Leveraging ChatGPT in Pharmacovigilance Event Extraction: An Empirical Study

Anonymous ACL submission

Abstract

With the advent of large language models (LLMs), there has been growing interest in exploring their potential for medical applications. This research aims to investigate the ability of LLMs, specifically ChatGPT, in the context of pharmacovigilance event extraction, of which the main goal is to identify and extract adverse events or potential therapeutic events from textual medical sources. We conduct extensive experiments to assess the performance of ChatGPT in the pharmacovigilance event extraction task, employing various prompts and demonstration selection strategies. The findings demonstrate that while ChatGPT demonstrates reasonable performance with appropriate demonstration selection strategies, it still falls short compared to fully fine-tuned small models. Additionally, we explore the potential of leveraging ChatGPT for data augmentation. However, our investigation reveals that the inclusion of synthesized data into fine-tuning may lead to a decrease in performance, possibly attributed to noise in the ChatGPT-generated labels. To mitigate this, we explore different filtering strategies and find that, with the proper approach, more stable performance can be achieved, although constant improvement remains elusive.

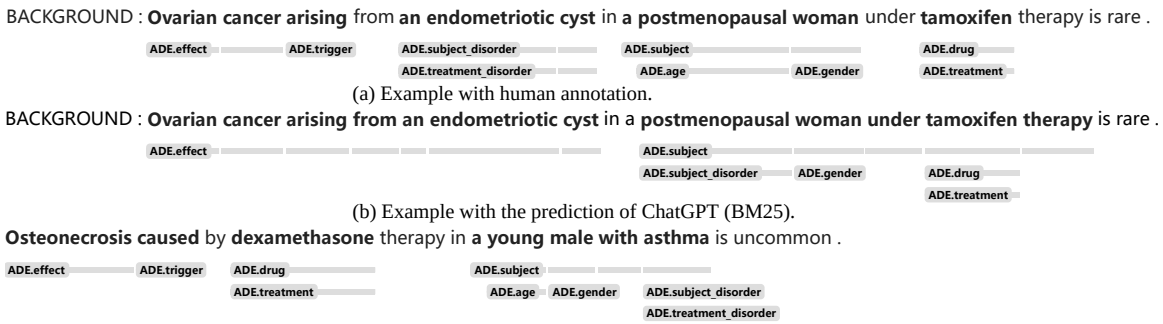
1 Introduction

Pharmacovigilance stands as a pivotal discipline in healthcare that encompasses a range of processes: identifying, evaluating, understanding, and preventing adverse effects and other medicine-related issues (World Health Organization, 2004). Within this domain, pharmacovigilance event extraction emerges as a crucial practice aimed at extracting structured medication-related event data from medical text sources, serving as valuable inputs for automatic drug safety signal detection. With the rapid expansion of electronic health records (EHR), medical case reports, and other textual resources, the need for efficient and accurate pharmacovigilance event extraction has become increasingly pressing.

Studies have been conducted to extract pharmacovigilance-related information from text data. However, previous research mainly focused on simple tasks such as entity extraction (Wunnava et al., 2017) or binary relation extraction (Gurulingappa et al., 2012; El-allaly et al., 2021). Recently, Sun et al. (2022) introduced a novel dataset for pharmacovigilance event extraction, which includes hierarchical annotations of adverse events and potential therapeutic events, capturing information about the subject, treatment, and effect. Additionally, they investigate the performance of various models, including sequence labelling and QA-based approaches, for this task, providing a foundation for further advancements in extracting structured event data for pharmacovigilance research.

The rise of large language models (LLMs), especially ChatGPT (OpenAI, 2022), has sparked considerable interest in their potential applications in the medical field (Agrawal et al., 2022; Kung et al., 2023). In this study, our focus is on exploring different ways to incorporate ChatGPT into the pharmacovigilance event extraction task. Figure 1(a) presents an example of this task.

