ActionIE: Action Extraction from Scientific Literature with Programming Languages

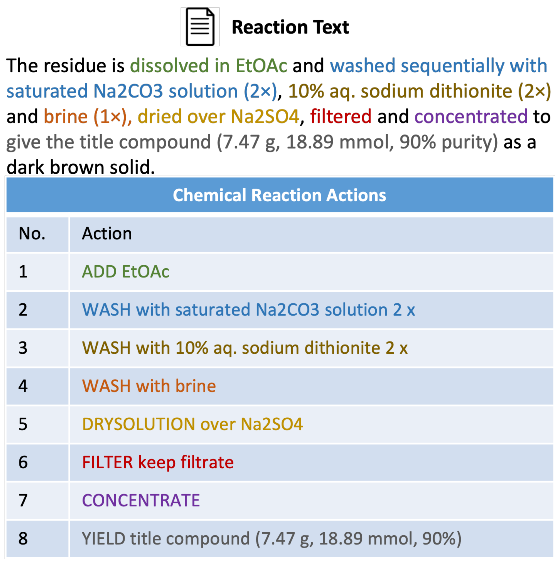
Xianrui Zhong[1*], Yufeng Du[1*], Siru Ouyang[1], Ming Zhong[1], Tingfeng Luo[1], Qirong Ho[2]
Hao Peng[1], Heng Ji[1], Jiawei Han[1]
[1] University of Illinois Urbana-Champaign [2] MBZUAI
{xzhong23,yufengd4, siruo2, mingz5, tluo9, haopeng, hengji, hanj}@illinois.edu
qirong.ho@mbzuai.ac.ae

Abstract

Extraction of experimental procedures from human language in scientific literature and patents into actionable sequences in robotics language holds immense significance in scientific domains. Such an *action extraction* task is particularly challenging given the intricate details and context-dependent nature of the instructions, especially in fields like chemistry where reproducibility is paramount. In this paper, we introduce ACTIONIE, a method that leverages Large Language Models (LLMs) to bridge this divide by converting actions written in natural language into executable Python code. This enables us to capture the entities of interest, and the relationship between each action, given the features of Programming Languages. Utilizing linguistic cues identified by frequent patterns, ActionIE provides an improved mechanism to discern entities of interest. While our method is broadly applicable, we exemplify its power in the domain of chemical literature, wherein we focus on extracting experimental procedures for chemical synthesis. The code generated by our method can be easily transformed into robotics language which is in high demand in scientific fields. Comprehensive experiments demonstrate the superiority of our method. In addition, we propose a graph-based metric to more accurately reflect the precision of extraction. We also develop a dataset to address the scarcity of scientific literature occurred in existing datasets.

1 Introduction

Recently, the integration of Natural Language Processing (NLP) techniques into various scientific fields has achieved significant success (Wang et al., 2019; Soleimani et al., 2022; Song et al., 2023; Lai et al., 2023; Ouyang et al., 2024). Among the applications, extracting information from unstructured



Reaction Text

The residue is dissolved in EtOAc and washed sequentially with saturated Na2CO3 solution (2×), 10% aq. sodium dithionite (2×) and brine (1×), dried over Na2SO4, filtered and concentrated to give the title compound (7.47 g, 18.89 mmol, 90% purity) as a dark brown solid.

Chemical Reaction Actions

No.	Action
1	ADD EtOAc
2	WASH with saturated Na2CO3 solution 2 x
3	WASH with 10% aq. sodium dithionite 2 x
4	WASH with brine
5	DRYSOLUTION over Na2SO4
6	FILTER keep filtrate
7	CONCENTRATE
8	YIELD title compound (7.47 g, 18.89 mmol, 90%)

Figure 1: An example of action extraction from literature that describes a sequence of chemical reaction actions. The text is drawn from Vaucher et al. (2020a).

scientific literature has been one with growing significance (Guo et al., 2022; Zhong et al., 2023a,b). For example, chemists typically look through a wide range of publications to select candidate protocols for one organic synthesis scene, based on their own reading and repetitive trial-and-error procedures (Davies, 2019; Vaucher et al., 2021).

Therefore, structured chemical data, including reaction formulae, chemical entities, and experiment conditions, facilitates effective utilization and automatic analysis, such as indexing and searching by keywords; discovering and analyzing relations between entities; clustering related objects and discovering potential patterns; automatically executing protocols; and predicting and optimizing experiment conditions. Representatively, Figure 1 presents a case of structured chemical experimental procedure, essential for guiding practitioners in their laboratory work (Vaucher et al., 2020b; Zeng et al., 2023). This task involves extracting a

[*]Equal contribution.

sequence of chemical reaction actions from a scientific text passage, where each action is defined by an operation and its corresponding attributes. For instance, in the example "*ADD EtOAc*" shown in Figure1, "*ADD*" represents the operation, and the chemical "*EtOAc*" is the attribute.

However, the discovery of new chemical experimental procedures is scattered across unstructured scientific text and described in various writing styles, posing a significant challenge to the automatic creation of reaction action databases. Existing chemical databases, predominantly commercial ones such as Reaxys (Elsevier B.V., 2023), SciFinder (Chemical Abstracts Service (CAS), 2023), and Pistachio (NextMove Software, 2023), depend extensively on the manual contributions of domain experts. Analyzing, indexing, and utilizing scientific literature typically requires extensive and costly annotation or labeling by human experts. Moreover, this method is prone to errors due to the sheer volume of rapidly expanding scientific data. Despite the considerable manual effort, these databases prioritize storing information on the reactants, products, and reaction conditions, rather than the concrete sequences of chemical actions. This is primarily because manually designing these experiment procedures is both time-consuming and costly.

To tackle this issue, various studies have employed text mining techniques (Hawizy et al., 2011; Swain and Cole, 2016; Vaucher et al., 2020b; Wang et al., 2022b; Zeng et al., 2023) to automatically extract structured information on procedures from unstructured text, leveraging the advancements in NLP field. However, extracting experimental procedures remains a challenging task. One major hurdle is the complexity and variability of scientific language, which often features intricate sentence structures, domain-specific terminology, abbreviations, and acronyms. These elements pose substantial difficulties for sequential tagging-based approaches. For example, as shown in Figure 1, a text describing a series of chemical reaction actions includes the "*WASH*" operation followed by three chemicals. While sequential tagging-based methods might recognize the chemical compounds, they often struggle to accurately identify the operations and associate them with their corresponding attributes. Furthermore, the scarcity of large, annotated datasets poses an additional obstacle to training deep learning models on chemical experimental data effectively.

In this paper, we choose chemical experiment procedures as a case study, and explore the potential of large language models (LLMs) to extract structured data from the complex and domain-specific language in chemical papers and patents. We propose a novel approach that frames the procedure extraction task as a code generation problem, where we express the experimental procedures as a series of pre-defined operations, and utilize the unique features of coding, such as classes, inheritance, and types, to structure this information. Our method leverages the capabilities of LLMs in few-shot in-context learning, reducing the need for large amounts of annotated data, and accelerating the preparation process. Moreover, our proposed framework also offers an easy solution to generate protocols for different automated platforms by applying different language configurations.

