Active Knowledge Structuring for Large Language Models in Materials Science Text Mining

Xin Zhang[1] **Jingling Yuan**[1*] **Peiliang Zhang**[1] **Jia Liu**[2] **Lin Li**[1]
[1]Hubei Key Laboratory of Transport Internet of Things, School of Computer Science and Artificial Intelligence, Wuhan University of Technology, China
[2]Hubei Key Laboratory of Big Data in Science and Technology, National Science Library (Wuhan), Chinese Academy of Sciences, China
{xinz, yjl, cathylilin}@whut.edu.cn, liuj@mail.whlib.ac.cn

Abstract

Large Language Models (LLMs) offer a promising alternative to traditional Materials Science Text Mining (MSTM) by reducing the need for extensive data labeling and fine-tuning. However, existing zero-/few-shot methods still face limitations in aligning with personalized needs in scientific discovery. To address this, we propose ClassMATe, an active knowledge structuring approach for MSTM. Specifically, we first propose a class definition stylization method to structure knowledge, enabling explicit clustering of latent material knowledge in LLMs for enhanced inference. To align with the scientists' needs, we propose an active needs refining strategy that iteratively clarifies needs by learning from uncertainty-aware hard samples of LLMs, further refining the knowledge structuring. Extensive experiments on seven tasks and eight datasets show that ClassMATe, as a plug-and-play method, achieves performance comparable to supervised learning without requiring fine-tuning or extra knowledge base, highlighting the potential to bridge the gap between LLMs' latent knowledge and real-world scientific applications.[1]

1 Introduction

Materials science texts, including research papers, patents, and reports, contain a large amount of valuable data, such as experimental procedures, material performance metrics, and other critical insights. This drives materials scientists to systematically develop specialized datasets from them to predict material properties and facilitate the discovery of new materials. (Tshitoyan et al., 2019; Suvarna et al., 2023; Zhang et al., 2024; Sun et al., 2024). To streamline this process, Materials Science Text Mining (MSTM) was developed to extract specific elements from these texts that are more challenging than general texts due to intricate property descriptions and condition-dependent terms, including tasks like sentence classification and synthesis action retrieval (Gupta et al., 2022; Song et al., 2023a).

Recent advancements in Large Language Models (LLMs) have made zero-shot or few-shot learning increasingly feasible, leading to the development of Zero/Few-Shot MSTM (Song et al., 2023b; Zhong et al., 2024), which significantly improves usability by eliminating the need for task-specific model training or fine-tuning (hereafter, Zero/Few-Shot MSTM will be called MSTM for simplicity). This paradigm shift empowers materials scientists, especially those aiming to employ MSTM as a data extraction tool, to focus on research needs, freeing them from model adaptation and lowering barriers to material discovery.

Current zero-shot or few-shot approaches (Song et al., 2023b; Dagdelen et al., 2024) focus mainly on fine-tuning LLMs with material knowledge datasets to improve domain-specific performance. However, our analysis reveals that a major bottleneck in applying LLM to MSTM tasks is **not the intrinsic understanding of materials science by LLMs but the ambiguity of task needs.** Specifically, materials scientists often have personalized information needs due to their different focuses in material discovery research, making it challenging for LLMs to grasp these needs accurately. As illustrated in Figure 1, based on real-world observations, the diverse contexts associated with ''materials'' make it difficult to establish a precise and comprehensive definition from the beginning, as it often needs to be

* Corresponding author.

[1]The code is available at https://github.com/xinzcode/ClassMATe.

Transactions of the Association for Computational Linguistics, vol. 13, pp. 1186–1203, 2025. https://doi.org/10.1162/tacl.a.36
Action Editor: Mauro Cettolo. Submission batch: 11/2024; Revision batch: 3/2025, 4/2025; Published 9/2025.

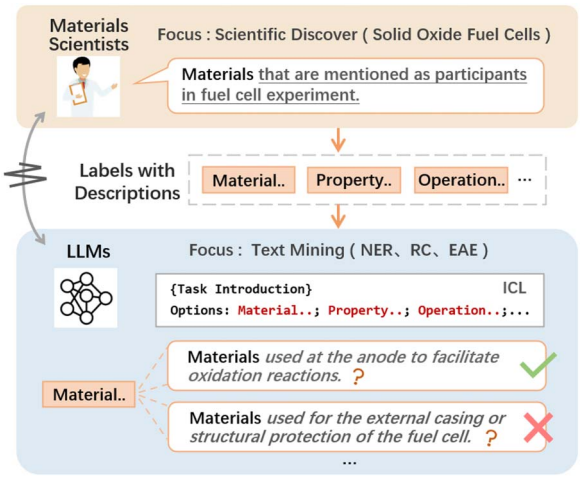


Figure 1: Main challenges in current MSTM approaches: The gap between LLMs' material knowledge and scientists' personalized needs.

determined based on specific texts, especially when faced with large volumes of materials science texts. Even with constraints on the materials (e.g., ''`materials referred to as participants in fuel cell experiments`''), LLMs still struggle to accurately distinguish between desired data (e.g., ''`anode materials`'') and undesired data (e.g., ''`shell materials`''), despite having a sufficient understanding of these concepts. Moreover, leveraging the material knowledge embedded in rapidly evolving LLMs will likely be more effective than substantial investments in domain-specific fine-tuning (Xia et al., 2024).

Thus, beyond domain enhancement through fine-tuning, addressing needs disambiguation while effectively leveraging internal material knowledge could be more crucial. Based on this observation, we propose the concept of **Active Knowledge Structuring (AKS)** for LLMs in MSTM, which aims to enable the LLM to adaptively organize and prioritize relevant material knowledge for inference and actively refine its understanding according to the specific needs of each MSTM task, thereby improving performance without relying solely on large-scale expert-labeled data.

To further the AKS approach, two key questions remain to be explored: **(1) How to effectively structure LLMs' knowledge for MSTM?** Inspired by code-style in-context learning (Li et al., 2024; Sainz et al., 2023), we propose constructing task labels as code class definitions to effectively structure knowledge, as task labels play a critical role in representing scientists' needs. Unlike existing methods that rely on external knowledge bases, we suggest using LLMs to generate relevant material concepts as the attributes of class, effectively aggregating their internal associative knowledge for inference. **(2) How to naturally help identify scientists' needs in MSTM?** To accurately align LLMs with scientists, additional supervisory information is inevitable. We propose an active needs refining strategy aimed at clarifying needs through multiple rounds of few-shot annotation and learning. This enables LLMs to gradually refine their understanding of scientists' needs. Rather than requiring scientists to pre-list all possible scenarios, allowing them to make judgments when encountering uncertain real-world examples offers a more practical approach.

In this paper, we propose ClassMATe, an active knowledge structuring approach for MSTM, designed to align LLMs' material knowledge with scientists' personalized needs. ClassMATe first introduces a class definition stylization method to explicitly structure LLMS' internal materials knowledge. Next, an uncertainty-guided active needs refining strategy is designed. Iterative few-shot learning empowers LLMs to actively clarify personalized needs and refine class definitions. Finally, the detailed needs and key material knowledge are accurately structured for in-context learning, effectively enhancing the inference of LLMs for MSTM. To verify the efficacy of ClassMATe, we further updated the benchmark dataset (Song et al., 2023a) to suit the era of LLMs better and evaluated the performance under zero-shot and few-shot conditions. In summary, the paper makes the following contributions:

- We are the first work that proposes Active Knowledge Structuring, a task-agnostic in-context learning (ICL) paradigm for Zero/Few-Shot MSTM, providing a sustainable pathway with practical applications for Materials Scientists.
- We propose ClassMATe, an AKS-based MSTM approach, effectively bridging the gap between the LLMs' internal materials knowledge and materials scientists' personalized needs.
- We conduct extensive experiments on eight datasets and seven tasks, showing that

ClassMATe, employing 8B LLMs, achieves performance comparable to supervised learning without requiring fine-tuning or external knowledge bases.

