al. (2025). By explicitly separating these decisions, the two-stage formulation better aligns with real-world material synthesis workflows, enabling more verifiable, and practically deployable planning.

**Precursor Prediction (PP).** This task aims to identify a set of starting materials required to synthesize a given target material. In this task, only the chemical composition of the target material $m$ is provided, without any explicit structural information, and the goal is to predict an appropriate precursor set $P$. For example, when the target material is $\mathrm{La_{1.2}Ca_{1.8}Mn_2O_7}$, the task is to predict a precursor set $\{\mathrm{CaCO_3}, \mathrm{La_2O_3}, \mathrm{MnO_2}\}$. Details of the preprocessing procedure for precursors are provided in Appendix C.

**Synthesis Operation Prediction (SOP).** This task focuses on determining the sequence of processing steps required to synthesize the target material, given both the target material and its corresponding precursor set. We define the synthesis operation as the sequence $O = (o_1, o_2, \ldots, o_n)$, where each $o_t$ is selected from a predefined set of seven canonical solid-state synthesis operations: *mixing*, *drying*, *heating*, *sintering*, *annealing*, *quenching*, and *shaping*. A valid synthesis route may involve repeating certain operations multiple times, and the objective is to predict the ordered sequence $O$. For instance, in the case of $\mathrm{La_{1.2}Ca_{1.8}Mn_2O_7}$ with the precursor set $\{\mathrm{CaCO_3}, \mathrm{La_2O_3}, \mathrm{MnO_2}\}$, a plausible synthesis sequence is *mixing* $\rightarrow$ *heating* $\rightarrow$ *mixing* $\rightarrow$ *shaping* $\rightarrow$ *sintering*. Details of the preprocessing of synthesis operations are provided in Appendix C.

## 3 Proposed Method

We develop task-specific LLM fine-tuning strategies for the two constituent subproblems of MSP: PP and SOP. To structure these tasks, we introduce a discrete material class variable that induces a decision-chain formulation (Section 3.1). We then describe the PP stage in Section 3.2 and the subsequent SOP stage in Section 3.3. We further propose explicit conditioning with hierarchical precursor types, which reinforces precursor-related constraints in the decoder state (Section 3.3.1), and provide an information-theoretic analysis explaining the effectiveness of this design (Section 3.5). Finally, we demonstrate the entire MSP with MSP-LLM (Section 3.4) by linking the independently trained PP and SOP models in a sequential manner, where precursors predicted in the PP stage are used as conditioning inputs for the SOP stage, forming a complete MSP pipeline.

### 3.1 Decision Chain via Discrete Material Class Variable

Materials often exhibit synthesis-related characteristics that are correlated with their chemical classes Rao & Gopalakrishnan (1997); West (2022). While a material group does not uniquely determine a specific synthesis pathway, materials belonging to different groups (e.g., oxides, phosphates, nitrides, sulfides) often exhibit systematic differences in precursor selection, decomposition behavior, and synthesis operations Schubert & Hüsing (2019). In practice, domain experts attempt to identify the chemical group of a target material to determine an appropriate synthesis strategy. Inspired by this expert reasoning process, we introduce a decision chain framework that utilizes the material group, denoted as $G$, as a discrete intermediate variable for both Precursor Prediction (PP) and Synthesis Operation Prediction (SOP). In the PP task, given a target material $m$, the model first predicts its material group and subsequently generates its precursor set $P$ conditioned on the predicted material group. Similarly, for SOP, the model predicts the material group $G$ based on the target material $m$ and the predicted precursor set $P$ before generating the synthesis operation sequence $O$. Accordingly, both tasks are formulated as structured decision chains that first infer the material group and then perform task-specific generation. To implement this decision chain, we curate input prompts that explicitly instruct the model to **"first classify its material group"** before proceeding to the PP/SOP. Consequently, while fine-tuning the LLM, it learns to perform material group classification as a prerequisite step. This approach introduces a task-specific inductive bias, effectively constraining the search space for precursors and operations to a chemically plausible subspace. Note that PP and SOP are implemented using independently fine-tuned LLMs, and are subsequently integrated within a unified framework where the material group decision chain is applied to both stages.

### 3.2 Precursor Prediction (PP) Stage

For the PP stage, we design a task-specific prompt to fine-tune the LLM such that the model first classifies the material group $G$ from the target material $m$ and subsequently selects an appropriate precursor set $P$. Let $X_{\text{PP}}$ denote the complete input prompt, which includes the task description and the chemical formula of the target material $m$. We train the model using a standard language modeling (LM) objective under teacher forcing over the entire output sequence. The target sequence is defined as $\mathbf{y}_{\text{PP}} = (G, P)$, where $G$ denotes the ground-truth material group and $P$ denotes the corresponding precursor set of the target material $m$. Formally, the training loss is given by:

$$\mathcal{L}_{\text{LM}}^{\text{PP}} = -\sum_{t=1}^{|\mathbf{y}_{\text{PP}}|} \log p_\theta\big(y_t \mid X_{\text{PP}}, \mathbf{y}_{\text{PP}<t}\big), \tag{1}$$

where $y_t$ is the token at position $t$, and $\mathbf{y}_{\text{PP}<t}$ denotes the prefix of the ground-truth sequence up to step $t-1$.

### 3.3 Synthesis Operation Prediction (SOP) Stage

#### 3.3.1 Challenges of Precursor Utilization

In SOP, precursors play a pivotal role in determining synthesis operation behavior, as they provide particularly informative cues for materials that require complex, multi-step synthesis procedures Rao & Gopalakrishnan (1997); West (2022); Schubert & Hüsing (2019). Intuitively, incorporating the precursor set into the input context is therefore expected to improve prediction accuracy compared to conditioning solely on the target material. To corroborate this, we compare two simple conditioning strategies applied at the LLM input stage: **target-only** and **implicit conditioning**, across two backbone LLMs, Qwen-2.5-7B and LLaMA-3.1-8B. The **target-only** approach conditions the LLM on the task description and the target material only, while **implicit conditioning** additionally incorporates the precursor set alongside the target material information to guide the model. We

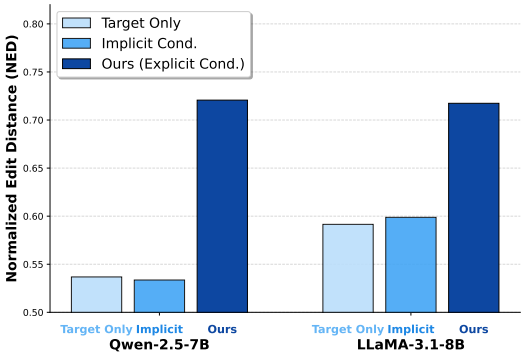


Figure 2: SOP performance under target-only, implicit, and explicit conditioning across two fine-tuned backbone LLMs.

evaluate SOP performance using Normalized Edit Distance (NED), which measures the token-level similarity between the predicted and ground-truth synthesis operation sequences (note that each synthesis operation is treated as a single token).

However, as illustrated in Figure 2, contrary to our initial expectations, **implicit conditioning** does not consistently outperform the **target-only** baseline, despite the inclusion of additional precursor information. This observation suggests that merely providing precursor information as a part of the LLM input is insufficient to ensure its effective utilization, especially in complex contexts—such as those involving long sequences or heterogeneous sequences of chemical formulas, where multiple precursor formulas are listed consecutively. Therefore, this motivates the need to explicitly enforce the preservation and utilization of precursor-related information throughout the decoding process.