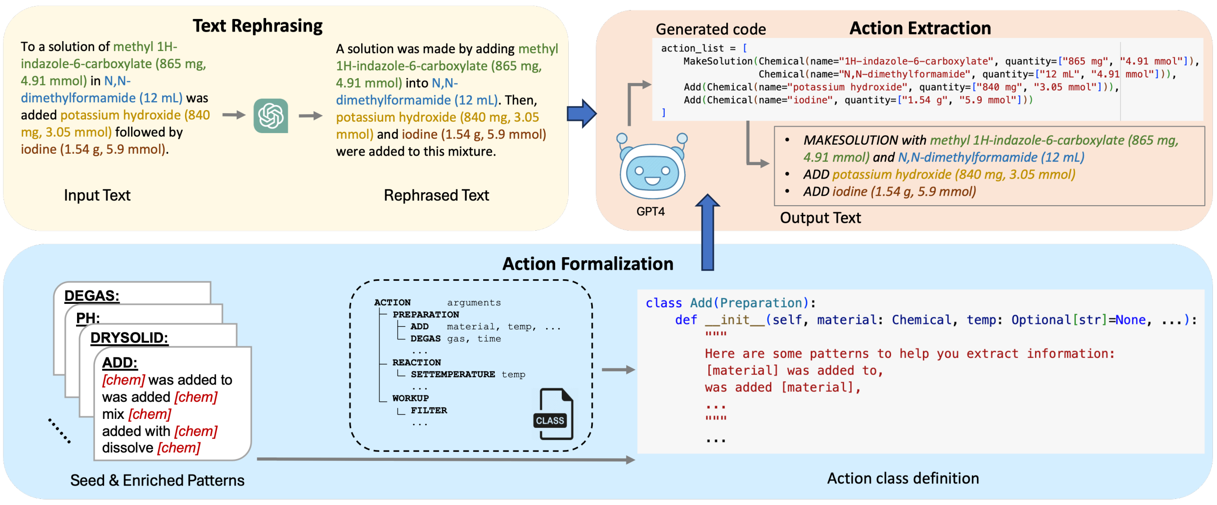
From the perspective of evaluation, we first pinpoint shortcomings within current evaluation metrics for the chemical action extraction task, and propose a novel metric based on graph-matching that substantially improves correlation with human judgments. Existing benchmarks largely concentrate on patent documents, which are inherently well-structured. To more accurately meet the real-world demands of practitioners, we meticulously annotate a test set derived from chemistry literature, which offers a more comprehensive evaluation of model performance. Notably, our new benchmark is considerably more extensive than previous ones, with an average length of 770.8 characters compared to 158.2 characters, providing a testing environment that mirrors realistic scenarios more closely. Experimentally, our method ActionIE demonstrates consistent superiority over strong baseline models, both against traditional benchmarks and our newly established testbed.

2 Related Work

The practice of using NLP in structured scientific data extraction has seen significant advancements, from utilizing traditional NLP techniques to integrating code generation methods into structure extraction, which is especially influenced by the growing capabilities of large language models (LLMs).

2.1 Action Extraction in Chemical Documents

The algorithms for action extraction in chemical texts evolve with the development of NLP. Earlier approaches, such as ChemDataExtractor (Swain


Text Rephrasing
To a solution of methyl 1H-indazole-6-carboxylate (865 mg, 4.91 mmol) in N,N-dimethylformamide (12 mL) was added potassium hydroxide (840 mg, 3.05 mmol) followed by iodine (1.54 g, 5.9 mmol).
A solution was made by adding methyl 1H-indazole-6-carboxylate (865 mg, 4.91 mmol) into N,N-dimethylformamide (12 mL). Then, potassium hydroxide (840 mg, 3.05 mmol) and iodine (1.54 g, 5.9 mmol) were added to this mixture.
Input Text
Rephrased Text
Generated code
Action Extraction
GPT4
Output Text
Action Formalization
DEGAS:
PH:
DRYSOLID:
ADD:
[chem] was added to
was added [chem]
mix [chem]
added with [chem]
dissolve [chem]
Seed & Enriched Patterns
CLASS
Action class definition

```
action_list = [
    MakeSolution(Chemical(name="1H-indazole-6-carboxylate", quantity=["865 mg", "4.91 mmol"]),
                 Chemical(name="N,N-dimethylformamide", quantity=["12 mL", "4.91 mmol"])),
    Add(Chemical(name="potassium hydroxide", quantity=["840 mg", "3.05 mmol"])),
    Add(Chemical(name="iodine", quantity=["1.54 g", "5.9 mmol"]))
]
```

- *MAKESOLUTION with methyl 1H-indazole-6-carboxylate (865 mg, 4.91 mmol) and N,N-dimethylformamide (12 mL)*
- *ADD potassium hydroxide (840 mg, 3.05 mmol)*
- *ADD iodine (1.54 g, 5.9 mmol)*

```
ACTION      arguments
├ PREPARATION
│  ├ ADD    material, temp, ...
│  └ DEGAS  gas, time
│    ...
├ REACTION
│  └ SETTEMPERATURE temp
│    ...
└ WORKUP
   └ FILTER
     ...
```

```
class Add(Preparation):
    def __init__(self, material: Chemical, temp: Optional[str]=None, ...):
        """
        Here are some patterns to help you extract information:
        [material] was added to,
        was added [material],
        ...
        """
        ...
```

Figure 2: Overview of ActionIE.

and Cole, 2016) and ChemicalTagger (Hawizy et al., 2011), used part-of-speech tagging techniques to perform named entity recognition on chemical literature. These methods were fast and effective at extracting key information, but had limited capabilities at handling more complex sentence structures in patent documents. In recent years, the transformer structure has also been introduced to action extraction. Paragraph2Actions (Vaucher et al., 2020b) used a transformer-based encoder-decoder architecture trained on human-annotated data to generate action sequences.

More recent advancements in NLP are led by pretrained LLMs. Wang et al. (2022b) finetuned a BERT model to perform named entity recognition on materials and extract synthesis actions on a dataset of solution-based inorganic materials synthesis. Zeng et al. (2023) finetuned both a T5 model and a GPT-3.5 model on a human-annotated dataset. While these transformer-based models excel in capturing the semantics of diverse scientific language, they rely on human-annotated datasets, which are created under extensive labor from domain experts, and are prone to human errors. Also, these methods hard-code structure definitions inside their framework, and have to infer structure semantics based on the training data, which could lead to inaccuracies if the training data is not representative enough.

Recently, some datasets have been constructed for different tasks in mining chemical patents. Fang et al. (2021) is proposed for evaluating the task of anaphora resolution. He et al. (2020) builds an evaluation benchmark in BART format (Stenetorp et al., 2012) for action extraction in chemical patents, but the ground truth is not public.

2.2 Leveraging Programming Languages for Structure Extraction Tasks

With the overwhelming success of very large decoder-only language models (such as GPT-3 (Brown et al., 2020), GPT-3.5 and GPT-4 (OpenAI, 2023), PaLM 2 (Anil et al., 2023), Llama 2 (Touvron et al., 2023), etc.) on a variety of NLP tasks, recent research has increasingly focused on the application of LLMs for scientific structure extraction tasks. Agrawal et al. (2022) demonstrated the power of zero-shot learning on GPT-3 for extracting information from clinical texts. Dunn et al. (2022) further performed chemical entities and relation extraction with a GPT-3 model finetuned on 500 input-output pairs. Zhong et al. (2023b) uses GPT-4 to capture the roles of chemical entities in scientific text.

On the other hand, the large language models show noteworthy improvement in code generation. Codex (Tyers et al., 2023), finetuned from a GPT model, has shown remarkable abilities in code completion. The recent year has seen the application of GPT-based agents (Hong et al., 2023; Zhou et al., 2023; Wang et al., 2024), which leverage the reasoning and decision abilities of GPT models along with Chain-of-Thought approaches, in programming tasks.

Among these developments of structure extraction and code generation, Code4Struct (Wang et al., 2022a) extracts structured event information from natural language using code generation. It aligns programming constructs, such as class definitions, inheritance, and functions with the entity and event

Module Name	Models
Pattern Mining	Flan-T5-Large & GPT-4-0613
Text Rephrasing	GPT-4-0613
Code Generation	GPT-4-0613
Code to Natural Language	Pre-defined Rules

Table 1: Models used for each module in ActionIE.

types of interest, utilizing both the structural and semantic information of coding.

