ULTRA: Unleash LLM’s Potential for Event Argument Extraction through Hierarchical Modeling and Pair-wise Refinement

Xinliang Frederick Zhang\textsuperscript{1,∗}, Carter Blum\textsuperscript{2}, Temma Choji\textsuperscript{2}, Shalin Shah\textsuperscript{2}, and Alakananda Vempala\textsuperscript{2}

\textsuperscript{1}Computer Science and Engineering, University of Michigan
\textsuperscript{2}Bloomberg
\textsuperscript{1}xlfzhang@umich.edu
\textsuperscript{2}\{cblum18, tchoji, sshah804, avempala\}@bloomberg.net

Abstract

Structural extraction of events within discourse is critical since it avails a deeper understanding of communication patterns and behavior trends. Event argument extraction (EAE), at the core of event-centric understanding, is the task of identifying role-specific text spans (i.e., arguments) for a given event. Document-level EAE (DocEAE) focuses on arguments that are scattered across an entire document. In this work, we explore the capabilities of open-sourced Large Language Models (LLMs), i.e., Flan-UL2, for the DocEAE task. To this end, we propose ULTRA, a hierarchical framework that extracts event arguments more cost-effectively — the method needs as few as 50 annotations and doesn’t require hitting costly API endpoints. Further, it alleviates the positional bias issue intrinsic to LLMs. ULTRA first sequentially reads text chunks of a document to generate a candidate argument set, upon which ULTRA learns to drop non-pertinent candidates through self-refinement. We further introduce LEAFER to address the challenge LLMs face in locating the exact boundary of an argument span. ULTRA outperforms strong baselines that include strong supervised models and ChatGPT by 9.8\% when evaluated by the exact match (EM) metric.

1 Introduction

Event extraction (EE), a long-standing and prominent information extraction (IE) task, aims to extract event structures consisting of core information elements (e.g., “who” did “what” to “whom”, “when”, “where”, and “why”) from unstructured texts (Mourelatos, 1978; Riloff, 1996; Walker et al., 2005; Du and Cardie, 2020b). Event-centric understanding is of great importance, not only in its inherent merits, but also by its role as an information-rich intermediate representation for downstream tasks such as summarization (Filatova and Hatzivassiloglou, 2004; Marujo et al., 2017; Li et al., 2021a), recommendation systems (Lu et al., 2016; Li et al., 2020a) and news narrative understanding (Jin et al., 2022; Zhang et al., 2022; Keith Norambuena et al., 2023). Event argument extraction (EAE), a crucial and challenging step in Event Extraction (EE), is the task of identifying role-specific text spans (i.e., arguments) for a given event (Nguyen et al., 2016; Kar et al., 2020; Du and Cardie, 2020a).

Existing research mainly focuses on sentence-level event argument extraction (Chen et al., 2015; Du and Cardie, 2020b; Lu et al., 2021) on the prevalent ACE dataset (Walker et al., 2005). Yet, in the realm of news journalism, events are usually de-
scribed at the document level, and arguments are typically scattered across an entire article (Hamburg et al., 2019). Therefore, there is a pressing need to systematically study the document-level EAE (DocEAE) task, since sentence-level EAE systems cannot accommodate long-distance dependency (Ebner et al., 2020), cross-sentence inference (Li et al., 2021b) and multi-answer (Tong et al., 2022) problems intrinsic to DocEAE task. Traditional supervised approaches have to consume large-scale annotations (e.g., Zheng et al., 2019; Pouran Ben Veyseh et al., 2022), each requires more than 30k annotated articles) in order to excel, and the state-of-the-art EAE model requires manual design of templates for each argument type (Hsu et al., 2022). These approaches are not only costly, but also not generalizable since they fail to handle emerging events (Yang et al., 2023).1 Recently, there has been a notable surge in applications of Large Language Models (LLMs) for NLP tasks, especially closed ones such as PaLM (Chowdhery et al., 2022), Claude (Bai et al., 2022) and GPT-4 (OpenAI, 2023). The most relevant work to ours is Li et al. (2023); Han et al. (2023), which, however, only performed preliminary analysis by assessing ChatGPT’s capability of solving IE tasks. Meanwhile, there is no prior research that has attempted to leverage LLMs to tackle the DocEAE task. In our preliminary investigation, we have identified at least three challenges that arise when employing closed LLMs: 1) hitting endpoints incurs substantial costs and poses scalability challenges during inference; 2) undesirable prompt hacking is needed to ensure performance (Ouyang et al., 2022); 3) given the nature of news, information spreading across the content, LLMs suffer from the positional bias issue2 (Hou et al., 2023; Liu et al., 2023).

