

Decompose, Enrich, and Extract! Schema-aware Event Extraction using LLMs.

Fatemeh Shiri, Farhad Moghimifar,
Reza Haffari, Yuan-Fang Li
Department of Data Science and AI
Faculty of Information Technology
Monash University, Melbourne, Australia
firstname.lastname@monash.edu

Van Nguyen, John Yoo
Information Sciences Division
Defence Science and Technology Group
Adelaide, Australia
firstname.lastname@defence.gov.au

Abstract—Large Language Models (LLMs) demonstrate significant capabilities in processing natural language data, promising efficient knowledge extraction from diverse textual sources to enhance situational awareness and support decision-making. However, concerns arise due to their susceptibility to hallucination, resulting in contextually inaccurate content. This work focuses on harnessing LLMs for automated Event Extraction, introducing a new method to address hallucination by decomposing the task into Event Detection and Event Argument Extraction. Moreover, the proposed method integrates dynamic schema-aware augmented retrieval examples into prompts tailored for each specific inquiry, thereby extending and adapting advanced prompting techniques such as Retrieval-Augmented Generation. Evaluation findings on prominent event extraction benchmarks and results from a synthesized benchmark illustrate the method’s superior performance compared to baseline approaches.

Index Terms—information extraction, event extraction, event argument extraction

I. INTRODUCTION

Enhancing strategic awareness for effective military decision-making requires processing and analysing information from various data sources. One category of data sources of particular interest in this work is textual data available in vast quantities in the public domain, such as online news articles and reports. However, handling such data sources poses significant challenges, including issues of volume and relevance [1]. Manual processing of the large volume and complexity of open-source data leads to cognitive overload. However, automated extraction of knowledge using traditional Natural Language Processing (NLP) techniques is also a challenging task.

Recent advances in Artificial Intelligence, especially with the emergence of Large Language Models (LLMs), have demonstrated an impressive ability to process and comprehend natural language at a sophisticated level [2]. This capability enables them to execute tasks like language generation, translation, summarisation, and beyond. Consequently, they offer promising potential in efficiently processing and analysing substantial volumes of data, crucial for augmenting human decision-making and military planning [3]. Specifically, LLMs can be harnessed to automatically extract structured knowledge, such as in the form of event extraction (EE). This not only aids human analysts in visualising and describing complex situations in the form of a knowledge graph, but also

facilitates automated downstream reasoning and learning for generating actionable insights.

Despite numerous remarkable breakthroughs, the hallucination problem associated with LLMs raises significant concerns, particularly in adopting these technologies in the defense context [4], [5]. Hallucination refers to generating content that strays from factual reality or includes fabricated information. This work is focused on leveraging LLMs for automating event extraction, toward supporting the creation of knowledge graphs and enhancing decision-making processes. To enhance the relevance and accuracy of the outcomes, we propose a new method to mitigate the hallucination problem. This is achieved by adapting state-of-the-art prompting approaches, such as Chain-of-Thought [6] and Retrieval Augmented Generation [7].

EE is a challenging NLP task that requires the identification and extraction of structured event records from unstructured text. It typically includes a trigger indicating the occurrence of the event and multiple arguments of pre-defined roles. As mentioned earlier, EE plays a crucial role as it provides valuable information for various downstream tasks, including knowledge graph construction [8] and question answering [9].

State-of-the-art generative event extraction approaches rely on fine-tuning sequence-to-sequence models [10], [11]. These models require large-scale annotated data which is expensive and time-consuming to obtain. The emergence of LLMs such as ChatGPT and GPT-4 [2] provides an opportunity to solve language tasks using in-context learning (ICL) without the need for task-specific datasets and fine-tuning [12]. ICL leverages the concept of “learning by demonstration,” a method where the pre-trained model is able to recognize and replicate tasks by giving it a few examples. This phenomenon allows the model to mimic task-specific behavior by following the instructions and adjusting its generated responses to match the demonstrated examples. By providing a well-devised prompt, LLMs can therefore learn to perform EE tasks. However, the challenge remains as to how to devise an appropriate input prompt to be able to extract complex structured events while minimizing the risk of hallucination.

To mitigate hallucinations generated by LLMs, we decompose the EE task into two steps: Event Detection (ED) and Event Argument Extraction (EAE). We utilize schema-aware prompts, including granular instructions and the most relevant

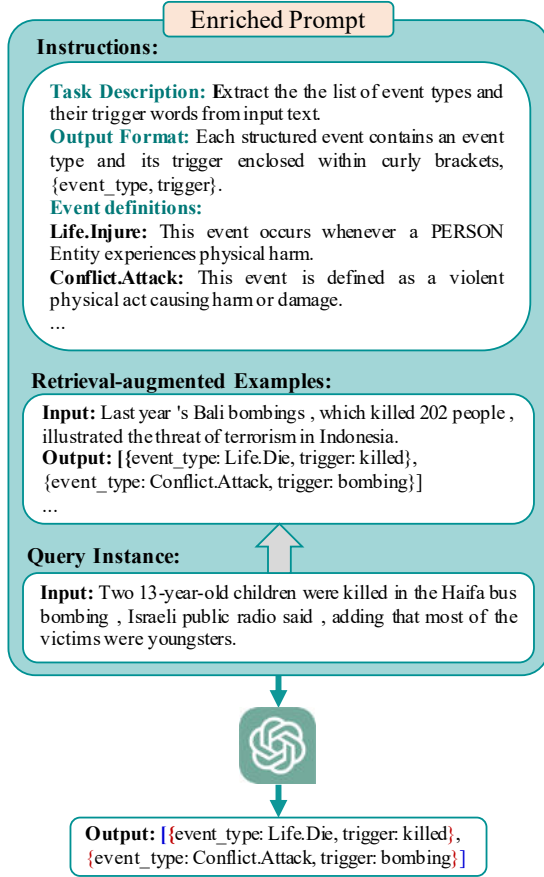


Fig. 1. An example of enriched prompt for event extraction using GPT-4. GPT-4 is tasked with query instances based on provided instructions, event type definitions, output format, and retrieval-augmented examples in this scenario. These examples are the most similar instances to the query instance retrieved from the existing knowledge base. GPT-4 is expected to produce responses for each query instance without any prior training on the specific task or data. (For simplicity, we only show the first step, Event Detection).