3 ActionIE Framework

3.1 Task Formulation

Given a text T, we aim to extract all procedures (actions) $P = \{(o_1, a_1), ..., (o_n, a_n)\}, o_i \in S$ mentioned in T in sequence, where S is a set of pre-defined operation types, and a_i is the pre-defined attributes of operation O_i. Note that rather than identifying the specific role a substance plays within a reaction, our task focuses on the category of attribute to which it belongs. Following prior work (Vaucher et al., 2020a), we set the pre-defined operation types as follows: *Add, CollectLayer, Concentrate, Degas, DrySolid, DrySolution, Extract, Filter, MakeSolution, Microwave, Partition, PH, PhaseSeparation, Purify, Quench, Recrystallize, Reflux, SetTemperature, Sonicate, Stir, Triturate, Wait, Wash, Yield, FollowOtherProcedure, InvalidAction, OtherLanguage*, and *NoAction*. Definitions for each action are described in Appendix A.

3.2 Action Extraction with Programming Languages

Previous methods utilize a large amount of rules and patterns provided by human or train a model in a supervised way which require cost-sensitive labelled data. In addition, the definitions of actions may change based on the needs of scientists. Under certain circumstances, re-creating rules and patterns by human may be required for unsupervised methods; and relabelling data may be needed for supervised methods.

Driven by the aforementioned drawbacks, and with the emergence of Large Language Models (LLMs), we propose to use LLMs to tackle this action extraction task, as they have demonstrated promising capabilities in information extraction, particularly in data-scarce scenarios. Naively, one may directly input a paragraph along with all definitions of actions and ask LLMs to extract the action

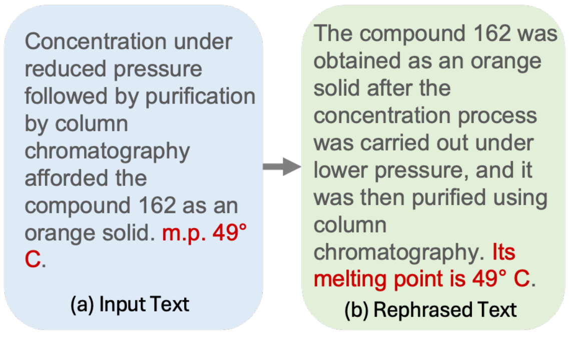

Concentration under reduced pressure followed by purification by column chromatography afforded the compound 162 as an orange solid. m.p. 49° C.
(a) Input Text
The compound 162 was obtained as an orange solid after the concentration process was carried out under lower pressure, and it was then purified using column chromatography. Its melting point is 49° C.
(b) Rephrased Text

Figure 3: Rephrasing Example.

information. However, this approach poses some problems. The first is the well-known hallucination problem of the generation from LLMs (Huang et al., 2023). LLMs may generate actions that are not in the pre-defined action set since LLMs may directly output the verb found in the paragraph as an action or output an action not in the pre-defined set based on its summarization. Furthermore, LLMs may output detailed action sequences while it should summarize some of the actions. For instance, the ground-truth action for text "Add HCl to pH 5." (adding HCl until the pH of the liquid is 5) is "PH with HCl to pH 5.", while LLMs also include the "ADD" action which results in "ADD HCl; PH with HCl to pH5." This demonstrates that LLMs fail to understand the relationship between actions.

In light of these limitations, we propose to reformulate the action extraction task as a code writing problem for LLMs that we transform each action type into a Python class. This has a few advantages. First, the abstract nature of class in programming languages and the relationship between classes including "Inheritance" and "Composition" relationships help LLMs better interpret the relationships between actions. Second, class variables in programming languages enable LLMs to understand what needs to be extracted for each action. Next, it is more suitable for an environment that needs changing the set of actions and the interested information for each action. The users can easily define the operations they want to extract and the attributes for each operation by simply modifying the Python class file which is fed to the LLMs. Finally, this minimizes the gap between natural language and robotics language as it is more convenient to transform the Python code produced by our method to the code that can be executed by robots. Figure 7 demonstrates the prompt we use for code generation.

3.3 External Information Guided Extraction

Text Rephrasing Scientific literature may have its own writing style that is different from ordinary writing, particularly in chemistry literature. We propose to first use LLMs to rephrase the given paragraph for two main reasons. First, rephrasing complex scientific texts into simpler language enhances their comprehensibility for large language models like GPT-4, which are pre-trained on general, non-scientific text sources. Second, it introduces domain knowledge encoded in the language model. This is examplified in the case presented in Figure 3. The original paragraph (a) contains a phrase “m.p. 49 °C”, which is usually been misinterpreted as environment temperature. By leveraging LLMs for rephrasing, “m.p.” is rephrased as “melting point”, shown in Figure 3, and leads to a correct extraction. In practice, we prompt GPT-4 to rephrase the input text, as well as feeding in the mined patterns to keep the structure of the rephrased text as much as possible. Figure 6 shows the prompt we use for text rephrasing.

Pattern-guided Extraction For human, it is possible to identify certain action information, even lack of any prior domain knowledge. Consider an example “Partition between water (100 mL) and ethyl acetate (100 mL)” (Vaucher et al., 2020a). We can identify that “water (100 mL)” and “ethyl acetate (100 mL)” are the chemicals involved in the “Partition” action. This can be accomplished by the guidance of linguistic cues, including the semantics of phrases and the structure of sentences.

Motivated by this observation, we utilize frequent patterns in the text that indicate specific reaction actions as linguistic cues to guide LLMs to extract action information. Take “PH” action as an example, we first use a special token “[Chemical]” to replace all occurrences of the chemical with ChemDataExtractor (Swain and Cole, 2016). Several seed patterns are created, such as *pH [pH] with [Chemical]*. The red [pH] indicates a pH value, and the blue [Chemical] indicates the chemical for adjusting the pH. With a set of seed patterns for each action, we mine the enriched patterns through 1) labeling all occurrences in the corpus with seed patterns, 2) training a Flan-T5 model in a question-answering fashion, 3) re-labeling the corpus with the trained Flan-T5 model, and 4) selecting the most frequent patterns as the enriched patterns. GPT-4-0613 is also used for creating a more diverse set of patterns.

After merging enriched patterns with seed patterns as new seed patterns, we repeat the aforementioned process to mine more reliable patterns iteratively. The whole process is illustrated in Figure 4.

3.4 Extracted Action Evaluation

We observe that some actions are equivalent to each other, for instance, [MakeSolution] with A and B is equivalent to [Add] A; [Add] B, and sometimes the order of actions does not matter. Previous evaluation metrics do not consider the order of actions nor the equivalence between actions, and penalize mismatches. In order to take the order of actions and their equivalences, we propose a graph-based metric called Graph Matching Similarity. Given a sentence t with $n \in \mathbb{Z}$ actions $a_1, a_2, ..., a_n$, and equivalent relations $f : A \to \{A\}$, where A is a set of actions and a_i is an arbitrary action, we first construct its corresponding graph G. Details can be found in Algorithm 1.

We first construct graphs for the ground truth sentence and the sentence to be evaluated, denoted as G_{gt} and G_{query}. Then we find the maximal common subgraph G_{sub} in G_{gt} given G_{query} with the algorithm described in (Hattori et al., 2003). Finally, we calculate the similarity score with Equation 1.

$$\text{score} = \frac{|G_{sub} \cap G_{query}|}{|G_{sub} \cup G_{query}|} \tag{1}$$

The evaluation with human judgements compared with other metrics can be found at Section 4.3.