To this end, we propose an easy-to-use framework that Unleashes LLM’s potential for event argument exTRAction through hierarchical modeling and pair-wise refinement, dubbed ULTRA. ULTRA, built on Flan-UL2 (Tay et al., 2022), first sequentially reads text chunks of a news article to generate a candidate argument set. ULTRA then learns to drop non-pertinent candidates through self-refinement by means of pairwise comparison. Further, A LEAFER module, LEAring From ERrors, is implemented to improve boundary iden-

---

1 COVID-19 became an emerging topic since 2020 (Wang et al., 2020; Zhang et al., 2021), but not covered in traditional EE corpora (Walker et al., 2005; Ebner et al., 2020).

2 Also known as lost in the middle.

---

1. Event Argument Extraction (EAE)

Most event argument extraction research has experimented on the 2005 Automatic Content Extraction (ACE 2005; Walker et al., 2005), while recent work delves into domain-specific areas such as biomedical texts (Zhao et al., 2020; He et al., 2022), legal documents (Li et al., 2020b; Shen et al., 2020), morality-bearing contents (Zhang et al., 2023) and conversations (Srivastava et al., 2023). Existing work primarily focused on the sentence-level event understanding task. Most methods can be categorized under one of the three following approaches: sequence labeling (Chen et al., 2015; Nguyen et al., 2016) where Lin et al. (2020) further constrains the inference by imposing global features; question answering (Du and Cardie, 2020b) which incorporates ontology knowledge about argument roles; and generative models for structured extraction (Paolini et al., 2021; Lu et al., 2021). Particularly, DEGREE reformulates EAE as a template-based conditional generation task, and archives impressive performance on various benchmarks (Hsu et al., 2022). However, it demands huge annotation efforts, which require one template for each argument type, and is therefore not generalizable or scalable. In this work, we are seeking to improve EAE performance without the need for argument-specific templates but general instructions (Table A3).

Lately, there has been an increasing interest in document-level EAE (DocEAE), since events are usually described at the document level and arguments are usually scattered across multiple sen-
Table 2: Results on DocEE dataset for document-level event argument extraction task, and breakdown of EM and HM scores by Precision (P), recall (R) and F1. We additionally report estimated monetary cost by model category, divided into training and inference costs. Best results are bold. Best F1’s in the literature are underlined. ULTRA achieves the best F1 performances at a reduced cost, and recovers more true positives than any baseline. Additional results based on part of ULTRA can be found in Table A2. *Results are directly taken from Tong et al. (2022). **Results are reported as the average performance of 5 custom instructions, where individual performances are included in Table A1. ***Unless otherwise noted, experiments are conducted in zero-shot manner.

<table>
<thead>
<tr>
<th>Category</th>
<th>Method</th>
<th>Performance</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>EM</td>
<td>HM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>P</td>
<td>R</td>
</tr>
<tr>
<td>Supervised ML</td>
<td>EEQA* (Du and Cardie, 2020b)</td>
<td>29.4</td>
<td>20.3</td>
</tr>
<tr>
<td></td>
<td>Onology QA* (Tong et al., 2022)</td>
<td>36.6</td>
<td>25.2</td>
</tr>
<tr>
<td>Closed LLM</td>
<td>ChatGPT (Li et al., 2023)</td>
<td>35.6</td>
<td>18.0</td>
</tr>
<tr>
<td></td>
<td>ChatGPT (single question)</td>
<td>30.9</td>
<td>22.7</td>
</tr>
<tr>
<td></td>
<td>CoT-ChatGPT (Wang et al., 2023b)</td>
<td>31.4</td>
<td>16.2</td>
</tr>
<tr>
<td>Flan-UL2</td>
<td>Custom instructions**</td>
<td>27.6</td>
<td>17.8</td>
</tr>
<tr>
<td></td>
<td>Aligned instruction</td>
<td>36.1</td>
<td>20.7</td>
</tr>
<tr>
<td></td>
<td>Few-shot aligned instruction***</td>
<td>36.8</td>
<td>20.8</td>
</tr>
<tr>
<td>ULTRA (Ours)</td>
<td>ULTRA-base</td>
<td>29.0</td>
<td>34.5</td>
</tr>
<tr>
<td></td>
<td>+ Ensemble</td>
<td>28.0</td>
<td>39.4</td>
</tr>
<tr>
<td></td>
<td>ULTRA-long</td>
<td>32.3</td>
<td>30.5</td>
</tr>
<tr>
<td></td>
<td>+ Ensemble</td>
<td>30.2</td>
<td>35.5</td>
</tr>
</tbody>
</table>