demonstration examples, called retrieval-augmented examples, tailored to the query instance.

Figure 1 displays an example of event detection performed by the GPT-4 model using a retrieval-augmented prompt. The input instance and the target events are both taken from the ACE2005 dataset [13].

In summary, our contributions are as follows:

- We construct automatic retrieval-augmented prompts for event detection and argument extraction sub-tasks on both high-resource and low-resource situations.
- Instead of relying only on task description and output options, our enriched prompt integrates extraction rules, specifies output formats, and provides retrieval-augmented examples to provide LLMs with an in-depth understanding of the structured extraction tasks.
- We synthesize a new event benchmark called MaritimeEvent with 10,000 data points. We evaluate our proposed framework on popular EE benchmarks and our MaritimeEvent benchmark. The results demonstrate its advantages over baselines.
- Through detailed analysis and case study, we demonstrate

the essential role of retrieval augmentation in boosting EE performance.

II. BACKGROUND

Our approach is related to two lines of research.

A. Generative Event Extraction

Traditional event extraction methods required fine-grained (token-level and entity-level) annotations [14], [15], [16]. Recent generative approaches tackle this problem as a sequence-to-sequence learning problem referred to as End-to-End event extraction [10], [17]. TANL [18] and GRIT [19] employ neural generation models for event extraction by focusing on sequence generation. While TEXT2EVENT [10] and KC-GEE [20] focus on structure generation, TEXT2EVENT directly generates event schema and text spans to form event records via constrained decoding.

B. Event Extraction with LLMs

LLMs are central to NLP due to their impressive performance on numerous tasks [21]–[23]. LLMs are inherently context-sensitive, they may produce different responses based on the specific formulation of the prompt. By carefully crafting and refining prompts, the capabilities of LLM can be fully utilized, tailoring the outputs to match the nuances of diverse tasks and objectives.

Chain-of-Thought (CoT) [6] is an enhanced prompting approach designed to enhance the proficiency of LLMs in handling intricate reasoning tasks, including arithmetic reasoning [24], commonsense reasoning [6].

Despite the potential benefits of LLMs in context learning (ICL), existing literature has limitations in evaluating these models' efficacy for this task. The most relevant to our research is leveraging ChatGPT for the information extraction task [25], [26], including event detection [27], and event argument extraction [28], [29]. These papers mainly focus on either curating new benchmark datasets [30] or benchmarking ChatGPT's performance with simple instructions without detailed extraction guidelines. The latter reports results that are inferior to those achieved by specialized supervised EE systems [25], [26].

Instead of using a single instruction to extract multiple information types, [30] propose fine-grained IE prompting for each information type extraction. For example, in the case of entity extraction, they manually craft fine-grained instructions for each event type and argument in the dataset. However, this approach demands substantial effort and results in lengthy context prompts for LLMs. In contrast, we propose a prompting strategy that includes granular instructions along with automatic retrieval augmented examples which are vital for adapting to EE tasks.

III. APPROACH

We propose an end-to-end framework that extracts the structured events records from input text in two steps using LLMs. Figure 2 illustrates a high-level overview of our approach. Given the query instance, our system generates the embedding

representation of the query instance and uses Facebook AI Similarity Search (FAISS) [31] to prioritize and retrieve the top K instances from the training dataset that demonstrate the highest similarity to the query. These retrieved examples, combined with the granular instructions in both the ED prompt and EAE prompt, are fed into the LLMs for event extraction. In the following, we explain each component of our framework in detail.

A. Task Decomposition

LLMs have advanced text input capabilities, yet encounter challenges in retrieving accurate information from lengthy contexts [32], [33]. The “lost in the middle” issue significantly impacts their accuracy, especially when accurate information is positioned within the middle of the text. Extracting structured events from text, especially from documents, necessitates providing task instructions, schema, extraction rules, event and argument definitions, output format, and demonstration examples. However, this lengthy prompt may challenge LLMs, potentially leading to inaccuracies in event extraction.

Decomposed prompting was proposed to solve complex tasks by decomposing tasks (via prompting) into simpler sub-tasks [34]. Each sub-task is tackled in sequence, with the outcome appended before progressing to the next [35]. Inspired by this research direction, we address the challenge of long-context comprehension in LLMs by breaking down Event Extraction (EE) into two sub-tasks: ED which focuses on identifying event triggers and event types, and EAE which deals with extracting event arguments for predefined roles. For each sub-task, we design specific prompts, along with demonstration examples. Figures 3 and 4 showcase two examples of our enriched prompts for the ED and EAE sub-tasks, respectively. We examine the efficacy of this decomposed prompting approach within the context of EE in Section VI-1.