2 Related Work

Materials Science Text Mining MSTM primarily focuses on text mining in materials science, involving various NLP tasks such as named entity recognition, relation extraction, and sentence classification. Current MSTM approaches, based on benchmark models and task specificity, can be categorized as follows: **(1) Rule-based**: Early approaches based on rule-based systems and manual ontologies (Tshitoyan et al., 2019; Kononova et al., 2019), heavily relying on scientists for design. **(2) BERT-based**: To overcome the above limitations, supervised learning approaches were developed, adopting BERT as the backbone for task training, including MatBERT (Walker et al., 2021), MatSciBERT (Gupta et al., 2022), BatteryBERT (Huang and Cole, 2022), MatSci-NLP (Song et al., 2023a), and MELT (Kim et al., 2024), still relying on annotated datasets. **(3) Task-Specific LLM Adaptation**: Some recent studies leverage LLMs to enhance MSTM via task-specific fine-tuning or in-context learning (Dagdelen et al., 2024; Zhong et al., 2024), designing prompts tailored to specific tasks. **(4) Task-Agnostic LLM Adaptation**: To reduce reliance on task-specific supervision, researchers have explored zero/few-shot learning by adapting LLMs using general materials corpora (Song et al., 2023b). This direction is the focus of our study, as it facilitates task generalization without costly annotations. However, current efforts mainly enhance material understanding while overlooking the personalized needs of scientists, which we believe deserve greater attention.

In-Context Learning with LLMs ICL has emerged as a powerful paradigm that enables LLMs to perform tasks without fine-tuning by leveraging context constructed from task descriptions and few-shot examples. The following categories are especially noteworthy: **(1) Vanilla ICL**: Early studies demonstrate that LLMs, when given natural language prompts along with a few examples, can effectively perform tasks such as text classification without parameter updates (Min et al., 2022; Wang et al., 2023a; Liu et al., 2023; Li et al., 2022). **(2) Multi-Step Reasoning**: Building on this, chain-style methods such as CoT (Wei et al., 2022), ToT (Yao et al., 2024), and GoT (Besta et al., 2024) aim to decompose tasks into a multi-step format, making them more suitable for tasks that require complex logical reasoning. **(3) Code-Style**: Meanwhile, there is increasing interest in integrating code-style into task inference (Li etal., 2024; Sainz et al., 2023; Wang et al., 2023b), where tasks are represented with code to leverage LLMs' coding abilities and introduce knowledge, making them suitable for knowledge-aware tasks like MSTM. However, task-specific fine-tuning and knowledge bases are still needed. Unlike these approaches, to the best of our knowledge, we are the first to enhance MSTM by fully leveraging the LLMs' internal material knowledge, offering a plug-and-play solution that adapts to their rapid development.

3 Method

In this section, we present ClassMATe, an active knowledge structuring approach for MSTM. The architecture is shown in Figure 2, comprising three main components: ClassDefinition-based Knowledge Structuring, Code-Style Task Inference, and Uncertainty-Aware Active Needs Refining. Following the Problem Formulation in Section 3.1, Sections 3.2– 3.4 will provide detailed explanations of each component.

3.1 Problem Formulation

Following (Song et al., 2023a,b; Kim et al., 2024), materials text mining spans a wide range of downstream tasks in materials science, including both classification and extraction, a simple text entity classification task is introduced to provide a unified understanding, as both classification and extraction tasks can essentially be viewed as text labeling tasks. Given an entity e mentioned within a contextual text T and a set of labels $D = \{d_1, d_2, \ldots, d_m\}$, where each label d_i is associated with a corresponding description $desc(d_i)$, the goal is to determine which label in D best matches e. In some tasks, such as sentence or paragraph classification, the e refers to the contextual text T itself, which should be noted.

3.2 ClassDefinition-based Knowledge Structuring

For many materials scientists with limited fine-tuning experience and facing challenges in

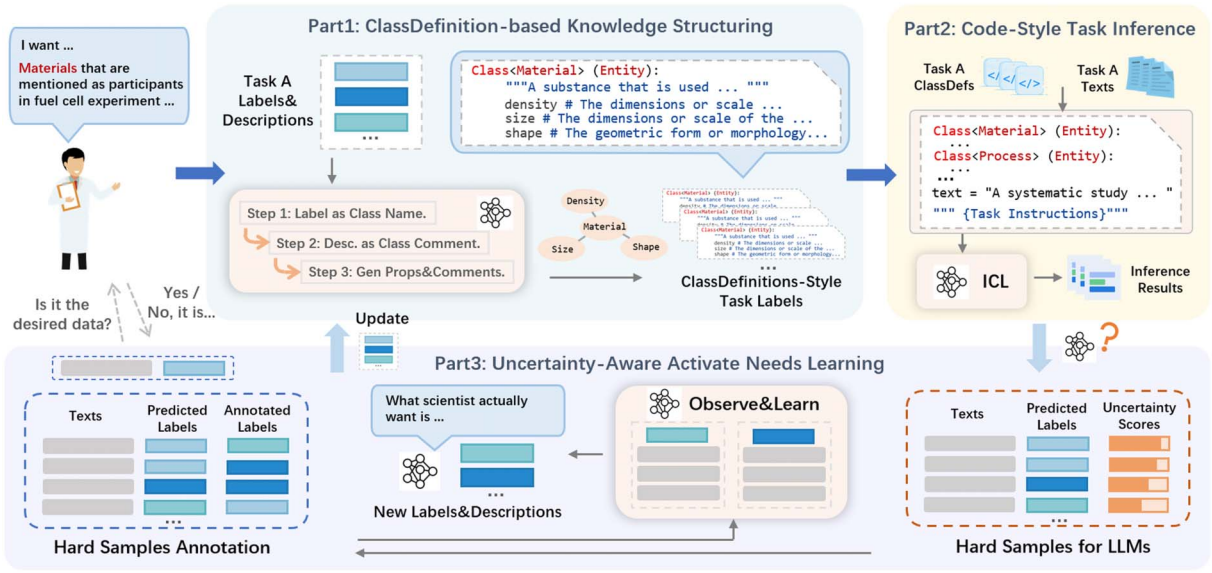


Figure 2: The framework of our proposed ClassMATe, an active knowledge structuring approach for Zero/Few-Shot MSTM, aims to align LLMs with scientists' needs and fully leverage LLMs' knowledge for inference.

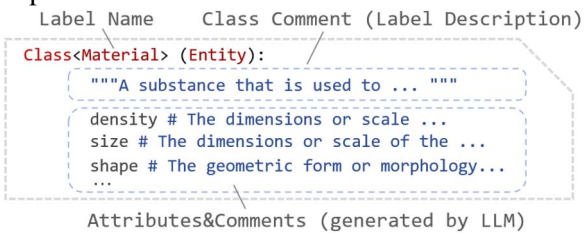
acquiring domain-specific knowledge bases in this multidisciplinary field, leveraging the rich materials science knowledge embedded in rapidly advancing LLMs offers significant potential for knowledge-sensitive MSTM tasks.

Based on this observation, we propose converting task labels into class definitions, thus fully leveraging relevant materials science concepts related to the labels as their attributes to enhance inference. Current approaches mainly rely on combining templates with structured knowledge from external knowledge bases (Sainz et al., 2023; Li et al., 2024), which is unsuitable for MSTM. This paper suggests addressing this challenge by adopting LLMs to construct the class definitions:

Class Name. Given a set of labels D, each label d serves as the class name, representing the desired entity type to be recognized, such as ''*Materials*'', ''*Property*'', etc. In sentence classification tasks, it refers to the desired sentence type, such as ''*GlassScienceText*''.

Class Comment. To further enrich the meaning of each label d, an initial description $c^d \in C^d$ is provided and set as the class comment, such as ''*A material is a substance, element, or class of materials that is mentioned as a participant in a fuel cell experiment*'', as a concrete embodiment of the needs of scientist.

Attributes&Comments. After that, class name d and class comment $c^d \in C^d$ are set as the anchor to generate a set of the most relevant attributes and corresponding comments using the LLMs: $(P, C^p) = \text{Generator}(d, c^d)$. To guide the generation, a concise prompt is designed (simplified version): ''*Given the definition of ''{label}'': {description}, list the {n} most common attributes in the format: [attribute name: definition]*''. Thereby, these generated attributes, such as ''*size*'', ''*shape*'', ''*density*'', etc., can be regarded as the relevant material knowledge of the label within LLMs. To provide a clearer illustration, a simplified class definition diagram is presented below:



Finally, the class definition γ is constructed using the pre-designed task templates (Li et al., 2024): $\gamma = \text{ClassTemplate}(d, c^d, (P, C^p))$. In this way, the scientists' needs and the most relevant material knowledge are effectively structured through the class definition to support subsequent task inference.

3.3 Code-Style Task Inference

After obtaining the class definitions of task labels, the focus shifts to effectively using these class definitions for LLM inference.