Table 2: Results on DocEE dataset for document-level event argument extraction task, and breakdown of EM and HM scores by Precision (P), recall (R) and F1. We additionally report estimated monetary cost by model category, divided into training and inference costs. Best results are bold. Best F1’s in the literature are underlined. ULTRA achieves the best F1 performances at a reduced cost, and recovers more true positives than any baseline. Additional results based on part of ULTRA can be found in Table A2. *Results are directly taken from Tong et al. (2022). **Results are reported as the average performance of 5 custom instructions, where individual performances are included in Table A1. ***Unless otherwise noted, experiments are conducted in zero-shot manner.

Tences (Sundheim, 1992; Hamborg et al., 2019; Tong et al., 2022). For example, RAMS (Ebner et al., 2020) and MEE (Pouran Ben Veyseh et al., 2022) both define “document” as a 5-sentence segment. In contrast, WikiEvents (Li et al., 2021b) and DocEE (Tong et al., 2022) present full articles and focus on argument extractions for the main event. In this work, we use DocEE as a benchmark since it features broad coverage of event types in the news domain. Methodology-wise, Du and Cardie (2020a) and Tong et al. (2022) handle DocEAE by extending sentence-level labeling and question-answering approaches, respectively. Li et al. (2021b) frames DocEAE as template-based conditional generation in the same vein as the sentence-level generative approach. Unfortunately, none of the aforementioned methods tackle the argument-scattering challenge; instead, they treat a full article as if it were an extended sentence. Zheng et al. (2019) is the first work to address this issue by modeling DocEAE as an entity-centric graph, which is further augmented with a “tracker” module to capture the interdependency among arguments and events (Xu et al., 2021). Nonetheless, the “tracker” is insufficient due to its limitation of not considering the results of later extractions when processing earlier ones. On the contrary, our ULTRA bridges the gap through the implementation of a self-refinement module, which is grounded in pairwise comparison and functions akin to a bi-directional tracker.

2.2 Using Large Language Models for IE

The past years have witnessed the rise of highly parallelizable, scalable transformer architecture (Vaswani et al., 2017), paving the way for a series of powerful language models that have significantly reshaped the NLP landscape (Devlin et al., 2019; Lewis et al., 2020; Raffel et al., 2019; Radford et al., 2019). Recently, studies have evinced that scaling up model sizes yields more impressive abilities on various tasks (Hoffmann et al., 2022), as well as unlocks an emergent ability that is not present in smaller models (Wei et al., 2022a). These large language models (LLMs), often exceeding a hundred billion parameters, are typically closed systems (i.e., no open checkpoints available). Notable examples include PaLM (Chowdhery et al., 2022), Claude (Bai et al., 2022), GPT-4 (OpenAI, 2023). Numerous methods are developed to enhance LLM’s reasoning and problem-solving capabilities, such as chain-of-thought (Wei et al., 2022b), self-correction (Pan et al., 2023), and external tool (e.g., python interpreter) augmentation (Gao et al., 2022; Chen et al., 2023b) among others. ChatGPT, one of the most burgeoning LLM, is

https://chat.openai.com/
trained on high-quality conversation datasets using reinforcement learning from human feedback (RLHF; Christiano et al., 2017), and had led to a transformative wave. The most relevant to our research is leveraging ChatGPT for the information extraction task (Li et al., 2023; Han et al., 2023), including named entity recognition (Xie et al., 2023), temporal relation extraction (Yuan et al., 2023), event detection (Sharif et al., 2023), and event argument extraction (Wei et al., 2023). Their primary focus is either benchmarking ChatGPT’s performances, which shows inferior results to specialized supervised IE systems (Li et al., 2023; Han et al., 2023), or curating new benchmark datasets (Gao et al., 2023). In contrast, our proposed ULTRA framework outperforms strong baselines including the previous SOTA models, capitalizing on the effectiveness of our designed LEAFER and self-reflection modules. Besides, to the best of our knowledge, we are the first to exploit LLMs for the DocEAE task.