B. Granular Instructions

An essential function of LLMs is their ability to execute natural language instructions, often referred to as zero-shot prompts [36]. The capability of LLMs to precisely understand and execute natural language instructions, especially structured information extraction is vital, ensuring both the accuracy of the outcomes and the reliability of their executions. Ambiguous or inadequate instructions can result in unintended outcomes, potentially leading to serious consequences. Therefore, it is crucial to offer granular and comprehensive instructions when interacting with an LLM.

In order to extract accurate structured events from text, we provide enriched prompts including task description, schema, extraction rules, event types and argument definitions, output format, and demonstration examples for each sub-task. Figures 3 and 4 show detailed instructions of both ED and EAE sub-tasks.

C. Retrieval Augmented Examples (RAE) for Prompt Enriching

A significant benefit of ICL, in contrast to traditional fine-tuning methods, is its decreased dependency on extensive

amounts of annotated data. A straightforward yet potent implementation of ICL is through one-shot and few-shot prompting. These methods employ standardized examples that remain consistent across all input instances, irrespective of their relevance. However, this may lead to a lack of flexibility in understanding diverse contexts and specific input variations, resulting in suboptimal performance.

We address this problem by dynamically adapting and refining LLMs’ understanding based on the specific context of each input instance. While ICL does not necessitate a large scale labeled dataset as in fine-tuning methods, the inclusion of a retrieval mechanism allows such external data, if available, to serve as a valuable resource. This facilitates the discovery of contextually relevant examples for input instances, thereby enhancing the performance of ICL. The retrieval of examples from the existing labeled data for the EE task can be facilitated through several search mechanisms. The choice of a specific example retriever is non-trivial, as it significantly impacts the retrieved examples and their subsequent contribution to task execution. Drawing inspiration from Retrieval Augmented Generation (RAG) [7], which incorporates the retrieval of external data into the generative process, we retrieve instances from the existing set corresponding to each query. Our approach mirrors the conventional process of indexing, retrieval, and generation, aligning with the methodology employed in RAG.

The first step in our approach is to transform existing instances into vector representations through an embedding model. We employ three different embedding techniques discussed in section V. Next, the encoded instance vectors are indexed using the *IndexFlatL2* structure provided by FAISS.¹ Upon receipt of a user query, the system employs the same encoding model utilized during the indexing phase to transform the input query into a vector representation. It then proceeds to compute the similarity scores between the query vector and the vectorized existing instances within the indexed corpus. The system prioritizes and retrieves the top K instances that demonstrate the highest similarity to the query. Finally, the retrieved context combined with the prompt is fed into the LLMs for event extraction.

IV. EXPERIMENTS

A. Experimental Setup

Dataset. We carry out experiments on two popular event extraction datasets: the sentence-level dataset Automatic Content Extraction 2005 (ACE05-EN) [13], and the document-level dataset: WIKIEVENTS [37]. It is worth noting that WIKIEVENTS presents significant challenges due to three factors. (1) Each instance in ACE05-EN contains only one sentence, whereas instances in WIKIEVENTS are documents. (2) Almost every instance in ACE05-EN contains only one event, whereas multiple events could be present in one instance in WIKIEVENTS. (3) The amount of training data in ACE05-EN is more than 77 times greater than that in WIKIEVENTS.

¹FAISS is an open-source library for dense vector clustering and similarity search.