Algorithm 1 Algorithm for Action Graph Construction

```
Input: Sentence $t = (a_1, a_2, ..., a_n)$ Equivalent
  Relations $f : A \to \{A\}$
Output: Graph $G = (V, E)$
  procedure ConstructGraph($t, f$)
    $V = \{a_1\}$
    for $i \leftarrow 2$ to $n - 1$ do
      $V \cup \{a_i\}; E \cup \{(a_i, a_{i-1}), (a_{i+1}, a_i)\}$
      if $a_i \in D(f)$ then
        $V \cup \{f(a_i)\}$
        $E \cup \{(f(a_i), a_{i-1}), (a_{i+1}, f(a_i))\}$
      end if
    end for
    return $G = (V, E)$
  end procedure
```

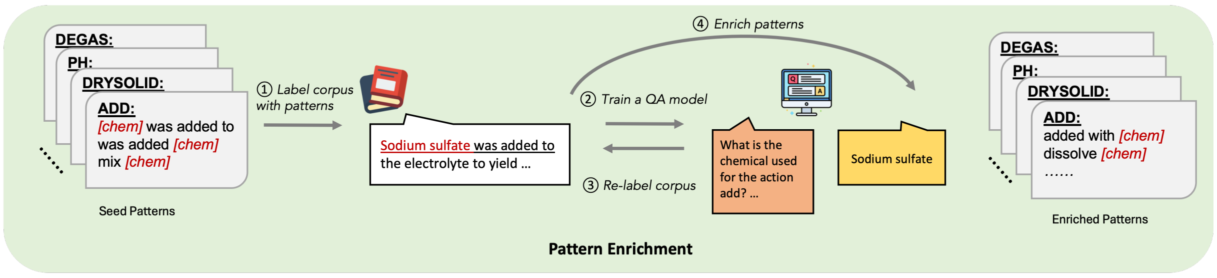


Figure 4: Pattern Enrichment Overview.

4 Experiments

4.1 Experimental Setup

Datasets We evaluate the effectiveness of our method on two datasets. One is the test set used in previous work (Vaucher et al., 2020a), which contains 352 texts related to an experimental procedure for chemical synthesis. We refer this dataset as PATENTACTION as all the paragraphs in it are from patent data. Dataset statistics can be found in Appendix B. The input is a paragraph from chemical literature which contains one or multiple actions. The output is a combination of pre-defined actions in natural language. This dataset is designed for evaluating action extraction in chemical literature setting based on chemist's need.

Since extracting action information from scientific literature is of the same significance as from patent data, collabrating with chemists, we construct a dataset called SCIENTIFICACTION. 100 long paragraphs are collected from ChemRxiv (Cambridge Open Engage, 2023). The average length (number of characters) of paragraphs in ScientificAction is 770.77, while the average length in PatentAction is only 158.24. SCIENTIFICACTION will be available at https://github.com/xianruizhong/ActionIE.

Baselines We compare ActionIE with several state-of-art methods: Paragraph2Actions (Vaucher et al., 2020a), ChemTrans (Zeng et al., 2023), GALACTICA-6.7b (Taylor et al., 2022), and GPT-4 (OpenAI, 2023).

Implementation Details We choose GPT-4-0613 (OpenAI, 2023) as the model for extraction, which supports up to 8,192 tokens. We use "google/flan-t5-large" (Raffel et al., 2020) for linguistic pattern extraction. GPT-4 (OpenAI, 2023) is accessed through OpenAI api. For the parameters of GPT-4 (OpenAI, 2023), we use sampling temperature $t = 0$, and set 500 as the maximum number of new tokens.

Evaluation Metrics for Natural Language Following previous work, we use BLEU score (Papineni et al., 2002) and Levenshtein Similarity (Levenshtein et al., 1966) to evaluate the quality of extracted actions in natural language. Following previous work, the BLEU score is modified since the original BLEU score does not consider short sentences which is common in the test data. The proposed GRAPH MATCHING SIMILARITY is also used for evaluating in the natural language level.

Evaluation Metrics for Operation Level In order to verify the quality of the extracted action sequence in operation level, we use precision, recall, and F1 scores. The sets of operations in ground truth and output are compared, and the attributes are ignored. To better consider the order of operations, we employ SeqMatch-O (SM-O) proposed in Zeng et al. (2023), an evaluation metric for sequence matching in operation level. For details of SeqMatch-O, please refer to Zeng et al. (2023).

Evaluation Metrics for Attribute Level Following previous work, we leverage SeqMatch-A (SM-A) proposed in Zeng et al. (2023) for verifying the quality of attribute-level extraction. For each matched position in SeqMatch-O, the levenshtein similarity is calculated for each argument pair, and the average argument score is used rather than the original 1 in SM-O. Please refer to Zeng et al. (2023) for more details.

4.2 Experimental Results

Results for Extraction in Natural Language The first part of Table 2 represents the results of extraction in natural language in PatentAction dataset. ChemTrans cannot output natural language action sequences, hence, its scores are not calculated. Our proposed ACTIONIE significantly outperforms all baselines in levenshtein similarity, and outperforms all baselines in BLEU except Paragraph2Actions,

Models	BLEU	Levenshtein Similarity	Precision	Recall	F1	Graph Matching Similarity	SM-O	SM-A
***Results for PatentAction** (Avg Length: 158.24)*								
Supervised Methods								
Paragraph2Actions	**0.8511**	0.8927	0.9017	0.9034	0.8985	0.8003	**0.8893**	**0.8629**
ChemTrans	-	-	0.5927	0.4325	0.4866	-	0.4401	-
Few-shot Methods (10-shot)								
Galactica-6.7b	0.0051	0.1336	0.3526	0.2705	0.2732	0.2921	0.1453	0.0534
GPT-4	0.4280	0.6822	0.7537	0.7758	0.7458	0.7923	0.7566	0.6633
ACTIONIE	0.8237	**0.9018**	**0.9126**	**0.9198**	**0.9101**	**0.8136**	0.8880	0.8521
- *Patterns*	0.6829	0.8070	0.8458	0.8220	0.8218	0.8074	0.8248	0.7583
***Results for ScientificAction** (Avg Length: 770.77)*								
Supervised Methods								
Paragraph2Actions	0.4907	0.5380	0.8643	0.5933	0.6633	0.6391	0.5922	0.5118
ChemTrans	-	-	0.9212	0.4583	0.5982	-	0.4924	-
Few-shot Methods (10-shot)								
Galactica-6.7b	-	-	-	-	-	-	-	-
GPT-4	0.4571	0.6625	0.7858	0.7175	0.7312	0.7574	0.6670	0.5137
ACTIONIE	**0.7808**	**0.8394**	**0.9236**	**0.8166**	**0.8584**	**0.8013**	**0.8277**	**0.7087**
- *Patterns*	0.7193	0.8160	0.8942	0.8033	0.8444	0.7980	0.8099	0.6757

Table 2: Overall experimental results. ChemTrans does not support outputting natural language, only the operations are evaluated. Galactica-6.7b fails when the input is too long, therefore, the result is not reported.

Input: The reaction of 1-(3,5-dichlorophenyl)-5-iodoimidazole 2o (0.3 mmol, 102 mg) with additional portion of Oxone® in H2SO4 (scaled down to 0.3 mmol) according to general procedure afforded 5,7-dichlorobenzo[d]imidazo[5,1-b][1,3]iodazol-4-ium hydrogen sulfate 3o as off-white solid, 110 mg, yield: 85%. mp = 205-208°C. 1H NMR (400 MHz, DMSO-d6) δ 9.18 (s, 1H), 8.45 (s, 1H), 7.80 (s, 1H), 7.56 (s, 1H). 13C NMR (100 MHz, DMSO-d6) δ 137.3, 137.1, 136.7, 133.9, 133.3, 126.8, 115.8, 99.5, 95.8. HRMS (ESI/Q-TOF, positive ionization): calcd for C9H4Cl2IN2 + (m/z: [M-HSO4] +): 336.8791, found: 336.8817.