3 Methodology

Figure 1 depicts the overall framework of ULTRA. Taking as input a news article \( \mathbf{d} \), ULTRA first sequentially reads text chunks of the article \( \mathbf{d} \) to generate a candidate argument set \( \{ \mathbf{a} \} \) (§3.1), upon which ULTRA learns to drop non-pertinent candidates through self-refinement and returns a final argument set \( \{ \mathbf{a}^f \} \) (§3.3). A new module LEAFER, LEArning From ERrors, is introduced to tackle LLMs’ incapability of locating the exact boundary of an argument span (§3.2). ULTRA+ is a variant augmented with extractions by a document-level extractor model to capture information snippets that require full-article discourse analysis (e.g., extracting “why”-type arguments).

Putting all together, we produce two versions of ULTRA: ULTRA-base and ULTRA-long, which consume 5-sentence and 15-sentence windows in layer-1, respectively. It is worth noting that, instead of conducting costly prompt hacking (Ouyang et al., 2022), we adopt an existing instruction from NIv2 (Wang et al., 2022) and tailor it our use case, named aligned instruction. We provide our used task instructions (\( \{ I \} \)) in Table A3.

3.1 Layer-1: Local Understanding

Given a document \( \mathbf{d} \), we first divide \( \mathbf{d} \) to multiple \( k \)-sentence passage windows with a step size of \( \lfloor \frac{k}{2} \rfloor \), denoted as \( \{ w_1, w_2, \ldots, w_l \} \). We adopt a fixed-window-size approach instead of a fixed-sequence-length approach (Devlin et al., 2019; Sun et al., 2019; Pappagari et al., 2019), which might cut a sentence in the middle, to allow each local extractor to comprehend each passage window in its entirety. Instantiated with Flan-UL2, the local extractor takes as input the concatenation of a task instruction (I), a passage window (\( w_i \)), and a question written in natural language (\( q_j \)), e.g., What is the “date” for the “Tsunami” event? We prompt the local extractor in a zero-shot fashion and explicitly instruct it to generate N/A if the input passage does not contain any relevant answer. After deduplication, we end up with a candidate argument set \( \{ \mathbf{a} \} \) for each question \( q_j \).

3.2 LEAFER Module

LLMs are deemed to have a knack for extracting relevant information (Li et al., 2023; Han et al., 2023), but we notice that LLMs still suffer from pinpointing the exact boundaries of an argument span. Specifically, as shown in Figure 1, local extractions (\( \{ \mathbf{a} \} \)) contain an apparently sensible answer “March and May” to the question “What is the ‘Date’ for the ‘Droughts’ event?”, which is syntactically similar but semantically different from the ground-truth answer “between March and May this year”. Such errors extend beyond number reasoning (e.g., time and count-type arguments), and are also evident in other argument types such as location. An illustrative case involves the LLM generating “Perth” which is part of the correct answer “Perth’s east”. To this end, we introduce a new module to alleviate this issue, LEAFER, short for LEArning From ERrors. LEAFER module is a small-scale LM trained on errors produced by Flan-UL2. The trained LEAFER is employed to generate a judgment, which is to explicitly inform what is wrong and why it is wrong. The insightful judgment enables ULTRA to rectify boundaries of candidate arguments in \( \{ \mathbf{a} \} \) and return \( \{ \mathbf{a}^f \} \).