We explore various strategies for prompting and demonstration selection to assess ChatGPT's performance in zero-shot and few-shot scenarios, comparing it with smaller fine-tuned models. Our findings indicate that, with suitable demonstrations, ChatGPT performs reasonably well but still falls short of the performance achieved by fully fine-tuned smaller models, as demonstrated in Figure 1(b). Furthermore, we delve into the utilization of ChatGPT for data augmentation, employing it to generate sentences structurally resembling demonstration samples (see Figure 1(c) for example). However, simply combining these generated samples with the training set leads to an overall performance decrease, possibly suggesting the sensitivity of data quality for fine-tuning methods. To address



(a) Example with human annotation.

(b) Example with the prediction of ChatGPT (BM25).

(c) ChatGPT-synthesized case using the example in (a) for demonstration.

Figure 1: Snippets from biomedical documents: a comparison of human annotations, ChatGPT predictions, and a ChatGPT-synthesized case.

this issue, we implement a filtering strategy for the augmented data, which leads to a performance improvement, bringing it closer to the levels achieved with the original training data but with reduced variance. This shows enhanced stability when working with ample high-quality data.

2 Prompt-based Learning with ChatGPT

2.1 Zero-shot Prompting

For zero-shot prompting, a manually designed instruction is employed to query ChatGPT for answers. In this study, we devise four approaches to prompt the model: a) **Schema:** providing instructions alongside enumeration of event types and argument types; b) **Explanation:** providing instructions with a detailed explanation of the schema; c) **Code:** formulating instructions and output format using a combination of text descriptions and code snippets; d) **Pipeline:** querying the model in a pipeline manner, which first prompts for the main arguments and then follows up with type-related questions for each sub-argument. Details of the prompts are presented in Appendix F.

2.2 Few-shot In-context Learning

For few-shot in-context learning, several demonstrations are provided together with the instruction. The selection of different demonstration examples can yield varying results. We explore different strategies for choosing in-context examples based on a given test instance, including: a) **Random:** randomly selecting examples from the training set; b) **SBERT:** choosing examples based on the similarity of their dense representations to the test sentence. We utilize Sentence-BERT(Reimers and Gurevych, 2019) to obtain the sentence representations; c) **BM25:** selecting examples based on the similarity of their lexical representations to the test sentence. We employ BM25 (Trotman et al., 2014) as the ranking function; d) **TreeKernel:** choosing examples based on the structural similarity to the test sentence. We implement the tree kernel by computing the Jaccard similarity of the subpaths within the dependency trees of the sentences.

3 ChatGPT as Data Synthesizer

We explore the potential of leveraging ChatGPT for data augmentation purposes. To achieve this, we incorporate an example from the training set, along with its annotated events, as input to ChatGPT. We then prompt ChatGPT to generate a sentence that exhibits a similar event structure to the given sentence and extract the events from the generated sentence. However, based on our initial study, we observed that ChatGPT tends to miss specific mentions of drugs or excessively use certain drugs, such as '*ibuprofen*'. We address this issue by restricting the inclusion of drug names and their corresponding effects sampled from the training data in generated sentences. Details of the prompt for data synthesizing are shown in Appendix F.

Recognizing that directly incorporating generated samples into the training data can lead to performance decline, possibly due to issues related to data quality, we have introduced filtering strategies. These include: a) **Train Filter:** Filtering the training set with $s_{gold} < \text{mean}(s_{gold})$, where s_{gold} is the average token probability given by the fine-tuned model on the ground-truth event label sequence; b) **Augment Filter:** Filtering augmented data with $z(s_{gold}) < 0$ or $z(s_{gold}) < z(s_{pred})$, where s_{pred} is the average token probability for predicted event label sequences. $z(s) = (s - \text{mean}(s^v))/\text{std}(s^v)$, and s^v represents the values of s in the validation set. The rationale behind this filter is to retain cases with annotations that are more certain than the predictions and have an above-average annotation.