Paragraph2Actions (Error: Only output FollowOtherProcedure action)

1. FollowOtherProcedure

ChemTrans (Error: Only output YIELD action; Misclassify the product as reagent; Yield information only contains percentage)

1. YIELD
   Reagent: 1-(3,5-dichlorophenyl)-5-iodoimidazole 2o, Oxone®, H2SO4, 5,7-dichlorobenzo[d]imidazo[5,1-b][1,3]iodazol-4-ium hydrogen sulfate 3o
2. YIELD
   Yield: 85%
   Reagent: 5,7-dichlorobenzo[d]imidazo[5,1-b][1,3]iodazol-4-ium hydrogen sulfate 3o

GPT4 (Error: Contain an extra SetTemperature action (mp is melting point instead of environment temperature); The order of actions is incorrect; Missing 85% in YIELD action)

1. ADD 1-(3,5-dichlorophenyl)-5-iodoimidazole 2o (0.3 mmol, 102 mg)
2. ADD Oxone® in H2SO4 (scaled down to 0.3 mmol)
3. YIELD 5,7-dichlorobenzo[d]imidazo[5,1-b][1,3]iodazol-4-ium hydrogen sulfate 3o (off-white solid, 110 mg)
4. SetTemperature 205-208 °C
5. FollowOtherProcedure

ActionIE (Perfect)

1. ADD 1-(3,5-dichlorophenyl)-5-iodoimidazole 2o (0.3 mmol, 102 mg)
2. ADD Oxone® in H2SO4 (0.3 mmol)
3. FollowOtherProcedure
4. YIELD 5,7-dichlorobenzo[d]imidazo[5,1-b][1,3]iodazol-4-ium hydrogen sulfate 3o (110 mg, 85%)

Figure 5: Case Study.

but still get a very close score. GALACTICA-6.7 performs poorly as it is not designed for this task. GPT-4 demonstrates its promising performance given its comparable scores with just 10 demonstrations.

The second part of Table 2 represents the result of extraction in natural language in ScientificAction, a more complex and challenging dataset than PatentAction. Paragraph2Actions is trained on patent data and does not generalize well in scientific literature. Sometimes, Paragraph2Actions only outputs FollowOtherProcedure action and ignores other actions described in the input paragraph. Even GPT-4 receive higher scores in levenshtein similarity, demonstrating better generalization than Paragraph2Actions. Ablation study highlights the significance of using patterns as linguistic cues, in all cases, we gain much improvement by utilizing the patterns.

Results for Operation-Level Extraction The middle columns of Table 2 represents the results of operational-level extraction. In the PatentAction dataset, ACTIONIE beats all baselines in precision, recall, and F1 scores, and have very close scores with Paragraph2Actions in SeqMatch-O (0.8880 vs 0.8893).

In the ScientificAction dataset, ACTIONIE outperforms all baselines. Both Paragraph2Actions and ChemTrans are trained on patent data, and achieve a high precision, but have a low recall and F1 scores.

As for the albation study, ACTIONIE benefits significantly from the improvement provided by the patterns, which suggests that the patterns effectively help identify the actions.

Results for Attribute-Level Extraction As listed in the last column of Table 2, in PatentAction dataset, ACTIONIE outperforms all baselines except Paragraph2Actions, but still has a competitive score (0.8521 vs 0.8629).

In ScientificAction dataset, ACTIONIE surpasses all baselines by a substantial margin. Note that GPT-4 receives a slightly higher score than Paragraph2Actions, which further implies the limitation of supervised methods such as Paragraph2Actions and ChemTrans.

4.3 Evaluation Metric Analysis

To better understand how well our proposed GRAPH MATCHING SIMILARITY metric aligns with human evaluation, we randomly sample 100 outputs produced by Paragraph2Actions, GPT-4, and ACTIONIE, which are then given a score by chemists from 1 to 5. We calculate three correlation coefficients, Pearson, Spearman, and Kendall's Tau. As the results shown in Table 3, the proposed GRAPH MATCHING SIMILARITY is better aligned with human judgements than BLEU and Levnshtein Similarity.

Metric Name	Pearson	Spearman	Kendall's Tau
BLEU	0.1791	0.2427	0.2055
Levenshtein Similarity	0.1742	0.2603	0.2179
Graph Matching Similarity	**0.3144**	**0.2976**	**0.3058**

Table 3: Metric Correlations with Human Judgements.

4.4 Case Study

We randomly sample an example from SCIENTIFIC-PATENT and study the output of different methods (see Figure 5). Paragraph2Actions only outputs FollowOtherProcedure action, and it has been noticed that it consistently does so whenever the input mentions another procedure. While the model is supervised to do so, this is an unwanted behavior since the output would ignore any other actions mentioned in the text. ChemTrans only captures the YIELD action, though it includes many details of the reagent. However, ChemTrans will fail if we are also interested in the melting point (mp) of the product given it is a supervised method. It also misclassifies the product as reagent. GPT-4 correctly extracts most of the actions and their attributes while missing the first ADD action, and the order of actions is wrong.

5 Conclusion and Future Work

In this paper, we propose ACTIONIE, a framework for extracting experimental action sequences from scientific literature. Our approach leverages the strength of LLMs by transforming the action extraction problem into a coding question for LLMs. Additionally, it incorporates text rephrasing and linguistic knowledge which further improve the overall performance. To more accurately evaluate the extraction quality, we introduce a graph-based metric, GRAPH MATCHING SIMILARITY. We have also developed a dataset, SCIENTIFICACTION, to offset the lack of scientific literature occurred in previous datasets. Experiments demonstrate that ACTIONIE outperforms state-of-the-art baselines and GRAPH MATCHING SIMILARITY is more aligned with human judgements than previous evaluation metrics. For future developments, one exciting yet challenging direction is to explore deeper into different aspects of the extraction process and integrating these parts into an automated workflow that transforms scientific papers into actionable experiments. This contains identifying relevant paragraphs from scientific papers that describe experimental procedures, creating a robotic system that runs the extracted chemical actions, and automated outcome validation.

Limitation

The limitations of this paper are stated as follows:

1. In our experiments, we use GPT-4 as the backbone model through OpenAI's API. Although ACTIONIE can be incorporated with other causal language models, the performance may change when using different language models. In addition, the performance might be changed by the modification of GPT-4 since its performance may be different over time (OpenAI, 2023). Replacing the GPT-4 API with a static large language model such as Llama-2 (Touvron et al., 2023) could alleviate this issue, but this may require considerable computing resources, which are often limited.

2. Although the dataset proposed in this paper is collected from scientific literature and is much longer than previous datasets, it is still shorter than a scientific paper. Extracting information from a full paper may not be possible if it is too long, given that current GPT-4 API has token limits. Integrating a text segmentation module may be one direction to solve this problem. Another direction may be deploying techniques that reduce the token limits (Bertsch et al., 2023).

Acknowledgement

We thank Professor Huimin Zhao, Xuan Liu, and Hongxiang Li from the University of Illinois Urbana-Champaign for their thoughtful discussions and expertise in chemistry. We sincerely appreciate the anonymous ARR reviewers for their helpful comments.

Research was supported in part by the Molecule Maker Lab Institute: An AI Research Institutes program supported by NSF under Award No. 2019897, US DARPA KAIROS Program No. FA8750-19-2-1004 and INCAS Program No. HR001121C0165, National Science Foundation IIS-19-56151, and the Institute for Geospatial Understanding through an Integrative Discovery Environment (I-GUIDE) by NSF under Award No. 2118329. Any opinions, findings, and conclusions or recommendations expressed herein are those of the authors and do not necessarily represent the views, either expressed or implied, of National Science Foundation, DARPA or the U.S. Government.

References

Swamy R. Adapa, Gregory A. Hunter, Narmin E. Amin, Christopher Marinescu, Andrew Borsky, Elizabeth M. Sagatys, Said M. Sebti, Gary W. Reuther, Gloria C. Ferreira, and Rays H.Y. Jiang. 2023. Porphyrin overdrive rewires pan-cancer cell metabolism. *bioRxiv*.