To support the training of LEAFER, we construct a LEAFER Bank using the few-shot training set of 50 annotated articles. Specifically, we prompt the same Flan-UL2 local extractor to extract arguments for each \( (k\text{-sentence passage, question}) \) input pair using the approach outlined in §3.1. For each input pair, we match the machine-extracted argument

\[ \text{Putting all together, we produce two versions of ULTRA: ULTRA-base and ULTRA-long, which consume 5-sentence and 15-sentence windows in layer-1, respectively. It is worth noting that, instead of conducting costly prompt hacking (Ouyang et al., 2022), we adopt an existing instruction from NIv2 (Wang et al., 2022) and tailor it our use case, named aligned instruction. We provide our used task instructions (\( \{ I \} \)) in Table A3.} \]

\[ \text{3.1 Layer-1: Local Understanding} \]

\[ \text{Given a document \( \mathbf{d} \), we first divide \( \mathbf{d} \) to multiple \( k \)-sentence passage windows with a step size of \( \lfloor \frac{k}{2} \rfloor \), denoted as \( \{ w_1, w_2, \ldots, w_l \} \). We adopt a fixed-window-size approach instead of a fixed-sequence-length approach (Devlin et al., 2019; Sun et al., 2019; Pappagari et al., 2019), which might cut a sentence in the middle, to allow each local extractor to comprehend each passage window in its entirety. Instantiated with Flan-UL2, the local extractor takes as input the concatenation of a task instruction (I), a passage window (\( w_i \)), and a question written in natural language (\( q_j \)), e.g., What is the “date” for the “Tsunami” event? We prompt the local extractor in a zero-shot fashion and explicitly instruct it to generate N/A if the input passage does not contain any relevant answer. After deduplication, we end up with a candidate argument set \( \{ \mathbf{a} \} \) for each question \( q_j \).} \]

\[ \text{3.2 LEAFER Module} \]

\[ \text{LLMs are deemed to have a knack for extracting relevant information (Li et al., 2023; Han et al., 2023), but we notice that LLMs still suffer from pinpointing the exact boundaries of an argument span. Specifically, as shown in Figure 1, local extractions (\( \{ \mathbf{a} \} \)) contain an apparently sensible answer “March and May” to the question “What is the ‘Date’ for the ‘Droughts’ event?”, which is syntactically similar but semantically different from the ground-truth answer “between March and May this year”. Such errors extend beyond number reasoning (e.g., time and count-type arguments), and are also evident in other argument types such as location. An illustrative case involves the LLM generating “Perth” which is part of the correct answer “Perth’s east”. To this end, we introduce a new module to alleviate this issue, LEAFER, short for LEArning From ERrors. LEAFER module is a small-scale LM trained on errors produced by Flan-UL2. The trained LEAFER is employed to generate a judgment, which is to explicitly inform what is wrong and why it is wrong. The insightful judgment enables ULTRA to rectify boundaries of candidate arguments in \( \{ \mathbf{a} \} \) and return \( \{ \mathbf{a}^f \} \).} \]

\[ \text{To support the training of LEAFER, we construct a LEAFER Bank using the few-shot training set of 50 annotated articles. Specifically, we prompt the same Flan-UL2 local extractor to extract arguments for each \( (k\text{-sentence passage, question}) \) input pair using the approach outlined in §3.1. For each input pair, we match the machine-extracted argument} \]

\[ \text{5We observe that few-shot prompting yields inferior results} \]

\[ \text{6For brevity, we omit the subscript \( j \) in later sections.} \]
3.3 Layer-2: Self-Refinement

While LEAFER addresses the semantic drift and imprecise boundary issues, ULTRA exhibits over-generation issue due to window-based local extractors. Motivated by the recent successes leveraging an LLM as a judge (Zheng et al., 2023; Wang et al., 2023a), we propose a self-refinement module that allows ULTRA to introspectively reflect on candidate arguments \( \{a_i^d\} \), and pare down unlikely candidates through pairwise ranking. There are usually two variations of LLM-as-a-judge, single-answer grading and pairwise comparison. As studied in Zheng et al. (2023) and observed in our preliminary study, we find that single answer grading cannot serve as an effective refinement judge since 1) absolute scores are extremely inflated and a considerably large portion of scores are close to 1 on a scale of 0 to 1; and 2) single answer grading fails to capture subtle differences between a specific pair. Therefore, in layer 2, we leverage ranking by pairwise comparison (Jamieson and Nowak, 2011; Lee and Vajjala, 2022; Jiang et al., 2023) to obtain the final argument set, \( \{a_i^d\} \), by first prompting Flan-UL2 to pick a better answer between a candidate pair, then ranking all candidates by aggregating pairwise-comparison scores, and finally filtering out candidates at low positions. To support dynamic filtering, we decide on \( \{a_i^d\} \) as follows:

\[
|\{a_i^d\}| = 1 + \log_2(|\{a_i^d\}|) \tag{1}
\]