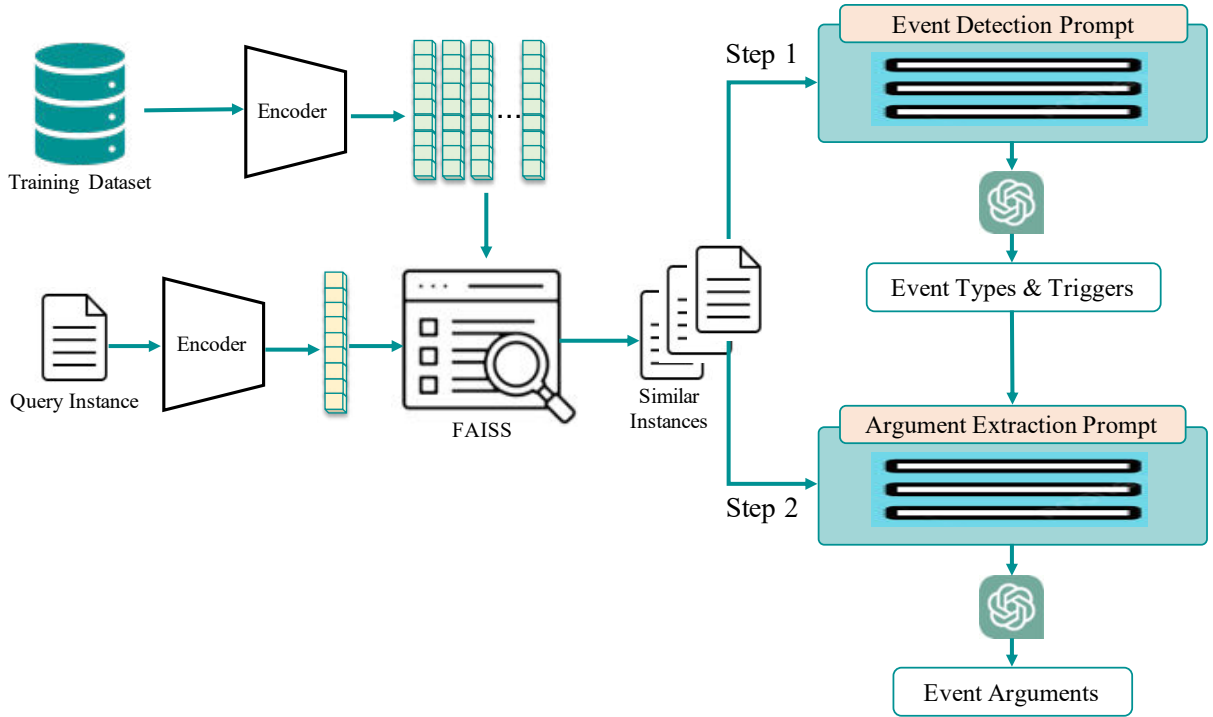


Fig. 2. An illustration of our end-to-end framework for event extraction, which performs event detection and event argument extraction jointly. The details of the Event Detection Prompt and Argument Extraction Prompt are shown in Figures 3 and 4.

Additionally, to evaluate our method in a new domain, we synthesized a new event dataset called *MaritimeEvent*. We collected 100 maritime-related reports containing multiple sentences and single or multiple events. We manually designed the *Maritime* schema, which contains 16 different event types, *depart*, *arrive*, *monitor*, *kidnap*, *robbery*, *escort*, *block*, *detach*, *commence*, *transmit*, *cease*, *alter_course*, *detect*, *pass*, *transit*, and *hail*, and 6 argument types, *date*, *time*, *location*, *shiptype*, *shipname* and *destination*. Figure 6 illustrates two maritime samples. We use ChatGPT (gpt-3.5-turbo via OpenAI API) to synthesize approximately 10,000 diverse maritime reports. The prompt template for synthesizing a report is shown in Figure 5. To generate each report, we begin by randomly selecting a subset of samples from the 100 available demonstration examples as seed examples. Additionally, we choose one or a few event types, along with their corresponding arguments, from the provided lists to replace the placeholders in the prompt template. For the experiments, the temperature of ChatGPT (gpt-3.5-turbo) is set to 1.6 to synthesize maritime reports with relatively high diversity. We set the maximum number of tokens to 4000. Figure 6 illustrates an original maritime report and a synthesized one from the *MaritimeEvent* dataset. The statistics of all datasets are provided in Table I.

Evaluation Metrics. The results are reported using F-measure (F-1) score metrics for both event detection (Trig-C) and argument extraction (Arg-C). Trig-C indicates trigger identification and event type classification. Arg-C indicates argument identification and role classification.

Baselines. To validate the proposed approach through exper-

TABLE I
STATISTICS OF THE EVENT EXTRACTION DATASETS USED IN THE PAPER, INCLUDING THE TYPE OF INSTANCES, EVENTS PER INSTANCE, AND THE NUMBER OF INSTANCES IN DIFFERENT SPLITS.

Dataset	Train	Dev	Test	Instance	#Events
ACE05-EN	17,172	923	832	One sentence	Single/Multiple
WIKIEVENTS	206	20	20	Document	Multiple
MaritimeEvent	9851	8851	1000	Multi-sentence	Multiple

imental comparison, we selected the following event extraction models as the baselines:

- **Text2Event** [38]: We compare our method with Text2Event, which is a framework that utilizes T5 models [39] to approach event extraction by framing it as a sequence-to-sequence generation task. In this method, all triggers, arguments, and their corresponding labels are generated as natural language words.
- **OntoGPT** [40] is a tool for extracting knowledge from text by recursively querying an LLM such as GPT-3 to obtain responses using zero-shot learning. OntoGPT applies a knowledge schema on the input text and returns information conforming to the knowledge schema. To utilize this tool for EE, we manually created the LinkML [41] schema for the datasets for the evaluation of the OntoGPT model.
- **Fine-Grained IE** [30]: We compare our method with the Fine-Grained IE approach that benchmarks LLMs for Information Extraction (IE) by incorporating detailed instructions and examples for each information type.

Event Detection Prompt

Task Description: You are an assistant that helps extract the list of event types and their trigger words from input text.

Extraction Rules:

- Each instance should contain a minimum of one event and a maximum of four.
- Limit responses to event types and their triggers only.
- Refrain from providing additional explanations.
- Do not enumerate the list.

Event Type Definitions. The possible event types and their definitions are as follows:

Movement.Transport: This event occurs whenever an ARTIFACT (weapon or vehicle) or a person is moved from one place to another.

Life.Injure: This event occurs whenever a PERSON Entity experiences physical harm.

Conflict.Attack: This event is defined as a violent physical act causing harm or damage.

...

Output Format: Each event contains an event type and its trigger enclosed within curly brackets, {event_type, trigger}.

Retrieval-augmented Examples:

Report: <report 1>

Events: [<event 1>]

.

.

Report: <report 5>

Events: [<event 5>]

Examples end here.

Fig. 3. An illustration of our granular Instructions and placeholder for retrieval augmented examples within the ED prompt

Argument Extraction Prompt

Task Description: You are an assistant that helps extract the list of arguments for each event from input text.

Detected Events: [{event_type: Conflict.Attack, trigger: bombing}, {event_type: Life.Die, trigger: killed}]

Extraction Rules:

- Each instance should contain a minimum of zero argument and a maximum of 6 arguments.
- Limit responses to arguments of detected event types only.
- Refrain from providing additional explanations.
- Do not enumerate the list.