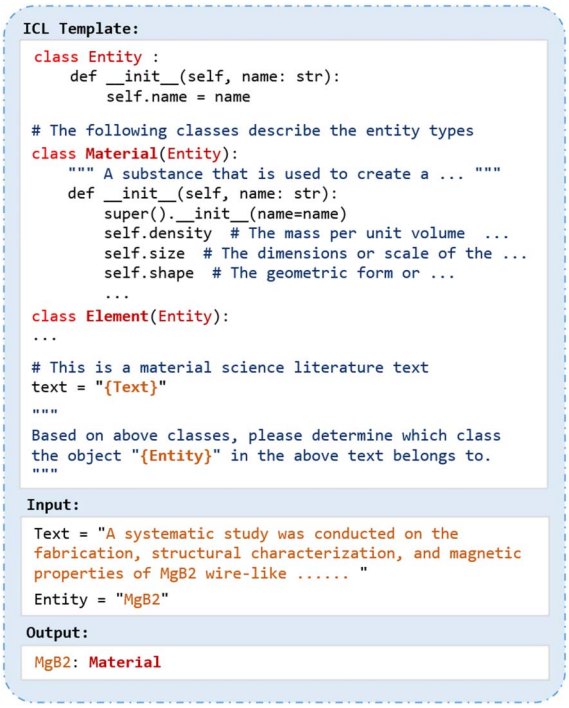


Figure 3: By transforming the task into code form, code-style in-context learning takes full advantage of LLMs' powerful code understanding capabilities for task inference.

To achieve zero-shot learning, code-style in-context learning is introduced to leverage the object-oriented class identification in the code understanding capability of LLMs (Yang et al., 2024). As shown in Figure 3, MSTM tasks are represented in Python programming language, where the object e serves as an instance of a class. Task labels $\{\gamma_1, \gamma_2, \ldots, \gamma_m\}$ are expressed as the corresponding class definitions. Both are integrated into a code-style prompt using task templates. Taking the NER task as an example, task labels are defined as subclasses (e.g., ''`Entity`''), with the background text assigned to the variable ''*Text*'' along with comments. The task goal is defined as instance class-type judgment through annotation, thereby leveraging the instantiation capabilities of LLMs to achieve zero-shot learning. The code-style templates used for the various MSTM tasks are built on the framework in Li et al. (2024). Since task labels can be uniformly represented as class definitions, the templates can be easily adapted to different tasks with minimal modifications, enabling efficient task adaptation.

3.4 Uncertainty-Aware Active Needs Refining

Although class definition stylization effectively achieves the knowledge structuring to enhance inference, the gap remains between LLMs and the personalized needs of scientists. To address this, we propose an uncertainty-aware active needs refining strategy that enables LLMs to clarify these needs through the following steps progressively:

Step 1: Uncertainty-Aware Hard Sample Selection. We first evaluate the uncertainty scores of the LLM inference results and rank the processed samples accordingly, which is derived from the log probabilities[2] of each token provided by the LLMs, where lower log probabilities indicate the higher uncertainty of LLMs. The highest-uncertainty samples are then selected as uncertainty-aware hard samples, reflecting the parts where the needs are most ambiguous.

Step 2: Scientist-Guided Annotation. After selection, the hard samples are then presented to materials scientists for review, where they only need to determine which data align with their personalized scientific discovery needs. These annotated samples are then categorized according to the assigned labels, helping to identify uncertain labels and descriptions highlighting LLMs' key limitations in MSTM.

Step 3: Needs Understanding. These uncertain labels and their corresponding samples are then fed back to the LLM for observation and learning, with a predefined prompt template guiding the generation of more accurate label descriptions. In contrast to conventional ICL, which augments input by directly placing labeled samples into context, our approach focuses on refining the label descriptions. This strategy offers broader coverage of potential cases without being constrained by context length limitations. An example of a prompt is shown in Figure 4. By learning from these samples, LLM can effectively refines the label description with scientists' personalized needs.

Step 4: Needs Evaluation and Update. With the clarified needs, the labels and descriptions are updated, followed by knowledge structuring and task inference. Inevitably, due to potential biases in the samples, not every learning iteration

[2] `https://cookbook.openai.com/examples/using_logprobs`.

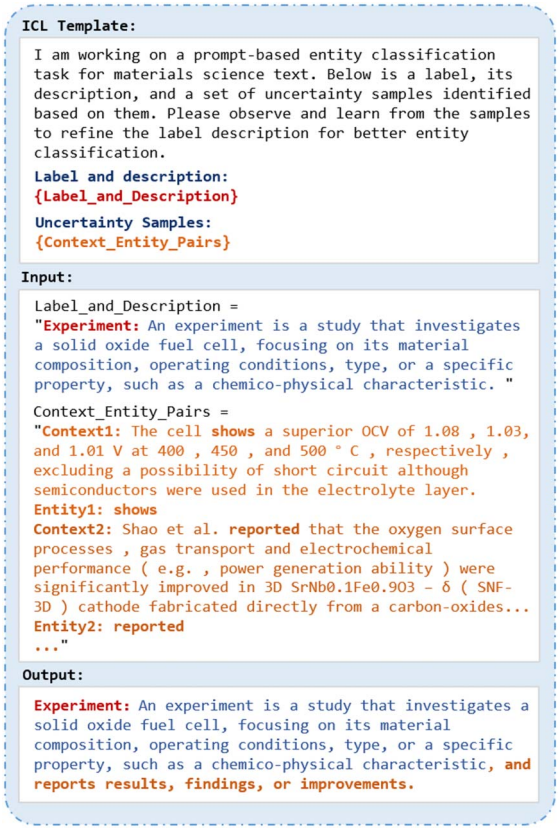


Figure 4: The LLM presents high-uncertainty samples to scientists for judgment and learns from these hard cases to refine label descriptions and better align with needs.

is perfectly accurate. After each update, the new inference results are evaluated for uncertainty. Effective refining is achieved when the average uncertainty decreases, which then serves as the foundation for further needs learning or task inference. Additionally, materials experts can judge whether the updated labels and descriptions meet their intended needs. If the uncertainty does not decrease or the updated results fail to meet expectations, more hard samples are added for the learning to achieve effective updates.

The detailed process of the strategy is outlined in Algorithm 1. Through multiple iterations, the LLM progressively learns from the uncertain-aware hard samples and clarifies the scientists' personalized needs, which in turn guides more focused knowledge structuring and enhances task inference, thereby effectively narrowing the gap.

4 Experimental Setup

4.1 Dataset

To evaluate the effectiveness of the ClassMATe, following the existing approaches of Song et al.

Algorithm 1 Uncertainty-Aware Active Needs Refining Strategy

Input: classDefinitions C, taskTexts T, iterations n
Output: refinedClassDefinitions C^r

while $n > 0$ **do**
  results R, correspondingUncertainties U = Inference (T, C)
  hardSamples $(T, R)^h$ = TopN(SortByUncertainties $((T, R), U)$)
  annotatedHardSamples $((T, R)^h, A)$ = Scientist-Judgment $((T, R)^h)$
  hardLabel&Samples $(a, (T, R)^a), a \in D$ = CategorizeByTrueLabels $(((T, R)^h, A))$
  newLabels&Descriptions D' = Observe&LearnBy LLM$((a, (T, R)^a), a \in D)$
  newClassDefinitions C' = Generator (D')
  newUncertainties U' = Inference (T, C')
  if Average(U') < Average(U) **then**
    $C \leftarrow C'$
  else
    back to Step 3 and add hardSamples
  end if
  $n \leftarrow n - 1$
end while
return C // refinedClassDefinitions C^r

Dataset	#Tasks	#Labels	Duplication(%)
MatScholar	1	7	0.286
SOFC-Token	2	22	0.367
SynthProcs	3	37	0.378
SC-CoMIcs	3	14	0.286
Glasses	1	1	–
MatSciRE	1	5	0.400
SynthActions	1	8	–
SOFC-Sent	1	1	–
Total	7	95	0.316

Table 1: The statistics of benchmark datasets and the proportion of duplicate labels among all labels.

(2023a,b) and Kim et al. (2024), experiments are conducted on a public MSTM benchmark (Song et al., 2023a) consisting of eight datasets and seven tasks, where the texts are all from the materials science literature. The tasks include Named Entity Recognition (NER), Relation Extraction (RE), Event Attribute Extraction (EAE), Paragraph Classification (PC), Synthesis Action Retrieval (SAR), Sentence Classification SC), and Slot Filling (SF). These MSTM tasks differ from typical NLP tasks in that they are primarily based on domain-specific materials science texts, with task labels reflecting precise research needs that are unique to the field of materials science. Detailed statistics are provided in Table 1, and further

Dataset	Focal Point	Personalized Needs of Label ‘‘Material’’
MatScholar	Materials Science	‘‘Any inorganic solid or alloy, any non-gaseous element.’’
SOFC	Solid Oxide Fuel Cells	‘‘A material is a substance, element, or class of materials that is mentioned as a participant in a fuel cell experiment.’’
SynthProcs	Synthesis Procedures	‘‘Materials that are used in the synthesis of the target.’’
SC-CoMIsc	Superconductivity	‘‘The terms of structural entities and sample descriptors such as tetragonal crystal symmetry, bulk/film sample, and grain boundary.’’