The pairwise comparison produces a non-trivial score and catches nuanced differences, is still trapped by the positional bias (Ko et al., 2020; Wang et al., 2023a; Liu et al., 2023) and lack of scalability due to the quadratic growth in pairwise comparisons. To mitigate these two issues, we resort to calibration and pruning, respectively.

**Calibration** In our context, positional bias depicts a model’s tendency to assign a higher score to an option at a particular position in a list, which has been shown existent in ChatGPT and GPT-4 (Wang et al., 2023a). The issue is manifested as Flan-UL2 biasing towards an answer displayed earlier. Drawing on the Contextual Calibration technique proposed in Zhao et al. (2021), as demonstrated in eq. (2), we calibrate the raw probabilities of each option between a pair to reveal the truthful probabilities, i.e., \( P(a_i|d) \).

\[
P(a_i|d) = \text{softmax}(g(P(a_i|d, I; \theta), P(a_i|I; \theta))) \tag{2}
\]

where \( P(a_i|\cdot) \) denotes the probability of an argument \( a_i \) being preferred given a certain input, and \( d \) and \( I \) denote the article and task instruction (see...
table A3 for the instruction). Following Zhao et al. (2021), \( g(x, y) \) is a calibration function that can be instantiated as additive, \( g(x, y) = x - y \), or multiplicative functions, \( g(x, y) = \frac{x}{y} \). Using our designed comparison instruction (I), we compute the prior probability \( P(a_i | I; \theta) \) by leaving the \{article\} field blank, while we fill in a concrete article when computing raw probability \( P(a_i | d; I; \theta). \) Through calibration function \( g(x, y) \), we manage to alleviate the positional bias induced by the input template I and the innate bias of LLMs, \( \theta \).

**Pruning** To tackle the scalability issue that the number of comparisons grows quadratically, we prune the candidate set to shrink its size upfront. Specifically, we design a strategy that aligns with the fundamental principles of news journalism, wherein journalists prioritize the presentation of crucial information at the outset of a news story, a practice commonly referred to as the “inverted pyramid” structure (Pottker, 2003; Hamborg et al., 2019; Liu et al., 2022). That is, we only consider up to 5 earliest candidate arguments, where the earliestness of an argument is determined by its first occurrence in a news article. In respect of computational efficiency, our pruning strategy empirically reduces the number of subsequent pairwise computations by half. Furthermore, we find that pruning itself can help improve precision, even without making pairwise comparisons. This also illuminates the validity of our designed pruning strategy.

3.4 **Ensembling: ULTRA+**

The ensembling technique consistently improves performance for a wide array of NLP tasks (Wang et al., 2019; Ganaie et al., 2022; Pitis et al., 2023; Jiang et al., 2023). LLM-Blender attempts to ensemble various LLM on output space (Jiang et al., 2023), which demands prohibitively many computational resources. Instead, we suggest a simpler and more efficient approach: merging outputs by both ULTRA and a document-level argument extractor, which reads the full article and a question when extracting arguments. This way, we manage to combine the benefits of both local (high recall) and document-level (high precision) extractions.

Similar to Labrak et al. (2023); Han et al. (2023), we also observe marginal improvement on the dev set when providing in-context examples. To reduce inference-time overhead, we prompt the document-level extractor in a zero-shot manner.

4 **Experiments**

**Dataset and Evaluation Metrics** We conduct experiments on DocEE dataset (Tong et al., 2022), which contains 27,485 news articles, classified into 59 event types, and 356 argument types. We use their cross-domain setting for our experiments since it only contains a minimally annotated target training set (i.e., 50 articles) which can best assess various models’ generalizability in the wild. Specifically, its test set contains 1,955 news articles covering 10 different event types, and each article is annotated with ~6.5 arguments. We use the same data split and processed texts as in the original DocEAE dataset (Tong et al., 2022) for fair comparison.