With these filtering rules, we compare the model's performance on several data settings, including: training data (Tr.), training data combined

with augmented data (Tr.+Aug.), filtered training data (Tr. Fil.), training data with filtered augmented data (Tr.+Aug. Fil.) and filtered training data with filtered augmented data (Tr. Fil.+Aug. Fil.).

4 Experiments

4.1 Experimental Settings

Dataset We conducted experiments on the PHEE dataset (Sun et al., 2022), an English event extraction dataset sourced from publicly accessible medical reports, encompassing annotations for two event categories: *adverse events* and *potential therapeutic events*. The annotations follow a hierarchical structure, with main arguments providing information on the *subject*, *treatment*, and *effect*, while sub-arguments offer more detailed information pertaining to the main arguments. However, during our analysis, we observed that certain argument types showed low consistency. To address this issue, we performed automatic and manual revisions on the *subject.disorder*, *time_elapsed*, and *duration* arguments. For further details, please refer to Appendix A.

Baselines We compare ChatGPT's performance with the best-performing Generative QA model proposed in (Sun et al., 2022) and two widely adopted seq-to-seq models: 1) **UIE** (Lu et al., 2022), a model that is specifically pre-trained on structured information extraction data; and 2) **Flan-T5** (Chung et al., 2022), a model trained on a diverse range of tasks using instructional prompts. For more information, see Appendix B.

Evaluation We follow Sun et al. (2022) to evaluate both exact matching F1 score (EM_F1) and token-level matching F1 score (Token_F1) for argument extraction. During our preliminary experiments, we observed that ChatGPT struggled to generate reasonable results for trigger extraction. Considering that even humans find trigger identification challenging, and that it doesn't significantly contribute to understanding pharmacovigilance events, we did not query ChatGPT for triggers, but we still ask ChatGPT to generate the event structure, enabling the differentiation of multiple events. For the trigger extraction results obtained from finetuning models, please check Appendix D.

We perform 5-fold cross-validation for finetuning and data augmentation experiments, while limiting ChatGPT-based zero-shot and few-shot learning to a single split due to cost-related reasons. For more details about the experimental setup, please refer to Appendix C.

4.2 Results and Discussion

	Main-arguments		Sub-arguments	
	EM_F1	Token_F1	EM_F1	Token_F1
Schema	30.31	47.41	22.50	26.51
Code	25.94	40.42	25.67	29.70
Explanation	**34.80**	**52.99**	**36.70**	**39.33**
Pipeline	32.57	49.41	27.79	33.80

Table 1: Argument extraction results for ChatGPT zero-shot prompting with different prompting strategies.

ChatGPT with Different Prompting Strategies Table 1 presents the argument extraction results for ChatGPT using different zero-shot prompting strategies. Providing only instructions yields unsatisfactory performance, but including a detailed explanation of the event schema leads to noticeable improvement, highlighting the importance of comprehensive guidance. Further human evaluation reveals that end-to-end generation tends to miss arguments, whereas the pipeline approach tends to generate numerous false positive cases. It is surprising that the model performs poorly on seemingly simple arguments such as '*population*', '*route*', and '*age*'. While providing explanations improves the performance of some arguments (e.g., '*route*' and '*age*'), all approaches still struggle with '*population*' extraction. This difficulty may due to the gap between the lexical meaning of the label '*population*' and the semantic meaning of the argument. Additionally, while the pipeline method has advantages in extracting certain argument types (e.g., '*gender*' and '*frequency*'), the inference time is proportional to the number of argument types, making it approximately 10 times longer than the end-to-end methods.