To effectively leverage the precursor information, we introduce an **explicit conditioning** strategy comprising two key components: (i) extracting hierarchical precursor types from the fine-grained precursor set to capture high-level behavioral constraints (Section 3.3.2), and (ii) enforcing conditioning via precursor constraint factorization (Section 3.3.3). As shown in Figure 2, this design leads to substantial and consistent performance gains across both backbone models, highlighting the importance of **explicit conditioning** over naive implicit input-level conditioning. Note that, since the PP and SOP modules are trained independently, we use the ground-truth precursor set to condition the SOP module during the model training.

### 3.3.2 Hierarchical Precursor Types

In SOP, it is important not only to identify fine-grained operational dependencies, such as the ordering of steps and the need for repeated operations, but also to capture higher-level synthesis trends that govern the overall process. For example, carbonate precursors typically require an initial heating step to remove $CO_2$ prior to phase formation Fuertes (2018). This implies that the presence of a carbonate precursor provides a strong behavioral signal indicating the necessity of a heating step in the synthesis route. Building on this insight, we leverage a hierarchical abstraction of the precursor set in the form of precursor types, which capture key synthesis-relevant behaviors such as decomposition, volatility, and reactivity Schubert & Hüsing (2019); Rao & Gopalakrishnan (1997); West (2022). Let $P = \{p_1, \ldots, p_l\}$ denote the precursor set for a target material, and $p_i$ denote the $i$-th precursor, where $l$ is the number of precursors in $P$. Each precursor belongs to one of a finite set of precursor types (e.g., carbonate, nitrate, ammonium, phosphate, oxide, or other). We define a simple rule-based mapping function $\tau$ such that $Types = \tau(P)$, where $Types$ denotes the set of unique precursor types associated with the precursor set $P$. The resulting $Types$ provides a hierarchical, behaviorally meaningful abstraction of $P$ that aligns closely with typical synthesis operation patterns. We denote the comprehensive precursor information as $Z := (Types, P)$. For instance, given a precursor set $P = \{\mathrm{CaCO_3}, \mathrm{La_2O_3}, \mathrm{MnO_2}\}$, the corresponding unique $Types$ would be $\{\text{carbonate}, \text{oxide}\}$.

### 3.3.3 Precursor Constraint Factorization

To ensure the effective utilization of precursor information beyond implicit input-level context, we introduce *precursor constraint factorization (PCF)*, which enforces the model to utilize the precursor information. Unlike naive implicit conditioning, which offers limited training pressure for precursor-specific representations in the decoder state, PCF introduces an explicit constraint that facilitates more consistent utilization of precursor information during synthesis operation generation. Concretely, SOP is formulated as learning a joint conditional distribution over $(G, Z, O)$ given $X_{\text{SOP}}$, where $X_{\text{SOP}} = (\text{task prompt}, m, Z_{\text{in}})$ and $Z_{\text{in}}$ denotes the ground-truth precursor information provided as part of the input. This induces the following factorization:

$$\begin{aligned} p_\theta(G, Z, O \mid X_{\text{SOP}}) = p_\theta(G \mid X_{\text{SOP}})\, p_\theta(Z \mid X_{\text{SOP}}, G) \\ \cdot\, p_\theta(O \mid X_{\text{SOP}}, G, Z). \end{aligned} \tag{2}$$

This formulation provides a principled decomposition that first enables the model, through the term $p_\theta(G \mid X_{\text{SOP}})$, to identify the material group as an intermediate variable corresponding to the decision-chain formulation (Section 3.1). The subsequent term $p_\theta(Z \mid X_{\text{SOP}}, G)$ explicitly constrains the model to predict and utilize precursor information. Finally, the term $p_\theta(O \mid X_{\text{SOP}}, G, Z)$ trains the model to generate the synthesis operation sequence conditioned on both the inferred material group and the predicted precursor variables. We implement this factorization by structuring the output sequence as $\mathbf{y}_{\text{SOP}} = (G, Z, O)$ and optimizing a standard autoregressive language modeling objective,

$$\mathcal{L}_{\text{LM}}^{\text{SOP}} = -\sum_{t=1}^{|\mathbf{y}_{\text{SOP}}|} \log p_\theta(y_t \mid X_{\text{SOP}}, \mathbf{y}_{\text{SOP}<t}). \tag{3}$$

During training, the ground truth precursor information $Z_{\text{in}}$ is provided as part of the input context while simultaneously being modeled as an explicit random variable $Z$ in the output sequence. This requires the decoder state to remain sufficient for predicting $Z$ even when $Z$ is already observed in the input. Under this formulation, minimizing the SOP loss jointly constrains the model to accurately predict both $Z$ and $O$, thereby discouraging solutions in which precursor-specific information is not retained in the decoder representation during operation generation. Consequently, during operation generation, the decoder state must remain aligned with precursor-specific constraints, enforcing retention and utilization of $Z$ rather than treating precursor information as a passive component of the input context.

### 3.4 A Unified Framework for Complete MSP

Our framework addresses the complete MSP setting, in which only the target material $m$ is provided as input. Given this target, (1) the fine-tuned PP model first generates a ranked set of candidate precursor combinations. (2) Conditioned on a selected precursor set, the fine-tuned SOP model then produces coherent sequences of synthesis operations that specifies how to realize the target material using the predicted precursors. By integrating these steps, the framework effectively achieves the complete MSP task.

### 3.5 Why Precursor Constraint Factorization Works: An Information-Theoretic View.

For clarity, we omit $G$ and focus on the precursor information $Z$ and the operation sequence $O$. Let $S_t$ denote the decoder state at generation step $t$ and $o_t$ the next operation token. Please note that we distinguish between the ground truth precursor information $Z_{\text{in}}$, which is provided as part of the input prompt, and the structured precursor variable $Z := (Types, P)$, which denotes the same chemical information but is treated as an explicit random variable in our conditional factorization. We decompose the full input context as $X_{\text{SOP}} = (X, Z_{\text{in}})$, where $X = (\text{task prompt}, m)$ denotes the non-precursor context and $Z_{\text{in}}$ denotes the ground truth precursor information. This decomposition is used to distinguish between the two learning objectives: (i) implicit conditioning, which optimizes $-\log p_\theta(O \mid X_{\text{SOP}})$, and (ii) precursor constraint factorization, which optimizes $-\log p_\theta(Z, O \mid X_{\text{SOP}})$.

**Marginalization under implicit conditioning.** Under implicit conditioning, the model is trained to minimize $-\log p_\theta(O \mid X_{\text{SOP}})$, in which precursor information appears only as part of the input context and is not introduced as an explicit random variable. In contrast, our precursor constraint factorization trains the model to minimize $-\log p_\theta(Z, O \mid X_{\text{SOP}})$, thereby converting the precursor information present in the input to an explicit conditioning variable that must be predicted and utilized during generation. Because the implicit objective does not require the model to distinguish between different precursor-specific generation mechanisms (i.e., $Z_{in}$ and $Z$), it allows solutions that achieve low training loss while approximating a marginalized conditional distribution of the form $p(O \mid X)$. This marginalized behavior can be express as

$$p(O \mid X) = \sum_z p(O \mid X, Z{=}z)\, p(Z{=}z \mid X). \tag{4}$$

Consequently, distinct precursor-conditioned operation patterns are collapsed into a single aggregated distribution, and the decoder state is not required to retain instance-specific precursor information.