Table 2: The personalized needs of the same label across different research contexts in benchmark datasets.

information about the tasks and datasets can be found in Song et al. (2023a).

4.2 Label Description Supplementation

In MSTM tasks, a clear description, especially of the labels, is essential for LLMs to effectively understand the task. In previous supervised learning approaches (Gupta et al., 2022; Song et al., 2023a), this was not considered a problem, as task labels (such as ‘‘*material*’’, ‘‘*brand*’’, ‘‘*CMT*’’, ‘‘*coref_of*, etc.’’) were set by the corresponding scientists and used for one-hot style data annotation. Thus, the large volume of annotated training data inherently contained the scientists' personalized needs. However, when applied in prompt-based LLMs, these **oversimplified task label names fail to convey such specific needs effectively.**

Furthermore, we evaluated the duplicate rate of task labels in the benchmark datasets, which is defined as the proportion of duplicate labels among all labels. As shown in Table 1, it reaches **31.6%** overall. These duplicate labels often have different meanings across different datasets and tasks. This phenomenon stems from the diverse research focuses of materials scientists, which lead to significant differences in information needs for the same labels, as illustrated in Table 2. The duplicate rate further underscores the necessity of supplementing these labels to clarify their actual meaning.

To address this, inspired by scientists' practice of establishing annotation guidelines for consistency and accuracy (Friedrich et al., 2020), we propose enhancing the labels by retrieving their descriptions from these guidelines, as these guidelines can give a basic outline of the scientists' information needs, as shown in Figure 5. Specifically, Each label d in D will serve as a retrieval

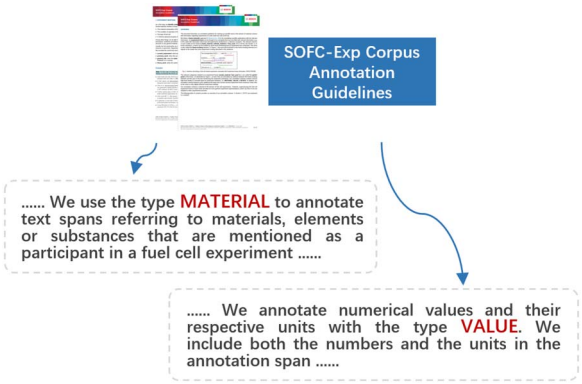


Figure 5: The dataset's annotation guidelines, exemplified by the SOFC-Exp Corpus (Friedrich et al., 2020), provide precise definitions of task labels, reflecting the scientists' needs in MSTM.

keyword to get its description $c^d \in C^d$ (used as the class comment in Section 3.2, and provided for other baselines) from annotation guideline G.

$$c^d = \begin{cases} \text{Retriever}(d, G), & \text{if } d \text{ in } G, \\ \text{Reasoner}(d, G), & \text{if } d \text{ not in } G. \end{cases} \quad (1)$$

In this work, we use prompt-based LLMs for retrieval by providing the annotation guideline text as context and labels as keywords through a pre-defined prompt template. If labels are not found in the annotation guidelines, LLM will infer this description based on other label descriptions and contextual information, thereby reducing the potential for hallucination. In this case, the LLM functions as a reasoner rather than a retriever. Part of the results can be found in Table 2, which showcases the varying needs of scientists across different studies.

4.3 Baselines and Evaluation Metrics

To analyze the performance of ClassMATe, we present the following commonly used baseline

approaches and several variants of ClassMATe, categorized by their reliance on supervised data:

1. Supervised Learning

Fine-tuned BERTs, such as MatBERT (Walker et al., 2021), MatSciBERT (Gupta et al., 2022), and MELT (Kim et al., 2024), are BERT (Devlin, 2018b) variants pre-trained on the task-specific labeled data, achieving considerable performance based on the training data and strategies.

2. Zero-Shot Learning

API-Accessible LLMs, such as GPT-4o (OpenAI, 2024), represent the state-of-the-art in general-purpose LLMs chosen for their superior text performance.[3] **Open-source LLMs**, such as LLaMA3 (Dubey et al., 2024), are widely used LLMs. Considering the performance improvement validation and low-resource advantages, LLaMA3-8B is selected as the focus of this study. **Domain LLMs**, such as HoneyBee (Song et al., 2023b), are state-of-the-art domain LLMs for MSTM, enhancing zero-shot via fine-tuning on materials science knowledge base. Task-specific methods such as ActionIE (Zhong et al., 2024), due to their limited applicability to broader MSTM tasks, are not included in the comparisons. **ClassMATe♦**, which relies solely on knowledge structuring without active needs refining, can be seen as a zero-shot approach, as it requires no learning samples.

3. Few-Shot Learning

Representative LLMs, such as GPT-4o and LLaMA3-8B, differ from zero-shot approaches by incorporating relevant samples with true labels into the context to improve inference. HoneyBee was not included as its weights are not available. ClassMATe♠ builds upon ClassMATe♦ by incorporating the hard samples-aware active needs refining. In this paper, ClassMATe refers to this complete version by default.

Evaluation Metrics: Similar to Song et al. (2023a,b) and Kim et al. (2024), the performance is evaluated with two widely adopted metrics: Micro-F1 and Macro-F1.

4.4 Implementation Details

Because ClassMATe is a plug-and-play approach, high-performance LLMs such as GPT-4o and open-source LLMs such as LLaMA3-8B were adopted as the backbone. All LLMs referenced in ClassMATe are based on their backbones, and LLaMA3-8B-based approaches are evaluated on two NVIDIA RTX-4090 GPUs. To ensure consistent experimental results, LLMs were used with a temperature setting of 0; the random seed was fixed, and the results were averaged for the final results. In ClassMATe, based on experimental analysis, the attribute number of ClassDefinition was set to 5, with 100 sampled texts for active needs refining and a maximum of 3 refining iterations. In terms of dataset configuration, our setup follows that of Song et al. (2023a). More details of the BERT series and Honeybee series methods in the baseline can be found in Song et al. (2023a,b) and Kim et al. (2024).

5 Evaluation Results

RQ1: How do existing MSTM approaches perform?

As shown in the results in Table 3, existing general-purpose LLMs exhibit objective performance, which can be attributed to the extensive material domain text included in their pre-training datasets and their strong generalization capabilities in text tasks. Although LLaMA3-8B performs slightly worse than GPT-4o, its open-source nature makes this trade-off acceptable. On the other hand, HonnyBee, the method fine-tuned for materials science based on the original LLaMA-7B and LLaMA-13B (Touvron et al., 2023), shows limited advantages in zero-shot performance compared to rapidly evolving general-purpose LLMs. We argue that fully leveraging the material knowledge embedded in LLMs might be a more effective strategy than domain-specific fine-tuning for materials scientists. Notably, the fine-tuned BERT series models still exhibit a performance advantage over the LLM-based zero-shot and few-shot approaches, making them a viable option, especially when labeled datasets are available.

RQ2: Is our approach competitive against existing MSTM approaches?

Outperformance Compared to Zero-Shot or Few-Shot Learning Baselines Our proposed zero-shot method, ClassMATe♦, using LLaMA3-8B and GPT-4o as backbone LLMs, significantly outperforms both general-purpose LLMs,

[3] https://openai.com/index/hello-gpt-4o/.