Table 2 displays the few-shot argument extraction results for ChatGPT using various in-context selection strategies. Dense representation-based demonstration retrieval with SBERT does not demonstrate superiority in this task, possibly due to limited domain knowledge captured by the pre-trained sentence representation model. Incorporating structured information improves performance, while the simplest lexical-based retrieval strategy shows the most noticeable performance gains. Upon examining the samples retrieved by different example selection strategies, we observed that SBERT and TreeKernel tend to retrieve structurally similar sentences, while BM25 is more inclined to retrieve sentences containing matching entities such as drugs (since entities usually serve as keywords in a sentence). This observation sug-

gests that the superior performance of BM25 in argument extraction can be attributed to the fact that this task is more sensitive to entities. When more examples with similar entities are covered, ChatGPT learns more effectively from them.

	Main-arguments		Sub-arguments	
	EM_F1	Token_F1	EM_F1	Token_F1
random	58.31	72.74	60.32	63.74
SBERT	56.90	71.65	62.29	64.25
TreeKernel	**60.54**	73.68	63.36	64.69
BM25	60.39	**76.15**	**67.35**	**68.67**

Table 2: Argument extraction results for ChatGPT few-shot prompting with different in-context demonstration selection strategies (results for 5-shot are reported).

Finetuning Models vs. ChatGPT Table 3 illustrates the argument extraction results for different methods. The findings indicate that there is minimal variation among the fine-tuning methods. Specifically, the Flan-T5 model, despite not being explicitly pre-trained for the information extraction task, demonstrates slightly better performance than the UIE model. In contrast, ChatGPT without demonstrations exhibits poor performance. However, when demonstrations are provided, ChatGPT shows improved results, although there remains a noticeable gap compared to the fine-tuning methods. For a detailed breakdown of the results for each argument type, refer to Appendix E.

	Main-arguments		Sub-arguments	
	EM_F1	Token_F1	EM_F1	Token_F1
Fully supervised				
Generative QA	68.85	81.63	77.33	78.83
UIE(Large)	$69.46_{\pm.49}$	$81.20_{\pm.40}$	$77.12_{\pm1.3}$	$78.83_{\pm1.4}$
Flan-T5(Large)	$\mathbf{70.78}_{\pm1.4}$	$\mathbf{82.34}_{\pm1.5}$	$\mathbf{77.63}_{\pm.1.6}$	$\mathbf{79.52}_{\pm1.3}$
Zero-Shot				
ChatGPT(Exp.)	34.80	52.99	36.70	39.33
Few-Shot				
ChatGPT(BM25)	60.39	76.15	67.35	68.67

Table 3: Argument extraction results for various methods. For fine-tuning methods, we report the $mean_{\pm std}$ value of 5-fold cross-validation. For ChatGPT(BM25), we provide the results for 5-shot. We obtain Generative QA results directly from the original paper.

Data Augmentation with ChatGPT Table 4 presents the performance of Flan-T5 when augmentation and various filtering strategies are employed. It can be seen that simply extending the training data with ChatGPT-synthesized cases could lead to an obvious performance drop. In contrast, with the filtered training set, although retained only 65% of training data, surpasses results obtained from over 5,000 augmented instances, which may indicate the critical role of data quality in pharmacovigilance event extraction. Furthermore, training with filtered augmented data effectively restores performance to the original level. In particular, training with both filtered training data and filtered augmented data displays only slight deviations from training with the original data, yet it notably reduces variance. The p-values for the variance difference significance, assessed through the F-test, are 0.29 and 0.39 for EM_F1 and Token_F1, respectively.

	EM_F1	Token_F1	Avg. Cases
Tr.	$\mathbf{74.45}_{\pm1.46}$	$\mathbf{81.30}_{\pm1.27}$	2897
Tr.+Aug.	$73.07_{\pm0.92}$	$79.93_{\pm1.51}$	5446
Tr. Fil.	$73.92_{\pm.1.28}$	$80.71_{\pm1.60}$	1873
Tr.+Aug. Fil.	$74.26_{\pm.1.27}$	$80.98_{\pm2.06}$	3702
Tr. Fil.+Aug. Fil.	$74.19_{\pm1.09}$	$81.05_{\pm1.09}$	2678

Table 4: Argument (including main and sub-arguments) extraction results for Flan-T5 (Large) with augmentation and filtering strategies. The *Avg. Cases* column displays the average number of training cases over 5 folds.