Monica Agrawal, Stefan Hegselmann, Hunter Lang, Yoon Kim, and David Sontag. 2022. Large language models are few-shot clinical information extractors. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 1998–2022, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Rohan Anil, Andrew M. Dai, Orhan Firat, Melvin Johnson, Dmitry Lepikhin, Alexandre Passos, Siamak Shakeri, Emanuel Taropa, Paige Bailey, Zhifeng Chen, Eric Chu, Jonathan H. Clark, Laurent El Shafey, Yanping Huang, Kathy Meier-Hellstern, Gaurav Mishra, Erica Moreira, Mark Omernick, Kevin Robinson, Sebastian Ruder, Yi Tay, Kefan Xiao, Yuanzhong Xu, Yujing Zhang, Gustavo Hernandez Abrego, Junwhan Ahn, Jacob Austin, Paul Barham, Jan Botha, James Bradbury, Siddhartha Brahma, Kevin Brooks, Michele Catasta, Yong Cheng, Colin Cherry, Christopher A. Choquette-Choo, Aakanksha Chowdhery, Clément Crepy, Shachi Dave, Mostafa Dehghani, Sunipa Dev, Jacob Devlin, Mark Díaz, Nan Du, Ethan Dyer, Vlad Feinberg, Fangxiaoyu Feng, Vlad Fienber, Markus Freitag, Xavier Garcia, Sebastian Gehrmann, Lucas Gonzalez, Guy Gur-Ari, Steven Hand, Hadi Hashemi, Le Hou, Joshua Howland, Andrea Hu, Jeffrey Hui, Jeremy Hurwitz, Michael Isard, Abe Ittycheriah, Matthew Jagielski, Wenhao Jia, Kathleen Kenealy, Maxim Krikun, Sneha Kudugunta, Chang Lan, Katherine Lee, Benjamin Lee, Eric Li, Music Li, Wei Li, YaGuang Li, Jian Li, Hyeontaek Lim, Hanzhao Lin, Zhongtao Liu, Frederick Liu, Marcello Maggioni, Aroma Mahendru, Joshua Maynez, Vedant Misra, Maysam Moussalem, Zachary Nado, John Nham, Eric Ni, Andrew Nystrom, Alicia Parrish, Marie Pellat, Martin Polacek, Alex Polozov, Reiner Pope, Siyuan Qiao, Emily Reif, Bryan Richter, Parker Riley, Alex Castro Ros, Aurko Roy, Brennan Saeta, Rajkumar Samuel, Renee Shelby, Ambrose Slone, Daniel Smilkov, David R. So, Daniel Sohn, Simon Tokumine, Dasha Valter, Vijay Vasudevan, Kiran Vodrahalli, Xuezhi Wang, Pidong Wang, Zirui Wang, Tao Wang, John Wieting, Yuhuai Wu, Kelvin Xu, Yunhan Xu, Linting Xue, Pengcheng Yin, Jiahui Yu, Qiao Zhang, Steven Zheng, Ce Zheng, Weikang Zhou, Denny Zhou, Slav Petrov, and Yonghui Wu. 2023. Palm 2 technical report.

Amanda Bertsch, Uri Alon, Graham Neubig, and Matthew R Gormley. 2023. Unlimiformer: Long-range transformers with unlimited length input. *arXiv preprint arXiv:2305.01625*.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, Red Hook, NY, USA. Curran Associates Inc.

Cambridge Open Engage. 2023. Chemrxiv: The preprint server for chemistry. https://chemrxiv.org. Accessed: 2023-12-15.

Chemical Abstracts Service (CAS). 2023. Scifinder. https://scifinder.cas.org. Accessed: 2024-01-15.

Ian W. Davies. 2019. The digitization of organic synthesis. *Nature*, 570(7760):175–181.

Alexander Dunn, John Dagdelen, Nicholas Walker, Sanghoon Lee, Andrew S. Rosen, Gerbrand Ceder, Kristin Persson, and Anubhav Jain. 2022. Structured information extraction from complex scientific text with fine-tuned large language models.

Elsevier B.V. 2023. Reaxys. https://www.reaxys.com. Accessed: 2024-01-15.

Biaoyan Fang, Christian Druckenbrodt, Saber A Akhondi, Jiayuan He, Timothy Baldwin, and Karin Verspoor. 2021. ChEMU-ref: A corpus for modeling anaphora resolution in the chemical domain. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1362–1375, Online. Association for Computational Linguistics.

Jiang Guo, A. Santiago Ibanez-Lopez, Hanyu Gao, Victor Quach, Connor W. Coley, Klavs F. Jensen, and Regina Barzilay. 2022. Automated chemical reaction extraction from scientific literature. *Journal of Chemical Information and Modeling*, 62(9):2035–2045. PMID: 34115937.

M. Hattori, Y. Okuno, S. Goto, and M. Kanehisa. 2003. Heuristics for chemical compound matching. *Genome Inform*, 14:144–153.

Lezan Hawizy, David M. Jessop, Nico Adams, and Peter Murray-Rust. 2011. Chemicaltagger: A tool for semantic text-mining in chemistry. *Journal of Cheminformatics*, 3(1):17.

Jiayuan He, Dat Quoc Nguyen, Saber A. Akhondi, Christian Druckenbrodt, Camilo Thorne, Ralph Hoessel, Zubair Afzal, Zenan Zhai, Biaoyan Fang, Hiyori Yoshikawa, Ameer Albahem, Lawrence Cavedon, Trevor Cohn, Timothy Baldwin, and Karin Verspoor. 2020. Overview of chemu 2020: Named entity recognition and event extraction of chemical reactions from patents. In *Experimental IR Meets Multilinguality, Multimodality, and Interaction*, pages 237–254, Cham. Springer International Publishing.

Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Ceyao Zhang, Jinlin Wang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, Chenyu Ran, Lingfeng Xiao, Chenglin Wu, and Jürgen Schmidhuber. 2023. Metagpt: Meta programming for a multi-agent collaborative framework.

Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, and Ting Liu. 2023. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions.

Tuan M. Lai, Chengxiang Zhai, and Heng Ji. 2023. Knowledge-enhanced biomedical language models. In *Journal of Biomedical Informatics*.

Vladimir I Levenshtein et al. 1966. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, volume 10, pages 707–710. Soviet Union.

NextMove Software. 2023. Pistachio: Structure to name. https://www.nextmovesoftware.com/pistachio.html. Accessed: 2024-01-15.

OpenAI. 2023. Gpt-4 technical report.

Siru Ouyang, Zhuosheng Zhang, Bing Yan, Xuan Liu, Yejin Choi, Jiawei Han, and Lianhui Qin. 2024. Structured chemistry reasoning with large language models.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, page 311–318, USA. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer.

Amir Soleimani, Vassilina Nikoulina, Benoit Favre, and Salah Ait Mokhtar. 2022. Zero-shot aspect-based scientific document summarization using self-supervised pre-training. In *Proceedings of the 21st Workshop on Biomedical Language Processing*, pages 49–62, Dublin, Ireland. Association for Computational Linguistics.

Yu Song, Santiago Miret, and Bang Liu. 2023. 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, Toronto, Canada. Association for Computational Linguistics.

Pontus Stenetorp, Sampo Pyysalo, Goran Topić, Tomoko Ohta, Sophia Ananiadou, and Jun'ichi Tsujii. 2012. brat: a web-based tool for NLP-assisted text annotation. In *Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 102–107, Avignon, France. Association for Computational Linguistics.

Matthew C. Swain and Jacqueline M. Cole. 2016. Chemdataextractor: A toolkit for automated extraction of chemical information from the scientific literature. *Journal of Chemical Information and Modeling*, 56(10):1894–1904. PMID: 27669338.

Ross Taylor, Marcin Kardas, Guillem Cucurull, Thomas Scialom, Anthony Hartshorn, Elvis Saravia, Andrew Poulton, Viktor Kerkez, and Robert Stojnic. 2022. Galactica: A large language model for science.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models.

Francis Tyers, Robert Pugh, and Valery Berthoud F. 2023. Codex to corpus: Exploring annotation and processing for an open and extensible machine-readable edition of the florentine codex. In *Proceedings of the Workshop on Natural Language Processing for Indigenous Languages of the Americas (AmericasNLP)*, pages 19–29, Toronto, Canada. Association for Computational Linguistics.

Alain C. Vaucher, Philippe Schwaller, Joppe Geluykens, Vishnu H. Nair, Anna Iuliano, and Teodoro Laino. 2021. Inferring experimental procedures from text-based representations of chemical reactions. *Nature Communications*, 12(1):2573.