Argument Definitions. The possible event types and their arguments are as follows:

Life.Die: This event has four arguments (Agent, Victim, Instrument, Place) Agent: The attacking agent / The killer Victim: The person(s) who died, Instrument: The device used to kill Place: Where the death takes place

Conflict.Attack: This event has four arguments (Attacker, Target, Instrument, Place). Attacker: The attacking/instigating agent, Target: The target of the attack, Instrument: The instrument used in the attack, Place: Where the attack takes place

Output Format: The output should be a list of arguments for each event type enclosed within curly brackets, {event_type, arguments: [Arg1, Arg2, Arg3,...]}.

Retrieval-augmented Examples:

Report: <report 1>

Events: [<event 1, arguments 1>]

.

.

Report: <report 5>

Events: [<event 5>, arguments 5>]

Examples end here.

Fig. 4. An illustration of our granular Instructions and placeholder for retrieval augmented examples within the EAE prompt

V. EVALUATION

This work aims to improve the automated extraction of events from text by leveraging LLMs. First, we utilize the global knowledge of LLMs through zero-shot prompts with minimal context for EE. Secondly, we evaluate LLMs in a few-shot settings in which we add a few demonstration examples to guide LLMs to extract more accurate structured events from text. Finally, we show that when combined with local knowledge through dynamic retrieval augmented examples (the most related examples to each query text), LLMs are effective in the field of EE. Table II shows the impact of different types/number of demonstration examples within the prompt for both ED and EAE steps on three datasets.

In zero-shot prompting, the LLM is provided with direct instruction without any additional examples. LLMs show a clear ability to learn from the context within prompts, underscoring the importance of incorporating relevant examples. In few-shot prompting we extend the number of examples to one and five canonical examples, aiming to provide additional context for the model to learn from. When applying retrieval augmented examples, both the F1 score of DE and EAE improves significantly across all LLMs, observed consistently in both ACE05-EN and MaritimeEvent datasets, characterized by a large number of instances in the training set. However, the gap remains small for the WikiEvent dataset due to a very small number of available instances (total of 206).

Prompt Template for Maritime Data Generation

Instruction: Given the list of events and their arguments, generate a corresponding maritime report. Produce a report that is diverse, drawing from the examples provided here. Each report may have more than one event.

Examples:

Events: [<sample event 1>]

Report: <Report 1>

Events: [<sample event 2>]

Report: <Report 2>

Events: [<sample event 3>]

Report: <Report 3>

Examples end here.

Event: [{event_type : <event_type1> arguments: [date: <date1>, time: <time1>, location: <Location1>, first_entity: <ship type1 or/and ship name1>, second_entity: <ship type2 or/and ship name2>, destination: <destination1>]},
{event_type : <event_type2> arguments: [date: <date2>, time: <time2>, location: <Location2>, first_entity: <ship type3 or/and ship name3>, second_entity: <ship type4 or/and ship name4>, destination: <destination2>]}]

Report:

Fig. 5. Prompt template for synthesizing maritime reports.

Original Sample

Report: On Oct 29, 2023, ADL ship ACCOLADE departed Adelaide for Melbourne. 2 X MEL vessels commenced escorting ADL vessel. ADL ship observed to require an adjustment in engineering mode to proceed with their designated path.

Events:

[{event_type: depart, arguments: [date: Oct 29, 2023, location: Adelaide, first_entity: ADL ship ACCOLADE, destination: Melbourne]},
{event_type: escort, arguments: [first_entity: 2 X Mel vessels, second_entity: ADL vessel]},
{event_type: alter_course, arguments: [First_entity: ADL ship]}]

Synthesised Sample

Events:

[{event_type: monitor, arguments: [date: March 01, 2023, time: 18:20, location: Abbot Point, first_entity: maritime authorities' vessel, second_entity: Seafarer's Pride vessel]},
{event_type: cease, arguments: [First_entity: Seafarer's Pride, second_entity: maritime authorities' vessel]}]



Report: On March 01, 2023, at 18:20, A maritime authorities' vessel started monitoring Seafarer's Pride vessel in the vicinity of Abbot Point. Seafarer's Pride was intercepted and detained by maritime authorities' vessel for suspected illegal activities pending inspection.

Fig. 6. Samples of original maritime reports and synthesized ones from MaritimeEvent dataset.

VI. ANALYSIS

In this section, we conduct comprehensive studies to analyze the design of our method from different perspectives.

1) *Impact of Decomposed Prompting:* For a complex and challenging EE task, to enhance the performance of our approach, we design effective demonstration examples using RAE, and further, decompose the task into simpler sub-tasks via prompting. Table II presents $Our_{Without\ Decomp}$, indicating the performance of our proposed approach utilizing gpt-3.5-turbo without prompt decomposition across two datasets, ACE05-EN and MaritimeEvent. In this experiment, we aim to extract all event types, their triggers, and corresponding arguments in a single step. Consequently, the prompt necessitates the inclusion of all schema information, encompassing event-type definitions and their arguments. Conversely, in a two-step decomposed prompting strategy, we utilize the event type definitions in the ED prompt (Step 1) to detect the event type. In the second step, we only incorporate the schema information of the detected events in the previous step in the EAE prompt (Step 2). It is important to note that for document-level EE, a full prompt would be excessively lengthy, requiring

the utilization of a costly OpenAI model (e.g., gpt-4-32k) for execution. For instance, employing decomposed prompting in a 5-shot RAE setting using ChatGPT(gpt-3.5-turbo) can improve F1 scores for ED and EAE in the ACE05-En dataset by 8.3 and 4.64 points, respectively.