During our manual examination, we identified a correlation between the model's predicted probability of generated label sequences (s_{gold}) and the label quality. Those cases with a lower s_{gold} often exhibited lower quality, frequently missing important arguments. Cases with higher s_{gold} demonstrated higher quality, but occasionally were still not more accurate than the model's prediction. To address this, we introduced a metric using the z-scores of s_{gold} and s_{pred} to compare the quality of annotations and model predictions. We retained cases with better annotation quality compared to predictions. Consequently, we observe that the performance of certain argument types, such as '*time_elapsed*' and '*duration*', which sharply decreased following augmentation, evidently rebounded after training with filtered data.

5 Conclusion

This paper provides empirical practice in various approaches to leveraging ChatGPT for the pharmacovigilance event extraction task. Overall, ChatGPT exhibits impressive few-shot learning capabilities in pharmacovigilance event extraction. Nevertheless, considering the sensitivity of the medical field, fine-tuned models retain a clear edge in the presence of abundant data. Properly filtered synthetic data could enhance stability, but the quest for more effective methods to augment and enhance overall performance remains for future exploration.

Limitations

In our preliminary study, we encountered limitations in exploring alternative open-source LLMs, such as LLaMA 30B (Touvron et al., 2023) and Flan-T5 XXL (Chung et al., 2022), for zero-shot/few-shot prompting. These models exhibited significant differences in generation quality compared to ChatGPT, and their slow inference speeds hindered a comprehensive evaluation. Despite these limitations, we highlight the importance of further research to investigate the potential of leveraging different open-source LLMs.

Secondly, our investigation focused solely on unsupervised methods for in-context demonstration selection. Future research could explore the incorporation of annotations in the selection process, which may yield valuable insights and improve the performance of ChatGPT in pharmacovigilance event extraction.

Furthermore, it is worth noting that the use of ChatGPT-synthesized data may alter the data distribution, potentially introducing incorrect knowledge about adverse events. While the impact of this approach on event extraction may be limited because the extraction results are constrained by the given sentence, caution is needed when applying the methods described in the text to other application domains, such as ADE generation, due to potential risks.

References

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.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2022. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*.

Ed-drissiya El-allaly, Mourad Sarrouti, Noureddine En-Nahnahi, and Said Ouatik El Alaoui. 2021. Mttlade: A multi-task transfer learning-based method for adverse drug events extraction. *Information Processing & Management*, 58(3):102473.

Harsha Gurulingappa, Abdul Mateen Rajput, Angus Roberts, Juliane Fluck, Martin Hofmann-Apitius, and Luca Toldo. 2012. Development of a benchmark corpus to support the automatic extraction of drug-related adverse effects from medical case reports. *Journal of biomedical informatics*, 45(5):885–892.

Tiffany Kung, Morgan Cheatham, Arielle Medenilla, Czarina Sillos, Lorie Leon, Camille Elepaño, Maria Madriaga, Rimel Aggabao, Giezel Diaz-Candido, James Maningo, and Victor Tseng. 2023. Performance of chatgpt on usmle: Potential for ai-assisted medical education using large language models. *PLOS Digital Health*, 2:e0000198.

Yaojie Lu, Qing Liu, Dai Dai, Xinyan Xiao, Hongyu Lin, Xianpei Han, Le Sun, and Hua Wu. 2022. Unified structure generation for universal information extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5755–5772, Dublin, Ireland. Association for Computational Linguistics.

OpenAI. 2022. Chatgpt.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Zhaoyue Sun, Jiazheng Li, Gabriele Pergola, Byron Wallace, Bino John, Nigel Greene, Joseph Kim, and Yulan He. 2022. PHEE: A dataset for pharmacovigilance event extraction from text. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 5571–5587, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Andrew Trotman, Antti Puurula, and Blake Burgess. 2014. Improvements to bm25 and language models examined. In *Proceedings of the 2014 Australasian Document Computing Symposium*, pages 58–65.