In terms of evaluation metric, we follow the literature on document-level event argument extraction (Du and Cardie, 2020a; Tong et al., 2022), and adopt Exact Match (EM) and Head noun phrase Match (HM) as evaluation metrics. EM assesses if an extracted argument exactly matches a reference, while HM is a relaxed metric that concerns if there is an overlap of head words of noun phrases between extractions and references.

**Baselines** In this paper, we compare ULTRA against three model families to comprehensively assess extraction performance and monetary costs. The first model family, Supervised ML characterized as using human annotations as supervision signal to train small-scale LMs, consists of EEQA (Du and Cardie, 2020b) and Ontology QA (Tong et al., 2022). Ontology QA is an extension of EEQA that additionally incorporates argument ontology knowledge, which achieves the SOTA performance for DocEAE. Secondly, we compare with ChatGPT using different prompting techniques, given its popularity and impressive capability. Specifically, Li et al. (2023) prompts ChatGPT to generate outputs in a dictionary format that contains both answers and rationales by extracting spans of all argument types in one pass. We further modify the original prompt to instruct ChatGPT to extract span(s) for only one argument type at a time (single-question variant). Wang et al. (2023b) proposes to generate a chain-of-thought rationale before summarizing an article. To have the best out of both worlds, we build the CoT-ChatGPT variant by replacing the summarizer in Wang et al. (2023b) with the

---

8We adopt the same evaluation script used in DocEE (Tong et al., 2022), which uses substring match as a proxy of original HM implementation (Du and Cardie, 2020a) for efficiency.
argument extractor used in Li et al. (2023). The last baseline suite involves prompting a document-level extractor with different instructions, utilizing Flan-UL2 as its backbone for a fair comparison. Concretely, it reads the entire article in one pass and extracts span(s) for only one argument type at a time. This baseline suite serves a threefold purpose: test Flan-UL2’s sensitivity to different custom instructions that are designed from scratch; illuminate the effectiveness of Aligned Instructions; examine the usefulness of few-shot examples when prompting Flan-UL2.

5 Results and Analysis

As shown in table 2, our proposed ULTRA achieves the best F1 scores across the board, especially compared to two strong baseline families, Supervised ML and Closed LLM, at a considerably reduced training- and inference-time monetary cost. It is worth noting that ULTRA significantly improves the EM recall by 56% over the best-performing model in the literature (39.4 vs. 25.2), demonstrating robust generalizability considering ULTRA’s exposure to at most 5-shot per event type.

Using ChatGPT for DocEAE Despite the common flaw of outputting seemingly coherent assertions that in reality are false, known as hallucination (Manakul et al., 2023; Feldman et al., 2023), we recognize another issue, which seems to be less studied in NLP community, that answer spans extracted by ChatGPT are verbose (Zheng et al., 2023; Chen et al., 2023a). This explains the reason why ChatGPT achieves the best HM scores in the literature since longer generations are more likely to contain relevant information, while the EM is low due to the nature of verbosity.

Cost Estimation Besides models' extraction performance, table 2 also presents the cost estimation of each model family. We briefly introduce the criteria used when estimating monetary costs. The training cost is mainly associated with document annotations.9 Regarding inference, we consider the expenses incurred in hitting API endpoints.10 Per Tong et al. (2022), both EEQA and Ontology QA are trained on 22k articles, each costs $0.9, totaling $20,000. Based on ChatGPT pricing,11 the base cost is $0.004/1k tokens. Processing each article and then producing answers would consume 5k to 50k tokens on average, depending on the input mode. The test set contains 2k examples, so the total cost is around $40 to $400. For Flan-UL2 baseline and our ULTRA, each only needs annotations of up to 50 articles, for retrieving few-shot examples and the training of LEAFER module, respectively. It is noteworthy that ULTRA enables cost-effective scaling at inference, while ChatGPT might face budget constraints.

Further Study on Window Size Despite ULTRA-base and ULTRA-long achieving almost identical EM F1 scores, they present different extraction properties where ULTRA-base reaches the highest recall while ULTRA-long is more balanced. In this subsection, we specifically look into the extraction property of the Layer-1-only variant of ULTRA. Figure 2 shows the performance trend with the window size. We notice that precision steadily goes up while recall goes down since fewer chunks are fed into ULTRA. The F1 performances plateau after the window size of 15.