2) *Impact of Embedding Methods in Retrieval Augmented Examples:* We utilize three different state-of-the-art embedding techniques to encode the existing instances as follows:

- **text-embedding-ada-002 (ADA-002).** ADA-002 [44] is OpenAI's second-generation embeddings-as-a-service API endpoint model specifically adapted for text embeddings. By default, the size of the embedding vector of ADA-002 is 1536. ADA-002 is recommended by OpenAI for text similarity tasks. To retrieve the most relevant instances, ADA-002 uses the cosine similarity between the embedding vectors of the query instance and each instance in the training dataset and returns the highest-scored instances.
- **Universal Sentence Encoder (USE).** USE [42] is a widely used sentence encoding model released by Google that provides sentence-level embedding vectors imple-

TABLE II

EXPERIMENT RESULTS ON THREE DATASETS. TRIG-C INDICATES TRIGGER IDENTIFICATION AND CLASSIFICATION. ARG-C INDICATES ARGUMENT IDENTIFICATION AND CLASSIFICATION. THE EVALUATION METRIC IS F1-SCORE. *: THIS MODEL WAS FINE-TUNED ON THE TRAINING DATASET. THE LLMs WERE TESTED USING DIRECT IN-CONTEXT LEARNING AND WERE NOT TRAINED OR FINE-TUNED.

ACE205-EN									
Model	Prompt type	Zero-shot		One-shot		5-shot		5-shot RAE	
	Language Model	Trig-C	Arg-C	Trig-C	Arg-C	Trig-C	Arg-C	Trig-C	Arg-C
Text2Event * [38]	T5-large	71.9	53.8						
OntoGPT [40]	ChatGPT	43.58	20.56	52.38	31.7	66.53	38.55	70.43	44.68
	GPT-4	50.14	24.79	60.46	43.04	72.92	47.09	77.91	51.07
Fine-Grained IE [30]	ChatGPT	37.38	7.39	57.87	14.64	71.17	34.40		
Ours _{without Decomp}	ChatGPT	38.08	7.39	56.11	16.61	68.05	33.87	69.19	45.43
	Llama2-7B	13.46	5.31	18.67	9.63	23.96	16.61	30.49	18.58
Ours	ChatGPT	44.08	19.96	58.33	32.80	71.74	42.88	74.55	50.07
	GPT-4	49.87	25.98	63.27	45.51	75.91	51.95	81.09	58.24

MaritimeEvent									
Model	Prompt type	Zero-shot		One-shot		5-shot		5-shot RAE	
	Language Model	Trig-C	Arg-C	Trig-C	Arg-C	Trig-C	Arg-C	Trig-C	Arg-C
Text2Event * [38]	T5-large	78.52	59.34						
OntoGPT [40]	ChatGPT	51.63	36.99	64.63	42.49	68.89	46.5	71.25	50.94
	GPT-4	55.04	45.05	69.04	48.15	74.69	51.69	80.41	55.90
Ours _{without Decomp}	ChatGPT	38.08	7.39	56.11	16.61	69.56	33.87	69.99	49.43
	Llama2-7B	18.78	9.56	25.78	14.56	31.05	18.92	40.97	24.87
Ours	ChatGPT	50.56	38.17	63.56	45.17	68.32	49.28	74.39	55.67
	GPT-4	55.23	44.78	70.23	50.78	75.61	55.33	84.32	60.79

WikiEvent									
Model	Prompt type	Zero-shot		One-shot		5-shot		5-shot RAE	
	Language Model	Trig-C	Arg-C	Trig-C	Arg-C	Trig-C	Arg-C	Trig-C	Arg-C
Text2Event * [38]	T5-large	29.34	17.41						
OntoGPT [40]	ChatGPT	33.67	19.75	46.68	32.55	54.93	32.89	55.69	40.9
	GPT-4	41.55	29.67	55.65	43.78	60.99	45.87	61.38	44.91
	Llama2-7B	10.30	6.42	16.30	9.72	21.06	13.58	20.55	14.05
Ours	ChatGPT	39.08	24.96	48.08	36.85	56.73	40.96	57.89	41.07
	GPT-4	42.66	29.39	56.27	40.68	63.92	45.60	64.65	45.96

TABLE III

IMPACT OF EMBEDDING STRATEGY ON PERFORMANCE OF OUR APPROACH USING GPT-4-TURBO. ACE05-EN IS USED FOR THIS EXPERIMENTS.

Embedding Model	Embedding Vector(size)	Trig-C	Arg-C
USE [42]	512	77.52	56.78
RoBERTa-base [43]	768	78.87	56.34
ADA-002 [44]	1536	81.09	58.24

mented using a transformer. USE embeds each sentence with a 512 vector representation.

- **RoBERTa-base.** RoBERTa-base [43] is a pre-trained model for text embedding. By default, it represents a sentence or paragraph by a vector with a length of 768.

Next, the encoded instance vectors are indexed using the *IndexFlatL2* structure provided by Facebook AI Similarity Search (FAISS). Table III illustrates the performance of our approach using gpt-4-turbo when we utilize different embedding models on the ACE05-EN dataset. We observed that ADA-002 encodes the instances into vectors with the highest size, which is 1536. It is evident that encoding instances using the ADA-002 model achieves the highest F1 scores in both the ED and EVA sub-tasks.

VII. CONCLUSION

This work addresses the challenges of structured event extraction from natural language data. Our approach leverages the capabilities of LLMs for textual understanding and enhances the accuracy of extracted events by integrating prompt enrichment techniques inspired by prompt decomposition and Retrieval Augmented Generation. Our findings reveal that utilizing LLMs enriched with schema-aware granular instructions and the retrieved augmented examples for the query substantially enhances performance. This heightened accuracy in event extraction enhances the reliability and effectiveness of automated knowledge graph construction and visualization of knowledge graphs, thereby supporting downstream reasoning and facilitating the decision-making process.