World Health Organization. 2004. Pharmacovigilance: ensuring the safe use of medicines. Technical report, World Health Organization.

Susmitha Wunnava, Xiao Qin, Tabassum Kakar, Vimig Socrates, Amber Wallace, and Elke Rundensteiner. 2017. Towards transforming fda adverse event narratives into actionable structured data for improved pharmacovigilance. In *Proceedings of the Symposium on Applied Computing*, pages 777–782.

A Data Annotation Revision Details

In the original dataset, we observed particularly low levels of annotation inconsistency for '*subject.disorder*', '*time_elapsed*', and '*duration*' arguments, as illustrated by the examples provided in Table A1. To address this, we conducted an automatic revision for the '*subject.disorder*' annotation and hired annotators to manually correct the '*time_elapsed*' and '*duration*' annotations. For '*subject.disorder*' correction, if a '*treatment.disorder*' was present in the '*subject*' argument but not annotated as '*subject.disorder*', we added it to the '*subject.disorder*' annotation. For '*time_elapsed*' and '*duration*' correction, detailed guidelines were provided to the annotators to ensure consistent annotations. We employed three annotators and informed them about the purpose of the data. The annotators are all PhD students who volunteered for this task, receiving compensation through the university's payment platform for their annotation work. Two of the annotators have a background in computer science, and one annotator has a medical background. Two of the annotators are non-native English speakers, and one is a native English speaker. Following the approach used in (Sun et al., 2022), we evaluated the consistency among the annotators using the EM_F1 score. The averaged EM_F1 scores for both '*time_elapsed*' and '*duration*' annotations were 75.3%.

B Details of Baseline Implementation

For the implementation of seq-to-seq baselines, we formulate pharmacovigilance event extraction as a conditional text generation task. Concretely, given a sentence x and additional auxiliary information a, the model is trained to generate a linearized sequence y representing the output event structure.

For UIE, we refer to the methodology outlined in the original paper by utilizing the Structural Schema Instructor (SSI) as the auxiliary information a and constructing the target sequence y with Structural Extraction Language (SEL). However, special tokens used in SSI and SEL in UIE can result in a decrease in performance if no external pre-training is applied. Thus for Flan-T5, we substitute the SSI with a concise instruction accompanied by a natural language enumeration of the schema. Additionally, for the target sequence construction, we utilize square brackets as the structural symbol.

For both UIE and FLan-T5, we use the large model which comprises 770M parameters. Training an epoch typically takes around 2 minutes, and validation, which utilizes beam search, requires approximately 10 minutes with an NVIDIA A100 (80G) GPU. The fine-tuning models generally converge within 10 epochs.

C Details of Experimental Setup

C.1 Few-shot Prompting Settings

In the context of event extraction, each shot includes one example for each event type. In Section 4, we report the 5-shot results for in-context demonstration selection strategies, which entails providing a total of 10 examples for each instance. The selection of the number of demonstration cases was based on ChatGPT's input length capacity.

We further evaluate the argument extraction performance of several in-context demonstration selection strategies when different numbers of demonstration examples are selected in Figure A1. Notably, when the first example is added, all methods experience a significant performance boost. However, as the number of examples increases, the performance gains become more minimal. Five-shot prompting (involving 5 ADE examples and 5 PTE examples) has approached the maximum input limit that ChatGPT can handle. Nevertheless, we reasonably suspect that further increasing the number of examples would not get significant performance improvements.

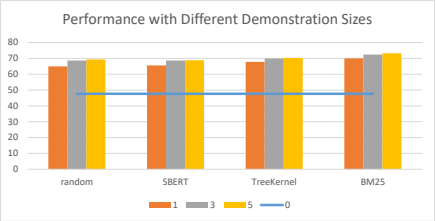


Figure A1: Token_F1 scores for argument extraction with different demonstration sizes. The blue line represents the performance of zero-shot prompting with the explained schema.