**Reduced intrinsic uncertainty.** The distinction between marginalized (i.e., $p(O \mid X)$) and precursor-conditioned generation (i.e., $p(O \mid X, Z)$) can be formalized using a basic property of conditional entropy $H(O \mid X, Z) \leq H(O \mid X)$, with equality if and only if $O \perp Z \mid X$. When precursor information is informative for operations, i.e., $I(O; Z \mid X) > 0$, explicitly modeling the conditional distribution $p(O \mid X, Z)$ admits a strictly lower irreducible uncertainty than the marginalized distribution $p(O \mid X)$. Such a lower-uncertainty requires the decoder state $S_t$ to utilize precursor constraints throughout generation.

As a result, the decoder is encouraged to maintain and utilize precursor-related information when generating synthesis operations, rather than relying on an aggregated, precursor-agnostic predictor. Under precursor constraint factorization, this requirement is enforced by training the model to predict $Z$ as part of the output sequence, which conditions operation generation on decoder states that utilize precursor-specific structure. This prevents precursor-conditioned generation mechanisms from collapsing into a marginalized, precursor-agnostic predictor, leading to lower intrinsic uncertainty and more consistent, precursor-aware synthesis operation sequences in practice.

# 4 EXPERIMENTS

We evaluate the effectiveness of MSP-LLM across three settings: the PP task (Section 4.1), which predicts precursor sets from a target material; the SOP task (Section 4.2), which predicts synthesis operations given the target material and ground-truth precursors; and the complete MSP task (Section 4.3), which predicts both precursors and synthesis operations simultaneously from the target material alone.

**Datasets.** We utilize an inorganic material synthesis dataset that extracts synthesis recipes from materials science papers Kononova et al. (2019), which has been used in prior studies focusing on PP Noh et al. (2024); Kim et al. (2024). To address the complete MSP task, we curate a subset of the data that contains full target, precursor, and synthesis operation information, resulting in 10,851 real-world synthesis records. For PP, we restrict the precursors to those with a frequency of at least five, yielding a set of 288 precursors. For synthesis operations, the original scheme consists of five categories, namely *mixing*, *quenching*, *heating*, *drying*, and *shaping*. In the SOP setting, we refine this scheme by decomposing *heating* into *sintering*, *annealing*, and a general *heating* category, yielding a total of seven operation types that better reflect practical synthesis scenarios. Additional details about the dataset are provided in Appendix C.

**Implementation Details.** We construct MSP-LLM by fine-tuning LLMs to perform both PP and SOP. We leverage open-source models LLaMA-3.1-8B Grattafiori et al. (2024) and Qwen 2.5-7B Qwen et al. (2025), and apply QLoRA Dettmers et al. (2023) to enable efficient fine-tuning with limited computational resources. As detailed in Section 3.4, we fine-tune and evaluate the LLMs independently for each task. For the MSP task, these separately fine-tuned models are deployed simultaneously to generate the final output. Additional implementation details are provided in Appendix B.

**Evaluation Protocol.** We consider two dataset split strategies with an 8:1:1 train/validation/test ratio for experiments. Each sample is associated with a unique precursor set and synthesis operation sequence. The first is a random split, in which target formulas may appear across different splits, whereas each occurrence corresponds to a different precursor set and synthesis operation sequence due to the one-to-many mapping. The second is a target-disjoint split, where target formulas are strictly separated across different splits, enabling evaluation on previously unseen target materials. Additional details are provided in Appendix C.

## 4.1 PRECURSOR PREDICTION (PP) TASK

**Baseline Methods.** We prepare two types of baselines: (1) Specialized models for PP: we consider retrieval-based approaches that leverage reference material information through retrieval mechanisms, including the method of He et al. (2023), which uses a composition-vector–based retrieval scheme, and Retrieval-Retro Noh et al. (2024), a state-of-the-art model that combines advanced retrieval mechanisms with attention. (2) LLM baselines: we follow the prompt design of Prein et al. (2025b) to evaluate proprietary LLMs on PP, and conduct experiments with GPT-4o-mini, GPT-4o, GPT-5.1, and Claude Haiku 4.5. Here, we report the performance of MSP-LLM using an LLM that is independently trained specifically for the PP task.

**Evaluation Protocol.** Following prior work Prein et al. (2025b); Noh et al. (2024), we adopt the Top-$K$ Exact Match metric to evaluate precursor prediction performance.

**Empirical Results.** In Table 1, we observe that both specialized models and API-based LLMs exhibit limited capability in precursor prediction. Specifically, while the retrieval-based specialized model (Retrieval-Retro) achieves performance comparable to GPT-4o, it relies on an external retrieval system and lacks flexibility in adapting the size of the predicted precursor set, which constrains its overall accuracy. Moreover, for API-based LLMs, while increasing model scale and the number of in-context examples improves the performance of API-based LLMs, a substantial performance gap remains. Even the most powerful model, GPT-5.1, falls short of our method by approximately 6–7 percentage points in Top-1 accuracy and around 10 percentage points in Top-10 accuracy across both dataset splits. In contrast, our framework, regardless of the underlying LLM backbone, consistently outperforms all baselines across all Top-$K$ levels. These results demonstrate that structured, task-specific fine-tuning and domain-aware inductive biases enable more effective learning of precursor selection constraints than either prompt-based conditioning or retrieval-dependent pipelines.

Table 1: PP performance on two dataset splits.

| Method | Split 1 | | | | Split 2 | | | |
|---|---|---|---|---|---|---|---|---|
| | Top-1 | Top-3 | Top-5 | Top-10 | Top-1 | Top-3 | Top-5 | Top-10 |
| *Specialized Models* | | | | | | | | |
| He et al. (2023) | 53.11 | 60.05 | 61.06 | 63.07 | 54.01 | 61.10 | 62.69 | 65.02 |
| Retrieval-Retro Noh et al. (2024) | 60.88 | 69.38 | 70.93 | 74.13 | 66.32 | 74.16 | 75.84 | 78.54 |
| *Large Language Models (API)* | | | | | | | | |
| GPT-4o-mini (20-shot) | 56.58 | 65.36 | 67.82 | 70.66 | 54.85 | 63.40 | 67.91 | 70.34 |
| GPT-4o-mini (40-shot) | 55.21 | 63.44 | 66.27 | 69.10 | 55.97 | 64.27 | 67.82 | 71.36 |
| GPT-4o (20-shot) | 60.05 | 68.56 | 71.76 | 75.05 | 60.17 | 69.31 | 73.04 | 76.31 |
| GPT-4o (40-shot) | 61.79 | 70.48 | 73.40 | 76.32 | 61.57 | 71.27 | 73.97 | 77.05 |
| GPT-5.1 (20-shot) | 64.72 | 77.42 | 80.53 | 83.27 | 67.07 | 77.61 | 82.37 | 85.17 |
| GPT-5.1 (40-shot) | 64.08 | 76.33 | 79.62 | 83.46 | 66.60 | 77.89 | 82.74 | 86.01 |
| Claude Haiku 4.5 (20-shot) | 55.03 | 68.37 | 74.50 | 78.88 | 56.44 | 68.84 | 73.13 | 78.92 |
| Claude Haiku 4.5 (40-shot) | 57.31 | 68.74 | 74.31 | 78.43 | 56.53 | 71.64 | 76.12 | 81.81 |
| **MSP-LLM (Qwen-2.5-7B)** | **71.02** | 85.37 | 90.49 | 94.06 | 71.92 | 88.62 | 92.72 | 95.53 |
| **MSP-LLM (LLaMA-3.1-8B)** | 70.75 | **85.56** | **91.68** | **94.97** | **72.85** | **88.99** | **93.56** | **96.36** |