REFERENCES

- [1] F. Shiri, T. Wang, S. Pan, X. Chang, Y.-F. Li, R. Haffari, V. Nguyen, and S. Yu, "Toward the automated construction of probabilistic knowledge graphs for the maritime domain," in *2021 IEEE 24th International Conference on Information Fusion (FUSION)*. IEEE, 2021, pp. 1–8.
- [2] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat *et al.*, "Gpt-4 technical report," *arXiv preprint arXiv:2303.08774*, 2023.
- [3] B. Jensen and D. Tadross, "How large-language models can revolutionize military planning," *War on the Rocks*, vol. 12, 2023.

- [4] Z. Ji, N. Lee, R. Frieske, T. Yu, D. Su, Y. Xu, E. Ishii, Y. J. Bang, A. Madotto, and P. Fung, "Survey of hallucination in natural language generation," *ACM Computing Surveys*, vol. 55, no. 12, pp. 1–38, 2023.
- [5] *Halueval: A large-scale hallucination evaluation benchmark for large language models*, 2023.
- [6] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou *et al.*, "Chain-of-thought prompting elicits reasoning in large language models," *Advances in Neural Information Processing Systems*, vol. 35, pp. 24 824–24 837, 2022.
- [7] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel *et al.*, "Retrieval-augmented generation for knowledge-intensive nlp tasks," *Advances in Neural Information Processing Systems*, vol. 33, pp. 9459–9474, 2020.
- [8] *ASER: A large-scale eventuality knowledge graph*, 2020.
- [9] R. Han, I.-H. Hsu, J. Sun, J. Baylon, Q. Ning, D. Roth, and N. Peng, "Ester: A machine reading comprehension dataset for reasoning about event semantic relations," in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 2021, pp. 7543–7559.
- [10] Y. Lu, H. Lin, J. Xu, X. Han, J. Tang, A. Li, L. Sun, M. Liao, and S. Chen, "Text2event: Controllable sequence-to-structure generation for end-to-end event extraction," *arXiv preprint arXiv:2106.09232*, 2021.
- [11] F. Moghimifar, F. Shiri, V. Nguyen, Y.-F. Li, and G. Haffari, "Theia: Weakly supervised multimodal event extraction from incomplete data," in *Proceedings of the 13th International Joint Conference on Natural Language Processing and the 3rd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics (Volume 2: Short Papers)*, 2023, pp. 139–145.
- [12] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, "Language models are few-shot learners," *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.
- [13] J. M. Christopher Walker, Stephanie Strassel and K. Maeda, "Ace 2005 multilingual training corpus," <https://catalog.ldc.upenn.edu/LDC2006T06>, 2006.
- [14] Y. Lin, H. Ji, F. Huang, and L. Wu, "A joint neural model for information extraction with global features," in *Proceedings of the 58th annual meeting of the association for computational linguistics*, 2020, pp. 7999–8009.
- [15] T. M. Nguyen and T. H. Nguyen, "One for all: Neural joint modeling of entities and events," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 33, no. 01, 2019, pp. 6851–6858.
- [16] F. Shiri, X. Yu, F. Porikli, R. Hartley, and P. Koniusz, "Recovering faces from portraits with auxiliary facial attributes," in *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 2019, pp. 406–415.
- [17] F. Shiri, T. Wu, Y. Li, and G. Haffari, "Tcg-event: Effective task conditioning for generation-based event extraction," in *Proceedings of the The 20th Annual Workshop of the Australasian Language Technology Association*, 2022, pp. 22–30.
- [18] G. Paolini, B. Athiwaratkun, J. Krone, J. Ma, A. Achille, R. Anubhai, C. N. d. Santos, B. Xiang, and S. Soatto, "Structured prediction as translation between augmented natural languages," *arXiv preprint arXiv:2101.05779*, 2021.
- [19] X. Du, A. M. Rush, and C. Cardie, "Grit: Generative role-filler transformers for document-level event entity extraction," *arXiv preprint arXiv:2008.09249*, 2020.
- [20] T. Wu, F. Shiri, J. Kang, G. Qi, G. Haffari, and Y.-F. Li, "Kc-gee: knowledge-based conditioning for generative event extraction," *World Wide Web*, vol. 26, no. 6, pp. 3983–3999, 2023.
- [21] J. Wei, Y. Tay, R. Bommasani, C. Raffel, B. Zoph, S. Borgeaud, D. Yogatama, M. Bosma, D. Zhou, D. Metzler *et al.*, "Emergent abilities of large language models," *arXiv preprint arXiv:2206.07682*, 2022.
- [22] K. Mahowald, A. A. Ivanova, I. A. Blank, N. Kanwisher, J. B. Tenenbaum, and E. Fedorenko, "Dissociating language and thought in large language models: a cognitive perspective," *arXiv preprint arXiv:2301.06627*, 2023.
- [23] F. Moghimifar, F. Shiri, R. Haffari, Y.-F. Li, and V. Nguyen, "Few-shot domain-adaptive visually-fused event detection from text," in *2023 26th International Conference on Information Fusion (FUSION)*. IEEE, 2023, pp. 1–8.