Model	Named Entity Recognition	Relation Extraction	Event Argument Extraction	Paragraph Classification	Synthesis Action Retrieval	Sentence Classification	Slot Filling	Overall All Tasks
Supervised Learning Performance								
BERT	0.657	0.782	0.418	0.665	0.656	0.910	0.520	0.658
(Devlin, 2018a)	0.461	0.494	0.225	0.532	0.515	0.633	0.257	0.439
SciBERT	0.738	0.818	0.458	0.671	0.701	0.909	0.500	0.693
(Beltagy et al., 2019)	0.517	0.600	0.290	0.568	0.528	0.612	0.258	0.482
MatSciBERT	0.707	0.791	0.436	0.719	0.692	0.914	0.436	0.671
(Gupta et al., 2022)	0.470	0.507	0.251	0.623	0.484	0.660	0.194	0.456
MatBERT	0.875	0.804	0.451	0.756	0.717	0.909	0.548	0.722
(Walker et al., 2021)	0.630	0.513	0.288	0.691	0.594	0.614	0.273	0.517
DAS	0.770	0.848	0.478	0.672	0.778	0.902	0.592	0.725
(Ke et al., 2023)	0.567	0.628	0.292	0.607	0.641	0.607	0.356	0.528
MELT	0.786	0.860	0.498	0.728	0.798	0.911	0.610	0.741
(Kim et al., 2024)	0.593	0.620	0.341	0.647	0.685	0.613	0.395	0.556
Zero-Shot Learning Performance								
Honybee-7B	0.267	0.245	0.290	0.490	0.688	0.490	0.393	0.409
(Song et al., 2023b)	0.190	0.178	0.189	0.343	0.342	0.365	0.289	0.271
Honybee-13B	0.429	0.412	0.481	0.611	0.801	0.589	0.578	0.557
(Song et al., 2023b)	0.372	0.346	0.378	0.467	0.429	0.503	0.423	0.417
LLaMA3-8B	0.639	0.627	0.539	0.674	0.790	0.589	0.607	0.638
(Dubey et al., 2024)	0.539	0.571	0.488	0.488	0.694	0.495	0.586	0.552
ClassMATe♦	0.662	0.636	0.535	0.810	0.813	0.847	0.709	0.716
(LLaMA3-8B)	0.543	0.569	0.489	0.801	0.758	0.582	0.694	0.634
GPT-4o	0.760	0.643	0.503	0.734	0.869	0.604	0.752	0.695
(OpenAI, 2024)	0.681	0.560	0.487	0.692	0.823	0.513	0.717	0.639
ClassMATe♦	0.782	0.649	0.540	0.821	0.912	0.757	0.822	0.755
(GPT-4o)	0.706	0.581	0.478	0.816	0.869	0.584	0.755	0.684
Few-Shot Learning Performance								
LLaMA3-8B	0.645	0.636	0.526	0.689	0.785	0.615	0.595	0.642
(Dubey et al., 2024)	0.543	0.574	0.480	0.496	0.691	0.509	0.578	0.553
ClassMATe♠	0.712	0.663	0.543	0.873	0.865	0.867	0.752	0.754
(LLaMA3-8B)	0.589	0.530	0.496	0.848	0.812	0.672	0.735	0.669
GPT-4o	0.768	0.647	0.510	0.758	0.874	0.623	0.746	0.704
(OpenAI, 2024)	0.685	0.559	0.481	0.716	0.823	0.521	0.712	0.642
ClassMATe♠	0.823	0.679	0.548	0.856	0.904	0.793	0.849	0.779
(GPT-4o)	0.738	0.601	0.504	0.843	0.863	0.589	0.786	0.703

Table 3: Performance results of our proposed ClassMATe and the baseline methods under zero-shot learning, few-shot learning and supervised learning. Macro-F1 (top) and micro-F1 (bottom) scores are presented, with the best, second-best, and third-best zero/few-shot MSTM approaches highlighted.

achieving improvements of **7.8%** and **6.0%**, respectively, despite the results of these LLMs are already based on our best efforts in prompt design. This result strongly validates the potential of leveraging the LLMs' internal materials knowledge and the effectiveness of the ClassDefinition-based knowledge structuring for MSTM. Moreover, it is noteworthy that by employing LLaMA3-8B as the backbone, ClassMATe outperforms the paid GPT-4o. This makes low-cost, large-scale text mining feasible, especially compared to the token costs associated with API usage. Furthermore, our proposed ClassMATe♠, after introducing the hard samples-aware active needs refining strategy, achieves performance improvements of **11.2%** and **7.5%** over baseline methods in the few-shot setting, validating the effectiveness of our proposed active knowledge structuring, which actively clarifies scientists' personalized needs and optimizes the utilization of internal knowledge to enhance LLM inference.

Outperformance Compared to Supervised Learning Baselines Compared to these supervised learning methods, our proposed ClassMATe (GPT-4o) achieves an F1 score of **0.775**, surpassing the optimal supervised learning model MELT (Kim et al., 2024). This reflects that ClassMATe can achieve performance comparable to supervised learning without requiring task-specific

Model	Named Entity Recognition	Relation Extraction	Event Argument Extraction	Paragraph Classification	Synthesis Action Retrieval	Sentence Classification	Slot Filling	Overall All Tasks
Performance of ClassMATe♦ and its variants (Code Style)								
ClassMATe♦	0.662	0.636	0.535	0.810	0.813	0.847	0.709	0.716
(LLaMA3-8B)	0.543	0.569	0.489	0.801	0.758	0.582	0.694	0.634
w/o	0.548	0.596	0.515	0.738	0.627	0.826	0.640	0.641
ClassComment	0.517	0.524	0.482	0.735	0.683	0.510	0.622	0.582
w/o	0.658	0.614	0.513	0.587	0.789	0.604	0.686	0.636
Attributes	0.562	0.543	0.467	0.563	0.726	0.493	0.642	0.571
w/o	0.513	0.648	0.538	0.564	0.589	0.356	0.656	0.552
ClassCom.&Attr.	0.418	0.574	0.497	0.563	0.530	0.337	0.639	0.508
Performance of LLaMA3-8B, GPT-4o, and their variants (Natural Language Style)								
LLaMA3-8B	0.639	0.627	0.539	0.674	0.790	0.589	0.607	0.638
(Dubey et al., 2024)	0.539	0.571	0.488	0.488	0.694	0.495	0.586	0.552
LLaMA3-8B	0.602	0.614	0.528	0.623	0.742	0.630	0.548	0.612
+ Attributes	0.496	0.516	0.485	0.492	0.604	0.527	0.532	0.521
GPT-4o	0.760	0.643	0.503	0.734	0.869	0.604	0.752	0.695
(OpenAI, 2024)	0.681	0.560	0.487	0.692	0.823	0.513	0.717	0.639
GPT-4o	0.752	0.636	0.508	0.769	0.876	0.641	0.748	0.704
+ Attributes	0.669	0.548	0.479	0.747	0.835	0.527	0.683	0.641

Table 4: Performance comparison of the variants of ClassMATe♦ (Code Style), as well as LLaMA3-8B and GPT-4o (Natural Language Style). Macro-F1 (top) and micro-F1 (bottom) scores are presented.

data labeling or model fine-tuning, implying the possibility for direct practical applications in different subfields based on materials scientists' research interests. Furthermore, with only a small amount of labeled samples, ClassMATe based on LLaMA3-8B can achieve similar performance, revealing the great potential in bridging the gap between LLMs' internal knowledge and scientists' personalized needs.

RQ3: How does ClassDefinition-based knowledge structuring affect performance?

To effectively evaluate the class definition-based approach's validity, we use the ClassMATe♦ as the baseline LLM. Considering the practicality and cost-effectiveness, the LLaMA3-8B is primarily considered the backbone of this study.

Label Descriptions In the previous section about the dataset, we proposed supplementing task labels by retrieving annotation guidelines that offer the label definitions, thereby improving the validity of the dataset, making it more suitable for promot-based LLM understanding. To analyze their contribution, an ablation experiment was conducted with following variant:

- **w/o ClassComment**: It refers to the ClassMATe♦ that directly generates class comment using LLM, without employing label descriptions as class comment.

According to the results in Table 4, when common materials science domain meanings of labels are generated to replace the label description as the class comment in class definition, a performance drop to 0.641 is observed. This reflects the necessity of introducing label descriptions to support a clear understanding of the meaning of labels by LLMs, further emphasizing the importance of focusing on scientists' personalized needs.

ClassDefinition Stylization To further analyze the contribution of ClassDefinition-based knowledge structuring, another ablation experiment was conducted:

- **w/o Attributes**: It refers to the ClassMATe♦ that excludes attributes and their comments in class definitions.
- **w/o ClassCom.&Attr.**: It refers to the ClassMATe♦ that excludes both the class comment and attributes in class definitions.

From Table 4, it can be seen that an 8.0% overall performance decreases when attributes are removed. These attributes, generated by the backbone LLM, explicitly aggregate the relevant knowledge for inference, demonstrating the effectiveness of incorporating attributes and their comments to leverage LLMs' material knowledge. Further removal of class comments leads to a performance decline to 0.552, below the natural

language-based LLaMA3-8B. This drop may be due to a reduced proportion of label information in the overall context, consistent with findings from Sainz et al. (2023). It is worth noting that the RE and EAE tasks maintain stable performance, probably resulting from task constraint information in these tasks' class definitions compensating for the additional task knowledge.