Table 2: SOP performance on two dataset splits.

| Method | Split 1 | | | | Split 2 | | | |
|---|---|---|---|---|---|---|---|---|
| | Top-10 | NED↑ | LCS↑ | F-1↑ | Top-10 | NED↑ | LCS↑ | F-1↑ |
| *Specialized Models* | | | | | | | | |
| Retrieval based | 10.4 | 0.6072 | 0.7194 | 0.7596 | 11.19 | 0.6030 | 0.7198 | 0.7564 |
| SPENDE Na (2023) | N/A | 0.4508 | 0.5824 | 0.6013 | N/A | 0.4522 | 0.5821 | 0.6010 |
| *Large Language Models (API)* | | | | | | | | |
| GPT-4o-mini (20-shot) | 5.85 | 0.5717 | 0.6992 | 0.7355 | 7.09 | 0.5847 | 0.7089 | 0.7407 |
| GPT-4o-mini (40-shot) | 5.94 | 0.5844 | 0.7092 | 0.7454 | 7.93 | 0.5934 | 0.7141 | 0.7442 |
| GPT-4o (20-shot) | 5.03 | 0.5654 | 0.6925 | 0.7219 | 3.82 | 0.5670 | 0.6865 | 0.7166 |
| GPT-4o (40-shot) | 4.34 | 0.5693 | 0.6890 | 0.7195 | 4.94 | 0.5749 | 0.6924 | 0.7245 |
| GPT-5.1 (20-shot) | 2.93 | 0.5334 | 0.6583 | 0.7101 | 2.33 | 0.5301 | 0.6547 | 0.6968 |
| GPT-5.1 (40-shot) | 2.74 | 0.5434 | 0.6660 | 0.7146 | 2.33 | 0.5261 | 0.6498 | 0.6925 |
| Claude Haiku 4.5 (20-shot) | 2.56 | 0.5534 | 0.6727 | 0.7174 | 1.68 | 0.5439 | 0.6656 | 0.7017 |
| Claude Haiku 4.5 (40-shot) | 4.11 | 0.5669 | 0.6801 | 0.7172 | 3.26 | 0.5569 | 0.6730 | 0.7037 |
| **MSP-LLM (Qwen-2.5-7B)** | **26.51** | **0.7207** | **0.8100** | **0.8359** | **32.18** | 0.7420 | **0.8240** | 0.8493 |
| **MSP-LLM (LLaMA-3.1-8B)** | **26.51** | 0.7174 | 0.8075 | 0.8353 | 31.44 | **0.7424** | 0.8235 | **0.8498** |

### 4.2 Synthesis Operation Prediction (SOP) Task

**Baseline Methods.** We evaluate two categories of baselines: To emulate real experimental practices of chemists, who search for materials similar to a target and adapt their reported synthesis procedures, we use the retriever proposed by He et al. (2023) to identify similar materials in the training set and take their associated synthesis operations as predictions. In addition, we evaluate the SPENDE Na (2023) as a specialized baseline. LLM baselines: we design task-specific prompts to evaluate proprietary LLMs on SOP, and conduct experiments with GPT-4o-mini, GPT-4o, GPT-5.1, and Claude Haiku 4.5.

**Evaluation Protocol.** Since SOP is a sequence generation task, we adopt the Top-$K$ Exact Match metric as a strict evaluation criterion that requires both the operation set and the exact ordering of the sequence to match the ground truth. To provide a more fine-grained assessment, we additionally report the Normalized Edit Distance (NED), which measures the similarity between predicted and ground-truth operation sequences, and the Longest Common Subsequence (LCS), which captures the length of the longest ordered subsequence shared with the ground truth. Finally, to evaluate content-level coverage independent of ordering, we use the multiset F-1 score (F-1), which measures the overlap between the predicted and reference operation elements while accounting for duplicates. For these sequence-level metrics, we report the maximum value among the Top-10 candidate predictions. For SPENDE Na (2023), which produces only a single output per input, top-10 metrics are not applicable; instead, we report alternative evaluation metrics computed on the single generated output. We report only the Top-10 results in the main table, while the Top-1, Top-3, and Top-5 metrics are provided in Appendix E.

**Empirical Results.** Table 2 summarizes the results. Overall, LLM-based baselines exhibit limited performance in the SOP setting. We observe no consistent improvement with increasing model size; instead, smaller models such as GPT-4o-mini achieve relatively better performance compared to larger LLMs. This suggests that extensive general knowledge about target materials may act as noisy information when generating structured synthesis operation sequences, in the absence of an appropriate conditioning or decision strategy, rather than providing a clear benefit. SPENDE Na (2023), which is especially designed for material domain, shows low performance on more general materials, indicating that it is not well-suited for SOP prediction in broad chemical spaces. The strongest baseline performance is obtained when retrieval-based methods He et al. (2023) are employed, which more closely resemble the workflow of experimental chemists. Although this approach outperforms other models, it remains heavily dependent on the retrieved synthesis procedures of similar materials and therefore exhibits limited predictive generalization. By contrast, the proposed method consistently and substantially outperforms all baselines across all splits and evaluation metrics. This improvement can be attributed to the material group decision chain, which enables the model to select synthesis strategies tailored to specific material classes, as well as to the use of explicit conditioning that allows precursor information to be effectively leveraged during synthesis operation prediction.

### 4.3 Complete Material Synthesis Planning (MSP) Task

**Baseline Methods.** Since no existing methods directly address the complete MSP task, we construct several baseline pipelines. For LLM-based baselines, we prompt a single LLM to perform the entire MSP task in single process. Additionally, we combine the best-performing LLMs for each subtask into a hybrid pipeline for comparison. Note that MSP-LLM employs LLMs that are independently fine-tuned for the PP and SOP task.

**Evaluation Protocol.** To evaluate complete MSP, both the predicted precursor set and synthesis operation sequence must be correct. For a rigorous assessment, we adopt the Top-$K$ Exact Match

Table 3: MSP Performance on two dataset splits.

| Method | Split 1 | | | | Split 2 | | | |
|---|---|---|---|---|---|---|---|---|
| | Top-1 | Top-3 | Top-5 | Top-10 | Top-1 | Top-3 | Top-5 | Top-10 |
| *Large Language Models (API)* | | | | | | | | |
| GPT-4o-mini (20-shot) | 0.0 | 0.09 | 0.09 | 0.19 | 0.19 | 0.28 | 0.28 | 0.47 |
| GPT-4o-mini (40-shot) | 0.0 | 0.0 | 0.09 | 0.09 | 0.27 | 0.27 | 0.37 | 0.64 |
| GPT-4o (20-shot) | 0.27 | 0.55 | 0.64 | 0.73 | 0.47 | 0.47 | 0.47 | 0.56 |
| GPT-4o (40-shot) | 0.27 | 0.27 | 0.37 | 0.64 | 0.37 | 0.84 | 0.93 | 1.12 |
| GPT-5.1 (20-shot) | 0.46 | 0.64 | 0.73 | 1.01 | 0.65 | 0.75 | 0.75 | 0.75 |
| GPT-5.1 (40-shot) | 0.55 | 0.64 | 0.64 | 0.73 | 0.37 | 0.37 | 0.47 | 0.47 |
| Claude Haiku 4.5 (20-shot) | 0.56 | 0.65 | 0.65 | 0.74 | 0.56 | 0.65 | 0.65 | 0.75 |
| Claude Haiku 4.5 (40-shot) | 0.56 | 0.65 | 0.65 | 0.74 | 0.37 | 0.84 | 0.93 | 1.12 |
| GPT-5.1(PP) + GPT-4o-mini (SOP) | 0.91 | 1.65 | 2.01 | 3.75 | 1.4 | 2.71 | 3.64 | 5.41 |
| **MSP-LLM (Qwen-2.5-7B)** | **5.76** | **10.79** | **13.62** | **18.56** | 6.62 | **13.34** | **16.79** | 22.48 |
| **MSP-LLM (LLaMA-3.1-8B)** | 4.75 | 9.69 | 13.35 | 18.19 | **6.72** | 12.12 | 16.60 | **23.23** |