Alain C. Vaucher, Federico Zipoli, Joppe Geluykens, Vishnu H. Nair, Philippe Schwaller, and Teodoro Laino. 2020a. Automated extraction of chemical synthesis actions from experimental procedures. *Nature Communications*, 11(1):3601.

Alain C. Vaucher, Federico Zipoli, Joppe Geluykens, Vishnu H. Nair, Philippe Schwaller, and Teodoro Laino. 2020b. Automated extraction of chemical synthesis actions from experimental procedures. *Nature Communications*, 11.

Sheng Wang, Yuzhi Guo, Yuhong Wang, Hongmao Sun, and Junzhou Huang. 2019. Smiles-bert: Large scale unsupervised pre-training for molecular property prediction. In *Proceedings of the 10th ACM International Conference on Bioinformatics, Computational Biology and Health Informatics*, BCB '19, page 429–436, New York, NY, USA. Association for Computing Machinery.

Xingyao Wang, Yangyi Chen, Lifan Yuan, Yizhe Zhang, Yunzhu Li, Hao Peng, and Heng Ji. 2024. Executable code actions elicit better llm agents. In *arxiv*.

Xingyao Wang, Sha Li, and Heng Ji. 2022a. Code4struct: Code generation for few-shot structured prediction from natural language. *arXiv preprint arXiv:2210.12810*.

Zheren Wang, Olga Kononova, Kevin Cruse, Tanjin He, Haoyan Huo, Yuxing Fei, Yan Zeng, Yingzhi Sun, Zijian Cai, Wenhao Sun, and Gerbrand Ceder. 2022b. Dataset of solution-based inorganic materials synthesis procedures extracted from the scientific literature. *Scientific Data*, 9.

Ken Watanabe, Masayoshi Yuasa, Kida Tetsuya, Yasutake Teraoka, Noboru Yamazoe, and Kengo Shimanoe. 2010. High-performance oxygen-permeable membranes with an asymmetric structure using ba0.95la0.05feo3 perovskite-type oxide. *Advanced materials (Deerfield Beach, Fla.)*, 22:2367–70.

Zheni Zeng, Yi-Chen Nie, Ning Ding, Qian-Jun Ding, Wei-Ting Ye, Cheng Yang, Maosong Sun, Weinan E, Rong Zhu, and Zhiyuan Liu. 2023. Transcription between human-readable synthetic descriptions and machine-executable instructions: an application of the latest pre-training technology. *Chem. Sci.*, 14:9360–9373.

Ming Zhong, Siru Ouyang, Minhao Jiang, Vivian Hu, Yizhu Jiao, Xuan Wang, and Jiawei Han. 2023a. ReactIE: Enhancing chemical reaction extraction with weak supervision. In *Findings of the Association for*

Computational Linguistics: ACL 2023, pages 12120–12130, Toronto, Canada. Association for Computational Linguistics.

Ming Zhong, Siru Ouyang, Yizhu Jiao, Priyanka Kargupta, Leo Luo, Yanzhen Shen, Bobby Zhou, Xianrui Zhong, Xuan Liu, Hongxiang Li, Jinfeng Xiao, Minhao Jiang, Vivian Hu, Xuan Wang, Heng Ji, Martin Burke, Huimin Zhao, and Jiawei Han. 2023b. Reaction miner: An integrated system for chemical reaction extraction from textual data. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 389–402, Singapore. Association for Computational Linguistics.

Andy Zhou, Kai Yan, Michal Shlapentokh-Rothman, Haohan Wang, and Yu-Xiong Wang. 2023. Language agent tree search unifies reasoning acting and planning in language models.

A Action Type

We adopt the same action types as in the previous study, including 26 pre-defined action types. We include the detailed descriptions of action types in (Vaucher et al., 2020a) as a reference in Table 4 to help readers better understand the action types.

B Dataset statistics

The number of each action type mentioned in all 352 samples in PatentAction dataset are summarized in Table 5.

C Prompt

Figure 6 demonstrates the prompt for text rephrasing. Figure 7 represents the prompt for code generation.

D Case Study on Different Scientific Domains

In order to validate the effectiveness of our method on different scientific domains, we have conducted two additional case studies, one in biology and one in material science, to highlight the potential for future studies to extend ActionIE across different domains. The case study in biology can be found at Figure 8, and the case study in material science can be found at Figure 9.

You are an expert in chemistry.

Rephrase the paragraph if you think it is difficult for general readers to understand. Keep the structure of the text as much as possible. Use the provided patterns when it is possible.

Here is the paragraph: [Input Text]

Here is the patterns your output should utilize: [Enriched Patterns]

Figure 6: Prompt for Text Rephrasing.

You are an expert in chemistry, programming, and extracting information.

The following python script describes chemical reaction procedure actions.

```python
[User Defined Python Class File]
```
Extract chemical reaction procedure actions from the following text: [Input Text]

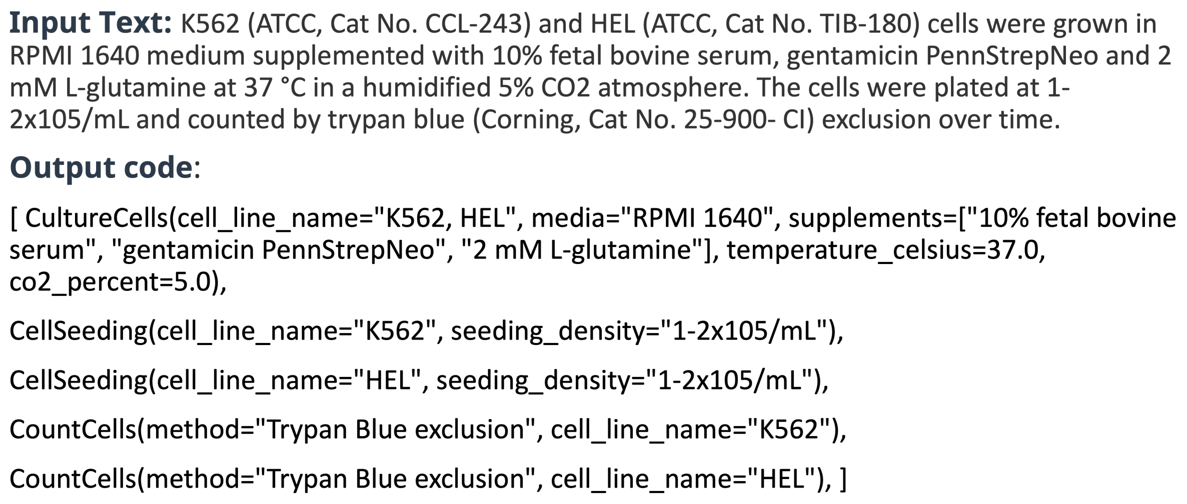
To clearly explain the task, we provide the following example:
[Demonstrations]

Following the above examples, complete the following code:
```python
procedure =
```
Remember to strictly follow the output format.

Figure 7: Prompt for Code Generation.

Action Type	Description
Add	Add a substance to the reactor
CollectLayer	Select aqueous or organic fraction(s)
Concentrate	Evaporate the solvent (rotavap)
Degas	Purge the reaction mixture with a gas
DrySolid	Dry a solid
DrySolution	Dry an organic solution with a desiccant
Extract	Transfer compound into a different solvent
Filter	Separate solid and liquid phases
MakeSolution	Mix several substances to generate a mixture or solution
Microwave	Heat the reaction mixture in a microwave apparatus
Partition	Add two immiscible solvents for subsequent phase separation
PH	Change the pH of the reaction mixture
PhaseSeparation	Separate the aqueous and organic phases
Purify	Purification
Quench	Stop reaction by adding a substance
Recrystallize	Recrystallize a solid from a solvent or mixture of solvents
Reflux	Reflux the reaction mixture
SetTemperature	Change the temperature of the reaction mixture
Sonicate	Agitate the solution with sound waves
Stir	Stir the reaction mixture for a specified duration
Triturate	Triturate the residue
Wait	Leave the reaction mixture to stand for a specified duration
Wash	Wash (after filtration, or with immiscible solvent)
Yield	Phony action, indicates the product of a reaction
FollowOtherProcedure	The text refers to a procedure described elsewhere
InvalidAction	Unknown or unsupported action
OtherLanguage	The text is not written in English
NoAction	The text does not correspond to an actual action

Table 4: Pre-defined action types used in this paper.