For further analysis, an experiment was conducted with the following variants of the natural language-based approaches, including LLaMA3-8B and GPT-4o as baselines:

- **+ Attributes**: This refers to a variant of the natural language-based LLaMA3-8B or GPT-4o that incorporates attribute information into the label descriptions.

As shown in Table 4, the addition of attributes&comments caused a performance drop for LLaMA3-8B due to ineffective use of the extra information. GPT-4o, benefiting from its stronger contextual understanding, achieved a modest gain of 0.9 by better leveraging the attributes, yet remained less effective than our proposed ClassMATe♦. This suggests that, given equivalent label information, our proposed Class-Definition-based approaches offer a distinct advantage over natural language approaches. They utilize the structured representation of class definitions to effectively apply material knowledge in LLMs to enhance inference.

Number of Attributes To further validate the effectiveness of attributes in class definitions, the evaluation will assess how varying the number of attributes affects knowledge aggregation performance. As shown in Figure 6, the performance of ClassMATe♦ (LLaMA3-8B) is evaluated across varying numbers of attributes. The results indicate a positive correlation between the number of attributes and the performance. It can be seen that performance improves as the number of attributes increases, with the peak observed at 5 attributes. However, as the number of attributes continues to rise, the benefits start to diminish. This decline might be due to an overabundance of attributes, which can reduce the relative importance of label-specific information and lead to negative effects from overly long contexts. Overall, the analysis of different attribute numbers on performance confirms the effectiveness of attributes in aggregating LLMs' knowledge for inference.

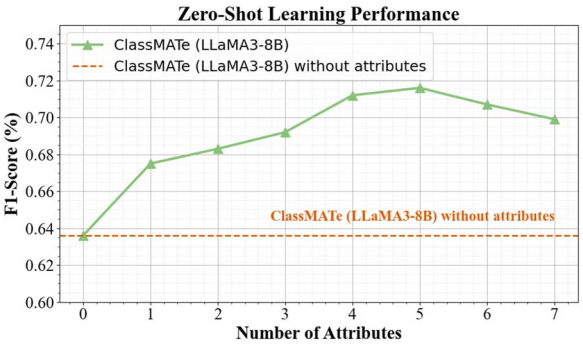


Figure 6: The performance under different attribute numbers, revealing ClassMATe effectively aggregates LLM knowledge for inference, as the attributes were originally generated by the LLMs themselves.

RQ4: How effective is the proposed uncertainty-aware active needs refining ?

Main Limitations of LLMs in MSTM To gain deeper insights into the main limitation of LLMs currently faced in MSTM, we analyzed the main sources of confusion during the inference process of LLMs. Specifically, we additionally introduced two possible options (***Option 1****: Lack of understanding of the current materials science text*; ***Option 2****: Understanding the current text but failing to find a matching task label*) into the prompt context for LLMs to select during inference. Despite rapid progress in LLMs, domain-specific models, even with extensive fine-tuning, often lag behind general-purpose LLMs. Knowledge injection methods also face challenges in fields like materials science due to limited domain knowledge. This raises the question: Is it possible to enhance inference by fully utilizing LLMS' internal materials knowledge?

As shown in Figure 7, the statistics of the selected results reveal that the main limitation in LLM performance arises from Option 2, which accounts for 65.3%. This finding can lead to two key insights: 1) LLMs likely possess a reasonable level of inherent material knowledge, suggesting that they already contain sufficient material knowledge, and 2) Current label descriptions are insufficient, as task labels critically represent objectives and reflect scientists' personalized needs. Based on this, we argue that it is necessary for LLMs to accurately capture and address scientist'

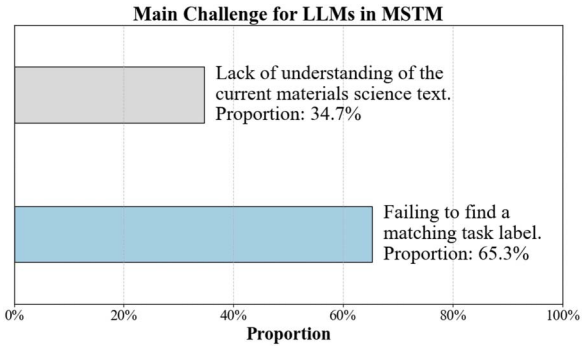


Figure 7: The main challenge faced by LLMs in MSTM, which stems more from failing to find a matching task label than from a lack of understanding of the current materials science text.

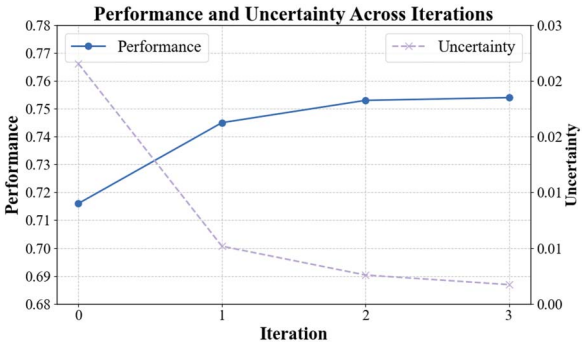


Figure 8: Performance and uncertainty with increasing iterative learning rounds. As learning progresses, uncertainty decreases and performance improves, demonstrating the effectiveness of uncertainty-aware active needs refining.

needs to fully leverage their internal knowledge for enhancing MSTM.

Effect of Uncertainty-Aware Active Needs Refining Previous results show that label descriptions provided by materials scientists at the beginning are often insufficient relative to their personalized needs for LLMs. To address this, we propose an uncertainty-aware active needs refining strategy, continuously learning from hard samples and refining label descriptions to better capture personalized research needs.

To validate the effectiveness of this method, we evaluate the performance improvements over multiple rounds of iterative refining, using the complete ClassMATe (LLaMA3-8B) as the LLM being evaluated. As shown in Figure 8, performance improves with each iteration while uncertainty decreases. This confirms the rationale behind selecting uncertainty-aware hard samples

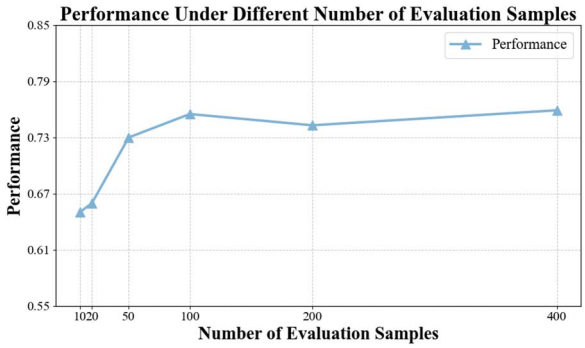


Figure 9: The performance of active needs refining with different evaluation sample sizes. Based on the result, around 100 samples are considered optimal for balancing performance and inference overhead.

for scientists to label, followed by observation and learning. It further demonstrates that the uncertainty in LLMs' responses can, to some extent, reflect the clarity of scientists' needs, addressing the main limitation of LLMs in MSTM. Furthermore, the results in Figure 8 show that performance improves significantly during the first 2 rounds, with diminishing gains in the 3rd round. This suggests that most of the performance improvements occur early in the process. To avoid excessive learning overhead, the refining process of ClassMATe is limited to a maximum of 3 rounds.

Number of Evaluation Samples In the process of active needs refining, evaluating the overall uncertainty of the samples requires multiple rounds of inference across all data. However, this approach is not ideal for MSTM tasks that involve large volumes of texts. To address this challenge, we propose a solution that randomly selects a small subset of texts as task-specific data for the LLM's evaluation and learning. In this context, the active needs refining strategy starts by learning from a small set of texts and then directly applying the refined scientific needs (specifically, the optimized label class definitions) to the entire dataset. Therefore, a detailed analysis of the number of texts selected is essential, as it impacts both performance and the overhead from multiple inference rounds.

Testing with various sample sizes, as shown in Figure 9, reveals that around 100 texts achieve relatively better performance while keeping the sample size in multiple inference rounds manageable. Additionally, this somewhat reflects the

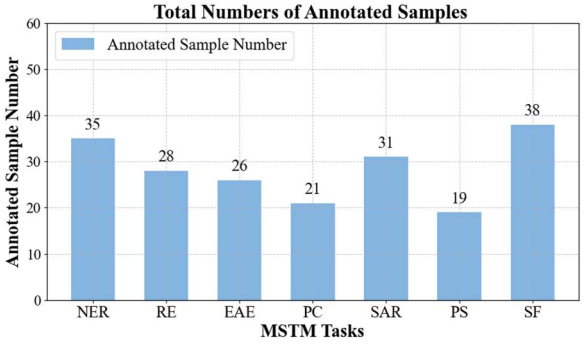


Figure 10: The number of hard samples that require judgment by materials scientists in uncertainty-aware active needs refining. Overall, only a small number of samples, considered acceptable, need to be annotated.

relatively uniform data distribution of the benchmark dataset. Based on this, 100 text samples were selected for our work. While increasing the sample size may result in slight performance gains, it comes with a trade-off in inference overhead.