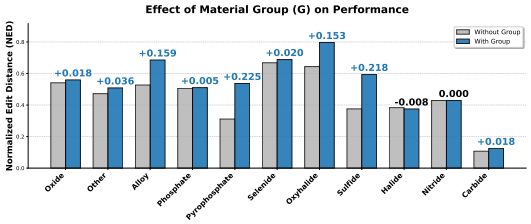


Figure 3: Effect of the material group decision chain on performance (NED) across 11 material classes.

metric, which counts a prediction as correct only when both subtasks, PP and SOP, are simultaneously matched to the ground truth. For both our MSP-LLM and the LLM-based hybrid pipeline, SOP is conditioned on the top-1 precursor prediction during evaluation.

**Empirical Results for MSP.** Table 3 summarizes the results. All LLM-based baselines struggle to solve the complete MSP task. With the exception of the hybrid pipeline (GPT-5 + GPT-4o-mini; each performs the best on PP and SOP, respectively), none of the methods exceeds 2% in terms of Top-10 accuracy. This indicates that predicting both precursors and synthesis operations jointly with a single model is highly challenging, and that a two-stage pipeline can provide some benefit. Nevertheless, even the hybrid pipeline achieves only 3.75% and 5.41% on the two splits, respectively. The complete MSP poses a significant challenge, as it requires correctly predicting both the precursor set and the corresponding synthesis operation sequence. Despite this difficulty, the proposed method attains substantially higher performance, achieving Top-10 scores of 18.56% on Split 1 and 23.23% on Split 2.

### 4.4 Ablation Studies

Table 4: Ablation study on precursor prediction in MSP-LLM (LLaMA-3.1-8B) on Dataset Split 1.

| Model | Group $G$ | Precursor Prediction (PP) | | | |
|---|---|---|---|---|---|
| | | Top-1 | Top-3 | Top-5 | Top-10 |
| (A) Naive Fine-tuning | | 70.59 | 84.90 | 90.62 | 94.39 |
| (B) + Material Group (**MSP-LLM**) | ✓ | **70.75** | **85.56** | **91.68** | **94.97** |

Table 5: Ablation studies on SOP on Dataset Split 1. EC represents explicit conditioning with precursor types.

| Model | Group $G$ | EC | Synthesis Operation Prediction (SOP) | | | |
|---|---|---|---|---|---|---|
| | | | Top-10 | NED↑ | LCS ↑ | F-1 ↑ |
| (A) Naive Fine-tuning | | | 11.24 | 0.5336 | 0.6311 | 0.6572 |
| (B) + Material Group | ✓ | | 13.35 | 0.5623 | 0.6605 | 0.6864 |
| (C) + Explicit Conditioning | | ✓ | 25.13 | 0.7168 | 0.8064 | 0.8342 |
| (D) $G$ + EC w/o precursor types | ✓ | ✓ | 25.50 | 0.7188 | 0.8067 | 0.8353 |
| **(E) MSP-LLM (Qwen-2.5-7B)** | ✓ | ✓ | **26.51** | **0.7207** | **0.8100** | **0.8359** |

To evaluate the contribution of each component in the MSP subtasks, we perform ablation studies in Tables 4 and 5. For both PP and SOP, removing the material group decision chain leads to a noticeable performance degradation, highlighting the importance of structured decision-making for both tasks. For SOP, we further examine the effects of explicit conditioning and the precursor type information. We observe that explicit conditioning and the precursor types plays a critical role in the observed performance gains, underscoring the importance of incorporating precursor constraints in synthesis operation prediction.

Moreover, to examine the effectiveness of the material group decision chain, we analyze how it improves performance across the 11 material classes. As shown in Figure 3, simply introducing the decision chain yields consistent performance gains in the majority of classes (9 out of 11). Notably, although oxides constitute approximately 85% of the dataset, the decision chain provides chemically plausible guidance across different material categories, rather than biasing predictions toward the dominant class.

## 5 Conclusion

This paper proposed MSP-LLM for autonomous MSP across general material spaces. MSP-LLM decomposes complex and unstructured MSP problems into two well-defined subproblems: precursor prediction (PP) and synthesis operation prediction (SOP). In the state of PP, MSP-LLM predicts a plausible precursor set for a target product material and its corresponding material group induced

by chemical principles. In the subsequent SOP stage, MSP-LLM generates a sequence of synthesis operations based on both the target product and the predicted precursor set. MSP-LLM achieved 94.97% top-10 accuracy in PP and an F1-score of 0.8359 in SOP, and it eventually outperformed competitor LLMs in the complete MSP tasks. These experimental results on the real-world materials dataset demonstrate the practical potential of MSP-LLM in autonomous MSP, a long-standing challenge in chemical sciences.

## ETHICS STATEMENT

In this work, we present MSP-LLM, a unified framework for MSP. It introduces an LLM fine-tuning strategy to perform the MSP subtasks, PP and SOP, and then sequentially links PP and SOP to achieve complete MSP. Although MSP-LLM significantly outperforms the baselines, demonstrating its superiority, guidance from domain experts remains essential for successful real-world application and use.

## REPRODUCIBILITY STATEMENT

To support reproducibility, we report all experimental details in the main paper and the appendix, and release the accompanying code at https://github.com/HeewoongNoh/MSP-LLM-Official. The computational resources used are described in Appendix B.

## REFERENCES


Milad Abolhasani and Eugenia Kumacheva. The rise of self-driving labs in chemical and materials sciences. *Nature Synthesis*, 2(6):483–492, 2023.

Adib Bazgir, Yuwen Zhang, et al. Matagent: A human-in-the-loop multi-agent llm framework for accelerating the material science discovery cycle. In *AI for Accelerated Materials Design-ICLR 2025*, 2025.

Andres M Bran, Theo A Neukomm, Daniel P Armstrong, Zlatko Jončev, and Philippe Schwaller. Chemical reasoning in llms unlocks steerable synthesis planning and reaction mechanism elucidation. *arXiv preprint arXiv:2503.08537*, 2025.

So Eun Choi, Miyoung Jang, SoHee Yoon, SangHyun Yoo, Jooyeon Ahn, Minho Kim, Ho-Gyeong Kim, Yebin Jung, Seongeon Park, Young-Seok Kim, et al. Llm-driven synthesis planning for quantum dot materials development. *Journal of Chemical Information and Modeling*, 65(6): 2748–2758, 2025.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. *Advances in neural information processing systems*, 36:10088–10115, 2023.

Qianggang Ding, Santiago Miret, and Bang Liu. Matexpert: Decomposing materials discovery by mimicking human experts. *arXiv preprint arXiv:2410.21317*, 2024.