Figure 2: The impact of window size on the performance of Layer-1-only variant of ULTRA. Results are based on dev set. With the window size, precision goes up while recall goes down since fewer chunks are fed into ULTRA. The F1 performances plateau after the window size of 15.
of a product or the target audience consists of vulnerable populations susceptible to misinformation.

6 Conclusion

In this study, we present ULTRA, a cost-effective event argument extraction framework, built upon an open-sourced LLM – Flan-UL2. Concretely, ULTRA reads a sequence of text chunks from an article, the outputs of which are refined through self-refinement by removing non-relevant answers. With minimal annotation efforts, a LEAFER module is implemented to improve argument span boundary identification. Our experimental results show the superiority of ULTRA in comparison to both supervised ML models and closed LLMs. Further analyses showcase the customizability of ULTRA to cater to different extraction criteria.

Acknowledgements

The authors thank Bloomberg AI team for their support of this work. We would like to thank Shijie Wu for hardware support and discussions on FLAN-UL2 setup configurations. We appreciate Igor Malioutov, Daniel Preotiuc-Pietro and Umut Topkara for their insightful inputs and helpful feedback at different stages of this project.

References


Lingjiao Chen, Matei Zaharia, and James Y. Zou. 2023a. How is chatgpt’s behavior changing over time? ArXiv, abs/2307.09009.


Jun Gao, Huan Zhao, Yice Zhang, Wei Wang, Changlong Yu, and Ruifeng Xu. 2023. Benchmarking large language models with augmented instructions for fine-grained information extraction. ArXiv, abs/2310.05092.


Felix Hamborg, Corinna Breitinger, and Bela Gipp. 2019. Giveme5w1h: A universal system for extracting main events from news articles. In INRA@RecSys.


Xinyu He, Ping Tai, Hongbin Lu, Xin Huang, and Yong-gong Ren. 2022. A biomedical event extraction method based on fine-grained and attention mechanism. BMC Bioinformatics, 23.


Di Lu, Clare Voss, Fangbo Tao, Xiang Ren, Rachel Guan, Rostyslav Korolov, Tongtao Zhang, Dongang Wang, Hongzhi Li, Taylor Cassidy, Heng Ji, Shih-fu Chang, Jiawei Han, William Wallace, James Hendler, Mei Si, and Lance Kaplan. 2016. Cross-media event extraction and recommendation. In Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations, pages 72–76, San Diego, California. Association for Computational Linguistics.


Xiang Wei, Xingyu Cui, Ning Cheng, Xiaobin Wang, Xin Zhang, Shen Huang, Pengjun Xie, Jinan Xu, Yufeng Chen, Meishan Zhang, Yong Jiang, and Wenjuan Han. 2023. Zero-shot information extraction via chatting with chatgpt. ArXiv, abs/2302.10205.


Table A1: Performances of each individual custom instruction. `[e_type]`, `[arg_role]` and `{news}` are placeholders to be filled with event type, argument role and news content, respectively. Flan-UL2 is considerably sensitive to the input instruction, and even with a tiny change in the question, the model performance varies a lot, as manifested by contrasting instruction ID 3 and 4.

Table A2: Performance of variants of ULTRA. ULTRA-base manages to improve recall by over-generating candidate answers, while the over-generation problem is redressed by self-refinement (layer-2) through pairwise comparison. ULTRA-base does not confront the over-generation issue, thus, layer-2 does not contribute positively to the performance as in ULTRA-base. Best results are **bold**.

Table A3: Instructions designed for each stage of ULTRA. The document-level extractor is utilized in the ensembling mode of ULTRA, and serves as the Flan-UL2 baseline. The *aligned instruction* is adapted from from task 179 (participant extraction) in NIv212 (Wang et al., 2022).

Table A4: Designed template-based judgments used to train the LEAFER module in order to correct boundary identification. We categorize the (extraction, ground truth) pairs into six classes. Here, “anything” refers to a generated extraction that is completely off, and “[GT]” acts as a placeholder to be replaced with a specific ground-truth argument.