Number of Annotated Samples After limiting the scope of the learning samples, we further conduct a more detailed statistical analysis to determine how many samples scientists need to annotate during the refining process. This allows for an accurate estimation of the total number of labeled samples required under the setting of few-shot learning. As shown in Figure 10, the results demonstrate that the number of samples needed for judgment generally falls between 20 and 30, which remains within an acceptable range. This is attributed to the design in Algorithm 1, where the number of hard samples per iteration is not fixed but grows progressively. New samples are only introduced for learning if the overall uncertainty remains unchanged. Moreover, the overlap between hard samples selected in each round further reduces the total number of samples that need to be annotated.

6 Further Discussion

Performance with DeepSeek-R1 As the DeepSeek-R1 (DeepSeek-AI, 2025) has recently gained attention for its capabilities in deep reasoning, it is further considered an important benchmark for evaluating our proposed methods. In this section, we compare the performance and time cost of representative methods and proposed ClassMATe variants with ClassMATe♠ (DeepSeek-R1-Distill-Llama-8B)[4] and ClassMATe♠(DeepSeek-R1).[5] As shown in Table 5, ClassMATe based on DeepSeek-R1-Distill-Llama-8B does not yield performance improvements compared to its counterpart using the original LLaMA3-8B. This may be attributed to the limited capacity of the 8B-parameter model, where fine-tuning with a focus on deep reasoning potentially compromises its internal coding ability and materials-related knowledge. Additionally, R1 may be more suitable for complex tasks that require multi-step reasoning to reach conclusions, thus offering limited advantages in MSTM tasks. On the other hand, the full-scale DeepSeek-R1-based ClassMATe♠ accessed via API achieves improvements over GPT-4o-based ClassMATe♠. This gain can be attributed to its 671B parameters and enhanced reasoning capability. However, a key drawback lies in its prolonged inference process during generation, which substantially slows down response time and limits the overall inference efficiency.

Comprehensive Comparison of MSTM Approaches To understand the workflow of various MSTM methods in real-world applications, as shown in Table 5, we divide the overall process into two phases: preparation and inference. **The preparation phase encompasses all steps prior to task execution**, including data annotation, model training, and prompt construction and refinement. **The inference phase refers to performing the MSTM task inference once the preparation is finalized**. It can be observed that BERT-based methods (e.g., MELT) require extensive annotated data and task-specific training during preparation, which can take days or weeks and pose challenges for materials scientists without model training experience. In return, they offer fast inference through classification over vectorized text. LLM-based methods mainly involve prompt design during preparation, with only a small amount of labeled data needed for refinement in more fine-grained and personalized scenarios. As discussed in RQ4 regarding the Number of Evaluation Samples, evaluation can be performed on a subset of the full data. This allows for iterative refinement without incurring substantial time costs, typically staying within minutes

[4] https://huggingface.co/deepseek-ai/DeepSeek-R1.

[5] https://api-docs.deepseek.com/.

Model	Performance (Macro-F1)	Time Cost (Preparation)	Time Cost (Inference)
MELT (Kim et al., 2024)	0.741	Days˜Weeks (Annotation&Training)	68.2 instances/s
LLaMA3-8B (Dubey et al., 2024)	0.638	Mins (Prompt)	4.8 instances/s
GPT-4o (OpenAI, 2024)	0.695	Mins (Prompt)	1.3 instances/s
ClassMATe♦ (LLaMA3-8B)	0.716	Mins (Prompt)	5.1 instances/s
ClassMATe♦ (GPT-4o)	0.755	Mins (Prompt)	1.4 instances/s
ClassMATe♠ (LLaMA3-8B)	0.754	Mins˜Hours (Prompt&Refining)	5.1 instances/s
ClassMATe♠ (GPT-4o)	0.779	Mins˜Hours (Prompt&Refining)	1.4 instances/s
ClassMATe♠ (DeepSeek-R1-Distill-Llama-8B)	0.747	Hours (Prompt&Refining)	0.1 instances/s
ClassMATe♠ (DeepSeek-R1)	0.785	Hours (Prompt&Refining)	0.1 instances/s

Table 5: The comparison covers performance, time cost of preparation, and time cost of inference across the following methods: MELT (fine-tuned, BERT-based), LLaMA3-8B and GPT-4o (zero-shot, natural language-based), ClassMATe♦ (zero-shot, under LLaMA3-8B and GPT-4o), and ClassMATe♠ (few-shot, under LLaMA3-8B, GPT-4o, DeepSeek-R1-Distill-Llama-8B and DeepSeek-R1).

Model	Effectiveness	Task-Adaptability	Efficiency	Hardware	API-Cost
MELT (Kim et al., 2024)	☆		☆		☆
LLaMA3-8B (Dubey et al., 2024)		☆	☆		☆
GPT-4o (OpenAI, 2024)		☆	☆	☆	
ClassMATe♦ (LLaMA3-8B)		☆	☆		☆
ClassMATe♦ (GPT-4o)	☆	☆	☆	☆	
ClassMATe♠ (LLaMA3-8B)	☆	☆	☆		☆
ClassMATe♠ (GPT-4o)	☆	☆	☆	☆	
ClassMATe♠ (DeepSeek-R1-Distill-Llama-8B)	☆	☆			☆
ClassMATe♠ (DeepSeek-R1)	☆	☆		☆	

Table 6: The comparison of representative methods and proposed ClassMATe variants in terms of effectiveness, task adaptability, efficiency, hardware requirements, and API-based token cost, where ☆ indicates the relative strength in the corresponding aspect. Different methods have their strengths and weaknesses, allowing for selection based on specific experimental conditions and research needs.

to a few hours, as observed in our experiments. However, their token-by-token generation leads to slower inference, but the latency is generally acceptable. R1-based methods are better suited for small-scale scenarios due to their relatively lengthy reasoning process.

To provide a comprehensive comparison, a multi-dimensional analysis is conducted. As shown in Table 6, for scenarios with high-performance demands, MELT, ClassMATe♦ (GPT-4o), and ClassMATe♠ variants (particularly when personalized needs are involved) are recommended. For materials scientists who wish to quickly get started and adapt to diverse MSTM tasks, LLM-based methods are preferable due to their minimal data annotation and model training requirements. When processing texts of considerable scale, methods other than R1 are more suitable; in extremely large-scale scenarios, BERT-based models like MELT remain the most practical choice. Regarding hardware and API cost, the choice depends on available resources. If GPU resources are available, local deployment can be considered, as our LLaMA3-8B-based methods require only a single RTX 4090 or even lower without the need for significant hardware or power investment. For cases lacking such hardware or requiring maximum performance, API-based ClassMATe♠ is recommended, but careful consideration is needed as token costs can increase with data scale. Alternatively, running ClassMATe on rented cloud resources provides a viable option, reducing infrastructure investment. While rental fees apply, it may offer cost savings compared to API usage, depending on computational needs.

7 Conclusion

In this paper, we introduce ClassMATe, a simple and effective approach for Zero/Few-Shot MSTM tasks, which through active knowledge structuring enables LLMs to effectively address the gap between the material knowledge of LLMs and the personalized needs of scientists. As a task-agnostic ICL method, ClassMATe can be rapidly applied to various evolving LLMs and different MSTM tasks, making it user-friendly for materials scientists without the need of domain-specific fine-tuning or additional knowledge bases, thus effectively lowering the barriers to materials discovery. Experiments show that ClassMATe achieves performance comparable to supervised learning approaches in zero/few-shot settings, revealing the potential of the material knowledge within LLMs.

8 Limitations

One limitation of ClassMATe lies in its uncertainty-aware active needs refining strategy. As noted in RQ4, this approach requires evaluating a subset of task data. Although typically only a small portion is needed, unstable data distributions may demand more samples per round, increasing preparation time. Moreover, while decreasing average uncertainty generally leads to performance gains, this is not always guaranteed. Fortunately, the refinement results are expressed as natural language label descriptions, which remain interpretable and can be reviewed by humans to mitigate potential issues. Another limitation is the lack of empirical validation for cross-domain use. While ClassMATe may generalize to fields like chemistry or medicine by relying on LLMs' built-in knowledge, its effectiveness beyond materials science remains untested. Future work will explore this to support broader scientific discovery.