Yuanqi Du, Yingheng Wang, Yining Huang, Jianan Canal Li, Yanqiao Zhu, Tian Xie, Chenru Duan, John Gregoire, and Carla P Gomes. $M^2$ hub: Unlocking the potential of machine learning for materials discovery. *Advances in Neural Information Processing Systems*, 36:77359–77378, 2023.

Eberhard Engel. *Density functional theory*. Springer, 2011.

Amparo Fuertes. Synthetic approaches in oxynitride chemistry. *Progress in Solid State Chemistry*, 51:63–70, 2018.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

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. *arXiv preprint arXiv:2402.04379*, 2024.

Tanjin He, Haoyan Huo, Christopher J Bartel, Zheren Wang, Kevin Cruse, and Gerbrand Ceder. Precursor recommendation for inorganic synthesis by machine learning materials similarity from scientific literature. *Science advances*, 9(23):eadg8180, 2023.

Riley J Hickman, Malcolm Sim, Sergio Pablo-García, Gary Tom, Ivan Woolhouse, Han Hao, Zeqing Bao, Pauric Bannigan, Christine Allen, Matteo Aldeghi, et al. Atlas: a brain for self-driving laboratories. *Digital Discovery*, 4(4):1006–1029, 2025.

Martin Jansen. A concept for synthesis planning in solid-state chemistry. *Angewandte Chemie International Edition*, 41(20):3746–3766, 2002.

Seongmin Kim, Yousung Jung, and Joshua Schrier. Large language models for inorganic synthesis predictions. *Journal of the American Chemical Society*, 146(29):19654–19659, 2024.

Wan-Tae Kim, Weon-Gyu Lee, Hong-Eun An, Hiroyasu Furukawa, WooSeok Jeong, Sung-Chul Kim, Jeffrey R Long, Sohee Jeong, and Jung-Hoon Lee. Machine learning-assisted design of metal–organic frameworks for hydrogen storage: A high-throughput screening and experimental approach. *Chemical Engineering Journal*, 507:160766, 2025.

Olga Kononova, Haoyan Huo, Tanjin He, Ziqin Rong, Tiago Botari, Wenhao Sun, Vahe Tshitoyan, and Gerbrand Ceder. Text-mined dataset of inorganic materials synthesis recipes. *Scientific data*, 6 (1):203, 2019.

Youjia Li, Vishu Gupta, Muhammed Nur Talha Kilic, Kamal Choudhary, Daniel Wines, Wei-keng Liao, Alok Choudhary, and Ankit Agrawal. Hybrid-llm-gnn: integrating large language models and graph neural networks for enhanced materials property prediction. *Digital Discovery*, 4(2): 376–383, 2025.

Santiago Miret and NM Anoop Krishnan. Enabling large language models for real-world materials discovery. *Nature Machine Intelligence*, 7(7):991–998, 2025.

Gyoung S Na. Artificial intelligence for learning material synthesis processes of thermoelectric materials. *Chemistry of Materials*, 35(19):8272–8280, 2023.

Heewoong Noh, Namkyeong Lee, Gyoung S Na, and Chanyoung Park. Retrieval-retro: Retrieval-based inorganic retrosynthesis with expert knowledge. *Advances in Neural Information Processing Systems*, 37:25375–25400, 2024.

Ryotaro Okabe, Zack West, Abhijatmedhi Chotrattanapituk, Mouyang Cheng, Denisse Córdova Carrizales, Weiwei Xie, Robert J Cava, and Mingda Li. Large language model-guided prediction toward quantum materials synthesis. *arXiv preprint arXiv:2410.20976*, 2024.

Elton Pan, Soonhyoung Kwon, Sulin Liu, Mingrou Xie, Alexander J Hoffman, Yifei Duan, Thorben Prein, Killian Sheriff, Yuriy Roman-Leshkov, Manuel Moliner, et al. Diffsyn: A generative diffusion approach to materials synthesis planning. *arXiv preprint arXiv:2509.17094*, 2025.

Thorben Prein, Elton Pan, Sami Haddouti, Marco Lorenz, Janik Jehkul, Tymoteusz Wilk, Cansu Moran, Menelaos Panagiotis Fotiadis, Artur P Toshev, Elsa Olivetti, et al. Retro-rank-in: A ranking-based approach for inorganic materials synthesis planning. *arXiv preprint arXiv:2502.04289*, 2025a.

Thorben Prein, Elton Pan, Janik Jehkul, Steffen Weinmann, Elsa Olivetti, and Jennifer LM Rupp. Language models enable data-augmented synthesis planning for inorganic materials. *ACS Applied Materials & Interfaces*, 2025b.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2025. URL `https://arxiv.org/abs/2412.15115`.

Chintamani Nagesa Ramachandra Rao and Jagannatha Gopalakrishnan. *New directions in solid state chemistry*. Cambridge University Press, 1997.

Alice Rueda, Mohammed S Hassan, Argyrios Perivolaris, Bazen G Teferra, Reza Samavi, Sirisha Rambhatla, Yuqi Wu, Yanbo Zhang, Bo Cao, Divya Sharma, et al. Understanding llm scientific reasoning through promptings and model's explanation on the answers. *arXiv preprint arXiv:2505.01482*, 2025.

Ulrich S Schubert and Nicola Hüsing. *Synthesis of inorganic materials*. John Wiley & Sons, 2019.

Nathan J Szymanski, Bernardus Rendy, Yuxing Fei, Rishi E Kumar, Tanjin He, David Milsted, Matthew J McDermott, Max Gallant, Ekin Dogus Cubuk, Amil Merchant, et al. An autonomous laboratory for the accelerated synthesis of novel materials. *Nature*, 624(7990):86–91, 2023a.

Nathan J Szymanski, Bernardus Rendy, Yuxing Fei, Rishi E Kumar, Tanjin He, David Milsted, Matthew J McDermott, Max Gallant, Ekin Dogus Cubuk, Amil Merchant, et al. An autonomous laboratory for the accelerated synthesis of novel materials. *Nature*, 624(7990):86–91, 2023b.

Sara Szymkuć, Ewa P Gajewska, Tomasz Klucznik, Karol Molga, Piotr Dittwald, Michał Startek, Michał Bajczyk, and Bartosz A Grzybowski. Computer-assisted synthetic planning: the end of the beginning. *Angewandte Chemie International Edition*, 55(20):5904–5937, 2016.

Xiangru Tang, Qiao Jin, Kunlun Zhu, Tongxin Yuan, Yichi Zhang, Wangchunshu Zhou, Meng Qu, Yilun Zhao, Jian Tang, Zhuosheng Zhang, et al. Risks of ai scientists: prioritizing safeguarding over autonomy. *Nature Communications*, 16(1):8317, 2025.

Hanchen Wang, Tianfan Fu, Yuanqi Du, Wenhao Gao, Kexin Huang, Ziming Liu, Payal Chandak, Shengchao Liu, Peter Van Katwyk, Andreea Deac, et al. Scientific discovery in the age of artificial intelligence. *Nature*, 620(7972):47–60, 2023.

Anthony R West. *Solid state chemistry and its applications*. John Wiley & Sons, 2022.

Tianren Zhang and Dai-Bei Yang. Multimodal machine learning with large language embedding model for polymer property prediction. *Chemistry of Materials*, 37(18):7002–7013, 2025.