Input Text: K562 (ATCC, Cat No. CCL-243) and HEL (ATCC, Cat No. TIB-180) cells were grown in RPMI 1640 medium supplemented with 10% fetal bovine serum, gentamicin PennStrepNeo and 2 mM L-glutamine at 37 °C in a humidified 5% CO2 atmosphere. The cells were plated at 1-2x105/mL and counted by trypan blue (Corning, Cat No. 25-900- CI) exclusion over time.

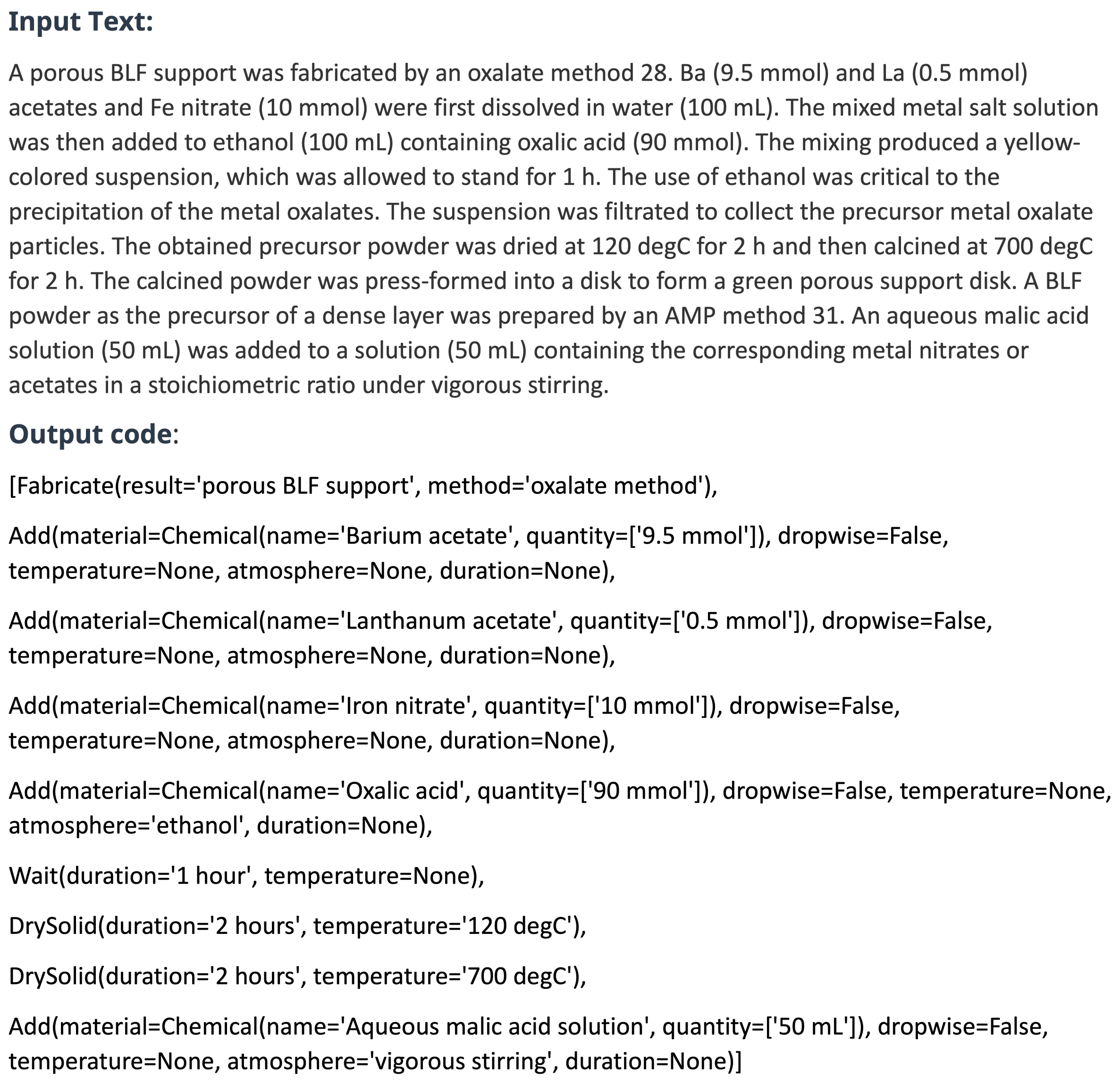
Output code:

```
[ CultureCells(cell_line_name="K562, HEL", media="RPMI 1640", supplements=["10% fetal bovine serum", "gentamicin PennStrepNeo", "2 mM L-glutamine"], temperature_celsius=37.0, co2_percent=5.0),

CellSeeding(cell_line_name="K562", seeding_density="1-2x105/mL"),

CellSeeding(cell_line_name="HEL", seeding_density="1-2x105/mL"),

CountCells(method="Trypan Blue exclusion", cell_line_name="K562"),

CountCells(method="Trypan Blue exclusion", cell_line_name="HEL"), ]
```

Figure 8: Case study on biology literature. The text is collected from (Adapa et al., 2023).

Action Type	**Total number of occurences**
Add	255
CollectLayer	37
Concentrate	54
Degas	1
DrySolid	12
DrySolution	22
Extract	34
Filter	34
MakeSolution	62
Microwave	0
Partition	5
PH	47
PhaseSeparation	4
Purify	24
Quench	8
Recrystallize	2
Reflux	7
SetTemperature	60
Sonicate	0
Stir	118
Triturate	3
Wait	19
Wash	45
Yield	37
FollowOtherProcedure	15
InvalidAction	11
OtherLanguage	2
NoAction	25

Table 5: PatentAction Dataset statistics.



Input Text:

A porous BLF support was fabricated by an oxalate method 28. Ba (9.5 mmol) and La (0.5 mmol) acetates and Fe nitrate (10 mmol) were first dissolved in water (100 mL). The mixed metal salt solution was then added to ethanol (100 mL) containing oxalic acid (90 mmol). The mixing produced a yellow-colored suspension, which was allowed to stand for 1 h. The use of ethanol was critical to the precipitation of the metal oxalates. The suspension was filtrated to collect the precursor metal oxalate particles. The obtained precursor powder was dried at 120 degC for 2 h and then calcined at 700 degC for 2 h. The calcined powder was press-formed into a disk to form a green porous support disk. A BLF powder as the precursor of a dense layer was prepared by an AMP method 31. An aqueous malic acid solution (50 mL) was added to a solution (50 mL) containing the corresponding metal nitrates or acetates in a stoichiometric ratio under vigorous stirring.

Output code:

```
[Fabricate(result='porous BLF support', method='oxalate method'),

Add(material=Chemical(name='Barium acetate', quantity=['9.5 mmol']), dropwise=False,
temperature=None, atmosphere=None, duration=None),

Add(material=Chemical(name='Lanthanum acetate', quantity=['0.5 mmol']), dropwise=False,
temperature=None, atmosphere=None, duration=None),

Add(material=Chemical(name='Iron nitrate', quantity=['10 mmol']), dropwise=False,
temperature=None, atmosphere=None, duration=None),

Add(material=Chemical(name='Oxalic acid', quantity=['90 mmol']), dropwise=False, temperature=None,
atmosphere='ethanol', duration=None),

Wait(duration='1 hour', temperature=None),

DrySolid(duration='2 hours', temperature='120 degC'),

DrySolid(duration='2 hours', temperature='700 degC'),

Add(material=Chemical(name='Aqueous malic acid solution', quantity=['50 mL']), dropwise=False,
temperature=None, atmosphere='vigorous stirring', duration=None)]
```

Figure 9: Case study on material science literature. The text is collected from (Watanabe et al., 2010).