- [24] K. Cobbe, V. Kosaraju, M. Bavarian, M. Chen, H. Jun, L. Kaiser, M. Plappert, J. Tworek, J. Hilton, R. Nakano *et al.*, "Training verifiers to solve math word problems," *arXiv preprint arXiv:2110.14168*, 2021.
- [25] B. Li, G. Fang, Y. Yang, Q. Wang, W. Ye, W. Zhao, and S. Zhang, "Evaluating chatgpt's information extraction capabilities: An assessment of performance, explainability, calibration, and faithfulness," *arXiv preprint arXiv:2304.11633*, 2023.
- [26] R. Han, T. Peng, C. Yang, B. Wang, L. Liu, and X. Wan, "Is information extraction solved by chatgpt? an analysis of performance, evaluation criteria, robustness and errors," *arXiv preprint arXiv:2305.14450*, 2023.
- [27] O. Sharif, M. Basak, T. Parvin, A. Scharfstein, A. Bradham, J. T. Borodovsky, S. E. Lord, and S. M. Preum, "Characterizing information seeking events in health-related social discourse," *arXiv preprint arXiv:2308.09156*, 2023.
- [28] X. F. Zhang, C. Blum, T. Choji, S. Shah, and A. Vempala, "Ultra: Unleash llms' potential for event argument extraction through hierarchical modeling and pair-wise refinement," *arXiv preprint arXiv:2401.13218*, 2024.
- [29] X. Wei, X. Cui, N. Cheng, X. Wang, X. Zhang, S. Huang, P. Xie, J. Xu, Y. Chen, M. Zhang *et al.*, "Zero-shot information extraction via chatting with chatgpt (arxiv: 2302.10205). arxiv," 2023.
- [30] J. Gao, H. Zhao, Y. Zhang, W. Wang, C. Yu, and R. Xu, "Benchmarking large language models with augmented instructions for fine-grained information extraction," *arXiv preprint arXiv:2310.05092*, 2023.
- [31] J. Johnson, M. Douze, and H. Jégou, "Billion-scale similarity search with gpus," *IEEE Transactions on Big Data*, vol. 7, no. 3, pp. 535–547, 2019.
- [32] H. Junqing, P. Kunhao, D. Xiaoqun, S. Zhuoyang, L. Yibo, L. Yuxin, W. Hao, S. Qianguo, Z. Songxin, X. Zejian *et al.*, "Never lost in the middle: Improving large language models via attention strengthening question answering," *arXiv preprint arXiv:2311.09198*, 2023.
- [33] P. Xu, W. Ping, X. Wu, L. McAfee, C. Zhu, Z. Liu, S. Subramanian, E. Bakhturina, M. Shoybi, and B. Catanzaro, "Retrieval meets long context large language models," *arXiv preprint arXiv:2310.03025*, 2023.
- [34] T. Khot, H. Trivedi, M. Finlayson, Y. Fu, K. Richardson, P. Clark, and A. Sabharwal, "Decomposed prompting: A modular approach for solving complex tasks," *arXiv preprint arXiv:2210.02406*, 2022.
- [35] A. Drozdov, N. Schärli, E. Akyürek, N. Scales, X. Song, X. Chen, O. Bousquet, and D. Zhou, "Compositional semantic parsing with large language models," *arXiv preprint arXiv:2209.15003*, 2022.
- [36] J. Wei, M. Bosma, V. Y. Zhao, K. Guu, A. W. Yu, B. Lester, N. Du, A. M. Dai, and Q. V. Le, "Finetuned language models are zero-shot learners," *arXiv preprint arXiv:2109.01652*, 2021.
- [37] S. Li, H. Ji, and J. Han, "Document-level event argument extraction by conditional generation," *arXiv preprint arXiv:2104.05919*, 2021.
- [38] Y. Lu, H. Lin, J. Xu, X. Han, J. Tang, A. Li, L. Sun, M. Liao, and S. Chen, "Text2event: Controllable sequence-to-structure generation for end-to-end event extraction," *arXiv preprint arXiv:2106.09232*, 2021.
- [39] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," *The Journal of Machine Learning Research*, vol. 21, no. 1, pp. 5485–5551, 2020.
- [40] J. H. Caufield, H. Hegde, V. Emonet, N. L. Harris, M. P. Joachimiak, N. Matentzoglou, H. Kim, S. A. Moxon, J. T. Reese, M. A. Haendel *et al.*, "Structured prompt interrogation and recursive extraction of semantics (spires): A method for populating knowledge bases using zero-shot learning," *arXiv preprint arXiv:2304.02711*, 2023.
- [41] S. A. Moxon, H. Solbrig, D. R. Unni, D. Jiao, R. M. Bruskiewich, J. P. Balhoff, G. Vaidya, W. D. Duncan, H. Hegde, M. Miller *et al.*, "The linked data modeling language (linkml): A general-purpose data modeling framework grounded in machine-readable semantics," in *ICBO*, 2021, pp. 148–151.
- [42] D. Cer, Y. Yang, S.-y. Kong, N. Hua, N. Limtiaco, R. S. John, N. Constant, M. Guajardo-Cespedes, S. Yuan, C. Tar *et al.*, "Universal sentence encoder," *arXiv preprint arXiv:1803.11175*, 2018.
- [43] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized bert pretraining approach," *arXiv preprint arXiv:1907.11692*, 2019.
- [44] A. Neelakantan, T. Xu, R. Puri, A. Radford, J. M. Han, J. Tworek, Q. Yuan, N. Tezak, J. W. Kim, C. Hallacy *et al.*, "Text and code embeddings by contrastive pre-training," *arXiv preprint arXiv:2201.10005*, 2022.