# *Supplementary Material*

*- MSP-LLM: A Unified Large Language Model Framework for Complete Material Synthesis Planning -*


**A Related Works** **15**

**B Implementation Details** **15**

B.1 Training Details . . . . . 15

**C Dataset** **16**

C.1 Preprocessing . . . . . 16

C.2 Dataset Split . . . . . 16

**D Baselines** **16**

**E Supplementary Results** **17**

**F Limitations** **17**

**G Prompt Templates** **17**

# A Related Works

**Material Synthesis Planning.** MSP is a long-standing challenge in chemical sciences due to the inherent complexity and uncertainty associated with chemical reactions Szymanski et al. (2023b); Abolhasani & Kumacheva (2023). In materials science, calculation- and simulation-based approaches on physical chemistry, often grounded in density functional theory (DFT) Engel (2011), have been used to screen precursor candidates and to analyze thermodynamically stable synthesis pathways toward target product materials Hickman et al. (2025). However, these conventional methods have faced a fundamental limitation in real-world applications: the atomic structures of product and candidate precursor materials, which are required for reliable first-principles calculations, are typically unavailable in most chemical experiments Prein et al. (2025a); Choi et al. (2025). To overcome this challenge, prior machine learning methods for composition-based MSP have focused on either predicting precursor sets He et al. (2023); Noh et al. (2024) or generating sequences of synthesis operations Na (2023). However, MSP remains unresolved because there is currently no computational method capable of solving the complete MSP task, which requires predicting both plausible precursors and corresponding synthesis pathways.

**LLM for Materials Science.** LLMs have been increasingly adopted in materials science to exploit their broad prior knowledge of physical chemistry, thereby supporting the construction of downstream predictive and generative models Bazgir et al. (2025); Miret & Krishnan (2025). In particular, LLMs have notably accelerated materials discovery by facilitating efficient high-throughput screening based on surrogate models that predict the physical and chemical properties of inorganic materials Li et al. (2025) and polymers Zhang & Yang (2025). With the advances in the reasoning capabilities of LLMs, they have recently emerged as a method for addressing the MSP task, which constitutes a next remaining challenge for materials discovery. In this direction, RHS2LHS and TGT2CEQ were proposed to infer precursor materials and chemical reactions for designing an initial stage of the material synthesis plan Okabe et al. (2024). However, both RHS2LHS and TGT2CEQ require additional information about ground-truth synthesis pathways when predicting precursors and do not address the SOP tasks. LLM-QDMD is an LLM-driven MSP method for designing experiment plans for the synthesis of quantum dot materials Choi et al. (2025). Although LLM-QDMD achieved substantial success within the domain of quantum dot materials, its applicability to real-world applications is inherently limited because it focuses on refining existing synthesis plans for quantum dot materials rather than generating new synthesis plans across general materials spaces.

# B Implementation Details

Implementation details of MSP-LLM are provided in this section.

## B.1 Training Details

All models are fine-tuned using Python 3.10.18, PyTorch 2.3.1, Accelerate 1.10.1, and Hugging Face Hub 0.34.4 on 2 NVIDIA RTX A6000 GPUs. To accommodate resource-limited settings, we employ QLoRA for parameter-efficient fine-tuning. The training hyperparameter configurations for each task and LLM are reported in Table 6

Table 6: Hyperparameter configuration for PP and SOP across different LLM backbones and dataset splits.

| LLM | Task | Split 1 | | | Split 2 | | |
|---|---|---|---|---|---|---|---|
| | | LoRA Rank ($r$) | LoRA Alpha | lr / bs | LoRA Rank ($r$) | LoRA Alpha | lr / bs |
| LLaMA-3.1-8B Grattafiori et al. (2024) | PP | 16 | 32 | 1e-4 / 4 | 16 | 32 | 1e-4 / 2 |
| LLaMA-3.1-8B Grattafiori et al. (2024) | SOP | 16 | 32 | 1e-4 / 2 | 16 | 32 | 1e-4 / 2 |
| Qwen-2.5-7B Qwen et al. (2025) | PP | 16 | 32 | 1e-4 / 4 | 16 | 32 | 1e-4 / 4 |
| Qwen-2.5-7B Qwen et al. (2025) | SOP | 16 | 32 | 1e-4 / 2 | 16 | 32 | 1e-4 / 4 |

# C DATASET

## C.1 PREPROCESSING

We preprocess the real-world experimental inorganic synthesis dataset Kononova et al. (2019).

**Precursor Prediction (PP).** We filtered out samples that do not contain synthesis operation–related information and removed data associated with compositional systems. We further excluded cases in which the corresponding precursor set contained only a single precursor. In addition, we applied a frequency-based filtering over the full dataset, retaining only precursors that appear at least five times. As a result, 288 unique precursors remained in the final dataset.

**Synthesis Operation Prediction (SOP).** In addition to the five basic synthesis operations provided in the original dataset—heating, mixing, shaping, quenching, and drying—we refine the heating category to better reflect real-world experimental practice. Specifically, we introduce two subcategories, sintering and annealing, and classify the remaining cases as generic heating. To distinguish these categories, we leverage synthesis-related subkeywords in the dataset: samples containing subkeywords related to "sinter" are labeled as sintering, while those containing subkeywords related to "anneal" are labeled as annealing.

To capture the synthesis intent and strategy of experimentalists, as well as coarse-grained material information (e.g., thin films), we use the titles of the source literature as contextual cues (synthesis context). We extract the following categories from the titles using GPT-4o: 'host material', 'dopant or substitution', 'material class', 'functional property', 'composition control', and 'processing or stimulus'. When relevant information is not present, the corresponding field is set to None. Since 'processing or stimulus' contains information directly related to synthesis operations, we exclude this field from the model inputs to ensure a precise evaluation. Instead, only the extracted information from 'host material', 'dopant or substitution', 'material class', 'functional property', and 'composition control', is used by all baselines as well as MSP-LLM.

## C.2 DATASET SPLIT

Fundamentally, a single material can be synthesized using multiple precursors and multiple operations. In other words, a single chemical formula can correspond to multiple synthesis routes, which implies a one-to-many relationship. Through data preprocessing, we removed duplicates and kept only unique combinations of chemical formula–precursor–operation.

In Split 1, the dataset was randomly divided into train/validation/test sets with an 8:1:1 ratio, without restricting chemical formulas. As a result, the same chemical formula may appear in the train, validation, and test sets. However, because the precursors and operations that must be predicted in the PP and SOP tasks differ from those seen during training, the model still needs to infer different precursors and operations even for a chemical formula it has already encountered.

In Split 2, the dataset was also divided with an 8:1:1 ratio, but chemical formulas were enforced to be non-overlapping across the train, validation, and test sets. This setup allows evaluation of the model's ability to perform PP and SOP on chemical formulas that were completely unseen during training, thereby measuring its generalization capability with respect to chemical formulas.

# D BASELINES

**LLM Baselines.** We compare LLM baselines including GPT-4o-mini[1] , GPT-4o[2] , GPT-5.1[3] , and Claude Haiku 4.5[4] . All models are accessed via their respective APIs. The GPT models are configured with a temperature of 0.8, while Claude Haiku is set to 0.3.

**Specialized Models for Precursor Prediction (PP).** We compared our approach with a retriever-based method He et al. (2023) that learns the relationship between target materials and precursors through masked precursor completion, as well as a more recent work, Retrieval Retro(Noh et al., 2024), which employs an advanced attention mechanism and incorporates thermodynamic relationships between targets and precursors to identify more feasible precursor candidates. For both methods, we

[1] https://openai.com/index/gpt-4o-mini-advancing-cost-efficient-intelligence/
[2] https://openai.com/index/hello-gpt-4o/
[3] https://openai.com/index/gpt-5-1/
[4] https://www.anthropic.com/claude/haiku

used their official implementations available on GitHub. In addition, when solving PP with the LLM baselines, we directly used the prompt from Prein et al. (2025b).