Acknowledgments

We sincerely thank the editors and reviewers for their thoughtful comments and constructive suggestions. This work was supported by the National Natural Science Foundation of China (no. 62472332, no. 62276196), the Hubei Key Laboratory of Big Data in Science and Technology (Wuhan Library of Chinese Academy of Science) (no. E3KF461001), and the Hubei Provincial International Science and Technology Cooperation Project (no. 2024EHA031).

References

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. Scibert: A pretrained language model for scientific text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3615–3620. https://doi.org/10.18653/v1/D19-1371

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, and Torsten Hoefler. 2024. Graph of thoughts: Solving elaborate problems with large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 17682–17690. https://doi.org/10.1609/aaai.v38i16.29720

John Dagdelen, Alexander Dunn, Sanghoon Lee, Nicholas Walker, Andrew S. Rosen, Gerbrand Ceder, Kristin A. Persson, and Anubhav Jain. 2024. Structured information extraction from scientific text with large language models. *Nature Communications*, 15(1):1418. https://doi.org/10.1038/s41467-024-45563-x, PubMed: 38360817

DeepSeek-AI. 2025. Deepseek-r1: Incentivizing reasoning capability in LLMs via reinforcement learning.

Jacob Devlin. 2018a. BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Jacob Devlin. 2018b. BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805v2*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien

Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, Danny Wyatt, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Francisco Guzmán, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Govind Thattai, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jack Zhang, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, et al. 2024. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783v2*.

Annemarie Friedrich, Heike Adel, Federico Tomazic, Johannes Hingerl, Renou Benteau, Anika Marusczyk, and Lukas Lange. 2020. The sofc-exp corpus and neural approaches to information extraction in the materials science domain. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1255–1268. https://doi.org/10.18653/v1/2020.acl-main.116

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

Shu Huang and Jacqueline M. Cole. 2022. Batterybert: A pretrained language model for battery database enhancement. *Journal of Chemical Information and Modeling*, 62(24):6365–6377. https://doi.org/10.1021/acs.jcim.2c00035, PubMed: 35533012

Zixuan Ke, Yijia Shao, Haowei Lin, Tatsuya Konishi, Gyuhak Kim, and Bing Liu. 2023. Continual pre-training of language models. In *Proceedings of The Eleventh International Conference on Learning Representations (ICLR-2023)*.

Junho Kim, Yeachan Kim, Jun-Hyung Park, Yerim Oh, Suho Kim, and SangKeun Lee. 2024. Melt: Materials-aware continued pre-training for language model adaptation to materials science. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 10690–10703. https://doi.org/10.18653/v1/2024.findings-emnlp.627

Olga Kononova, Haoyan Huo, Tanjin He, Ziqin Rong, Tiago Botari, Wenhao Sun, Vahe Tshitoyan, and Gerbrand Ceder. 2019. Text-mined dataset of inorganic materials synthesis recipes. *Scientific Data*, 6(1):203. https://doi.org/10.1038/s41597-019-0224-1, PubMed: 31615989

Lin Li, Dan Liu, Lingyun Zhao, Jianwei Zhang, and Jinhang Liu. 2022. Evidence mining for interpretable charge prediction via prompt learning. *IEEE Transactions on Computational Social Systems*.

Zixuan Li, Yutao Zeng, Yuxin Zuo, Weicheng Ren, Wenxuan Liu, Miao Su, Yucan Guo, Yantao Liu, Lixiang Lixiang, Zhilei Hu, Long Bai, Wei Li, Yidan Liu, Pan Yang, Xiaolong Jin, Jiafeng Guo, and Xueqi Cheng. 2024. KnowCoder: Coding structured knowledge into LLMs for universal information extraction. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8758–8779, Bangkok, Thailand. Association for Computational Linguistics. https://doi.org/10.18653/v1/2024.acl-long.475

Dan Liu, Lin Li, Xiaohui Tao, Jian Cui, and Qing Xie. 2023. Descriptive prompt paraphrasing for target-oriented multimodal sentiment classification. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 4174–4186. https://doi.org/10.18653/v1/2023.findings-emnlp.275

Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022. Rethinking the role of demonstrations: What makes in-context learning work? In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11048–11064. https://doi.org/10.18653/v1/2022.emnlp-main.759

OpenAI. 2024. Hello gpt-4o. https://openai.com/index/hello-gpt-4o/.

Oscar Sainz, Iker García-Ferrero, Rodrigo Agerri, Oier Lopez de Lacalle, German Rigau, and Eneko Agirre. 2023. Gollie: Annotation guidelines improve zero-shot information-extraction. *arXiv preprint arXiv:2310.03668v5*.

Yu Song, Santiago Miret, and Bang Liu. 2023a. Matsci-NLP: Evaluating scientific language models on materials science language tasks using text-to-schema modeling. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3621–3639. https://doi.org/10.18653/v1/2023.acl-long.201

Yu Song, Santiago Miret, Huan Zhang, and Bang Liu. 2023b. Honeybee: Progressive instruction finetuning of large language models for materials science. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 5724–5739. https://doi.org/10.18653/v1/2023.findings-emnlp.380

Liangtai Sun, Yang Han, Zihan Zhao, Da Ma, Zhennan Shen, Baocai Chen, Lu Chen, and Kai Yu. 2024. Scieval: A multi-level large language model evaluation benchmark for scientific research. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 19053–19061. https://doi.org/10.1609/aaai.v38i17.29872

Manu Suvarna, Alain Claude Vaucher, Sharon Mitchell, Teodoro Laino, and Javier Pérez-Ramírez. 2023. Language models and protocol standardization guidelines for accelerating synthesis planning in heterogeneous catalysis. *Nature Communications*, 14(1):7964. https://doi.org/10.1038/s41467-023-43836-5, PubMed: 38042926

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Vahe Tshitoyan, John Dagdelen, Leigh Weston, Alexander Dunn, Ziqin Rong, Olga Kononova, Kristin A. Persson, Gerbrand Ceder, and Anubhav Jain. 2019. Unsupervised word embeddings capture latent knowledge from materials science literature. *Nature*, 571(7763):95–98. https://doi.org/10.1038/s41586-019-1335-8, PubMed: 31270483

Nicholas Walker, Amalie Trewartha, Haoyan Huo, Sanghoon Lee, Kevin Cruse, John Dagdelen, Alexander Dunn, Kristin Persson, Gerbrand Ceder, and Anubhav Jain. 2021. The impact of domain-specific pre-training on named entity recognition tasks in materials science. *Available at SSRN 3950755*. https://doi.org/10.2139/ssrn.3950755

Lean Wang, Lei Li, Damai Dai, Deli Chen, Hao Zhou, Fandong Meng, Jie Zhou, and Xu Sun. 2023a. Label words are anchors: An information flow perspective for understanding in-context learning. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9840–9855. https://doi.org/10.18653/v1/2023.emnlp-main.609

Xingyao Wang, Sha Li, and Heng Ji. 2023b. Code4struct: Code generation for few-shot event structure prediction. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3640–3663. https://doi.org/10.18653/v1/2023.acl-long.202

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837.

Yuchen Xia, Jiho Kim, Yuhan Chen, Haojie Ye, Souvik Kundu, and Nishil Talati. 2024.

Understanding the performance and estimating the cost of LLM fine-tuning. *arXiv preprint arXiv:2408.04693*. https://doi.org/10.1109/IISWC63097.2024.00027

Ke Yang, Jiateng Liu, John Wu, Chaoqi Yang, Yi R. Fung, Sha Li, Zixuan Huang, Xu Cao, Xingyao Wang, Yiquan Wang, Heng Ji, and Chengxiang Zhai. 2024. If LLM is the wizard, then code is the wand: A survey on how code empowers large language models to serve as intelligent agents. *arXiv preprint arXiv:2401.00812*.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. 2024. Tree of thoughts: Deliberate problem solving with large language models. *Advances in Neural Information Processing Systems*, 36.

Peiliang Zhang, Jingling Yuan, Lin Li, Wen Luo, Jiwei Hu, and Xin Li. 2024. Key substructure learning with chemical intuition for material property prediction. In *International Conference on Database Systems for Advanced Applications*, pages 87–103. Springer. https://doi.org/10.1007/978-981-97-5575-2_6

Xianrui Zhong, Yufeng Du, Siru Ouyang, Ming Zhong, Tingfeng Luo, Qirong Ho, Hao Peng, Heng Ji, and Jiawei Han. 2024. Actionie: Action extraction from scientific literature with programming languages. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12656–12671. https://doi.org/10.18653/v1/2024.acl-long.683