**Specialized Models for Precursor Prediction (SOP).** Using the retriever from He et al. (2023) that was employed for PP, we leveraged the synthesis routes of materials similar to the target material as the predicted outputs, serving as a baseline that reflects the practices of real synthesis experts. In addition, we adopted SPENDE(Na, 2023) which is specifically designed for thermoelectric material synthesis prediction, using the official implementation provided on GitHub. The prompt used by the LLM baselines for the SOP task is provided in the Appendix 10.

## E Supplementary Results

In this section, we provide additional metrics—Top-1, Top-3, and Top-5—supplementing the existing SOP task evaluation.

Table 7: Supplementary Results: Synthesis operation prediction (SOP) performance on two dataset splits.

| **Method** | **Split 1** | | | | **Split 2** | | | |
|---|---|---|---|---|---|---|---|---|
| | **Top-1** | **Top-3** | **Top-5** | **Top-10** | **Top-1** | **Top-3** | **Top-5** | **Top-10** |
| *Specialized Models* | | | | | | | | |
| Retrieval based | 1.46 | 3.75 | 5.94 | 10.42 | 1.12 | 3.54 | 5.50 | 11.19 |
| SPENDE Na (2023) | 3.66 | N/A | N/A | N/A | 5.71 | N/A | N/A | N/A |
| *Large Language Models (API)* | | | | | | | | |
| GPT-4o-mini (20-shot) | 1.73 | 3.11 | 4.2 | 5.85 | 1.68 | 3.26 | 4.01 | 7.09 |
| GPT-4o-mini (40-shot) | 1.83 | 2.74 | 3.56 | 5.94 | 2.15 | 4.2 | 5.69 | 7.93 |
| GPT-4o (20-shot) | 0.09 | 1.09 | 2.56 | 5.03 | 0.47 | 1.03 | 1.77 | 3.82 |
| GPT-4o (40-shot) | 0.18 | 1.01 | 1.92 | 4.34 | 0.28 | 1.03 | 2.05 | 4.94 |
| GPT-5.1 (20-shot) | 0.73 | 1.55 | 1.65 | 2.93 | 0.75 | 1.31 | 1.4 | 2.33 |
| GPT-5.1 (40-shot) | 0.91 | 1.74 | 1.74 | 2.74 | 075 | 1.03 | 1.77 | 2.33 |
| Claude Haiku 4.5 (20-shot) | 0.18 | 0.73 | 1.56 | 2.56 | 0.47 | 0.84 | 1.21 | 1.68 |
| Claude Haiku 4.5 (40-shot) | 0.91 | 2.38 | 3.29 | 4.11 | 0.47 | 1.49 | 2.52 | 3.26 |
| **MSP-LLM (Qwen-2.5-7B)** | **7.86** | **15.36** | **19.65** | **26.51** | **8.77** | **18.47** | 22.57 | **32.18** |
| **MSP-LLM (LlaMA-3.1-8B)** | 6.40 | 13.71 | 19.20 | **26.51** | 8.12 | 17.35 | **22.58** | 31.44 |

## F Limitations

MSP-LLMsuccessfully solves both PP and SOP and connects them to address the complete MSP task. However, in the current setup, only the Top-1 PP prediction is used as input, which means that errors in PP can propagate to the SOP stage. A strategy is needed to enable the use of multiple precursor candidates predicted by PP and to incorporate them into the SOP stage so that the model can select better precursor conditions and improve overall performance.

## G Prompt Templates

In this section, we present the prompt templates used to fine-tune LLMs for PP 3.2 and SOP 3.3. Additionally, we provide the prompts used for the LLM baseline for the SOP task.

Table 8: Prompt for Fine-tuning LLMs for Precursor Prediction (PP).

**System Prompt:** You are an expert solid-state chemist planning a material synthesis.

**Prompt:** Task:
Given the target compound {target_formula}, first classify its material group, and you must select the precursor needed to synthesize the target.

Typically, two or more precursors are required.

The possible material groups are:
`['oxide', 'alloy', 'phosphate', 'pyrophosphate', 'halide', 'oxyhalide', 'carbide', 'nitride', 'selenide', 'sulfide', 'other'].`

Output format:
`[ ['material group'], ['precursor1', 'precursor2', ..., 'precursorN'] ]`

Return only the Python list of single-quoted strings with no extra text or explanation.

Table 9: Prompt for Fine-tuning LLMs for Synthesis Operation Prediction (SOP).

**System Prompt:** You are an expert solid-state chemist planning a material synthesis.

**Prompt:**
[PRECURSOR_TYPES]
types_present: {precursor types}
[/PRECURSOR_TYPES]

Task:
Given the target compound {target_formula}, first classify its material group, then based on the selected precursors {precursors}, propose one plausible synthesis operation sequence using only the following operations:
`['heating', 'sintering', 'annealing', 'mixing', 'shaping', 'quenching', 'drying'].`

Synthesis context: {keyword}

Typically, two or more precursors are required.

The possible material groups are:
`['oxide', 'alloy', 'phosphate', 'pyrophosphate', 'halide', 'oxyhalide', 'carbide', 'nitride', 'selenide', 'sulfide', 'other'].`

Output format:
`[ ['material group'], ['precursor_type1', ..., 'precursor_typeK'], ['precursor1', ..., 'precursorN'], ['op1', ..., 'opM'] ]`

Return only the Python list of single-quoted strings with no extra text or explanation.

Table 10: Prompt for LLM Baselines in Synthesis Operation Prediction (SOP).

**System Prompt:** You are an expert solid-state chemist specializing in synthesis operations.

**Prompt:** Given:
- Target material: {target_formula}
- Precursors for the target material: {precursors}

Task:
Propose a plausible synthesis operation sequence for preparing the target from the given precursors.

Instructions:
1) Choose operations only from the following allowed set: `{"heating", "sintering", "annealing", "mixing", "shaping", "quenching", "drying"}`.
2) "Heating" refers to general heating processes (e.g., calcination, firing) and explicitly excludes Sintering and Annealing.
3) Output only the operation names — do not include temperatures, times, atmospheres, parameters, or explanations.
4) Format the answer as a Python list of lists of strings **exactly**.
- Each inner list is a single synthesis operation sequence, e.g., `["mixing", "heating", "sintering"]`.
- Return **exactly {num_combinations} sequences**.
5) Each sequence must contain **one or more** operations; repetitions are allowed when chemically reasonable.
6) Do not invent new operation names, abbreviations, or comments. Do not output anything other than the list of lists.

# Reference for operation semantics (do NOT include this section in the output):
# - Heating: any general heating process (e.g., calcination, firing) except sintering and annealing.
# - Sintering: high-temperature treatment to densify materials.
# - Annealing: controlled heating/cooling for structural ordering or stress relief.

Synthesis context: {keyword}

**Examples of synthesis operation sequences from target and its precursors:**
{examples}

Output format (example shape only; return exactly {num_combinations} sequences):
```
[
["mixing", "heating", "sintering"],
["mixing", "drying", "heating", "annealing"],
...
]
```
Return only the data as a plain text Python list. Do not use Markdown code fences or language tags, and do not print escaped characters like \n. Start with [ and end with ]. Use single quotes for strings. No extra text.