C.2 Hyperparameter Details

The order and occurrence of events and arguments in the generated sequence can impact the learning effectiveness of the model. To tackle this, Lu et al. (2022) introduced the 'Rejection Mechanism', which generates a null span when a specific type of event or argument is absent in the sentence. In our preliminary experiments, we determined that the noise injection ratio has little impact on the performance but the order of the argument generation matters. Therefore, we choose to set the noise in-

Subject.Disorder: We report two patients with acne vulgaris with a fourth type of minocycline-induced cutaneous pigmentation. We observed that when a disorder span is included in a 'subject' argument and also as 'treatment.disorder', annotations in the original dataset show inconsistency on whether to annotate this span as 'subject.disorder'.
Time_elapsed & Duration: In this article, we describe another case of subcutaneous changes following repeated glatiramer acetate injection, presented as localized panniculitis in the area around the injection sites, in a 46-year-old female patient who was treated with glatiramer acetate for 18 months. Annotation inconsistencies arise when a 'time_elapsed' argument can also be described as 'duration'.

Table A1: Inconsistent examples from the PHEE dataset.

jection ratio to 0 and keep the arguments generated in order to reduce the fluctuation caused by random insertion during model comparison.

To fine-tune the models, we establish a maximum length of 512 tokens for both input and output. We utilize a total batch size of 32 for the large model, and 64 for the base model. The learning rates are configured as 3e-4 for the large model and 5e-4 for the base model, with a warm-up ratio of 0.06. We train the models for a maximum of 50 epochs, early stopping if there is no improvement for 5 epochs. During the generation process, we employ beam search with a beam size of 3.

We employ the 'gpt-3.5-turbo-0301' version of ChatGPT for prompting-based event extraction and synthesized data generation. The temperature is set as 0 for zero-shot and few-shot prompting, and 0.2 for data generation.

D Trigger Extraction Results for Finetuning Methods

Table A2 displays the results of trigger extraction and event type classification for the fine-tuning models. In general, there is little difference in the performance of trigger extraction and event type classification between different models. Furthermore, training with filtered training and augmented data still exhibits the smallest variance, which is consistent with the observation for argument extraction.

	Trigger	Event Type
UIE(Large)	**69.92** $_{\pm 1.72}$	94.78$_{\pm .72}$
Flan-T5(Large)	69.60$_{\pm 1.87}$	95.04$_{\pm .97}$
w/ Tr.+Aug.	68.46$_{\pm 1.83}$	94.92$_{\pm .60}$
w/ Tr. Fil.	69.50$_{\pm 1.61}$	94.92$_{\pm .88}$
w/ Tr.+Aug. Fil.	69.68$_{\pm 1.36}$	95.00$_{\pm .79}$
w/ Tr. Fil.+Aug. Fil.	69.73$_{\pm 1.14}$	**95.13**$_{\pm .48}$

Table A2: Results for trigger extraction (EM_F1) and event type classification (F1).

E Argument Extraction Results for Each Argument Type

Table A3 provides a detailed overview of argument extraction results for Flan-T5 with two augmentation strategies and ChatGPT. In comparison, ChatGPT exhibits a specific vulnerability in accurately matching main arguments, likely attributed to their greater length, which poses challenges in precise boundary determination. When it comes to sub-arguments, ChatGPT demonstrates a performance distribution similar to fine-tuning models but achieves lower overall scores. Notably, for certain argument types of which ChatGPT performs notably worse, such as '*frequency*' and '*duration*', these shortcomings also negatively impact the performance when training with ChatGPT-generated data. However, after filtering, the performance on these argument types can be improved to the extent that they may even outperform fine-tuning with annotated training data alone.

F Prompt Details

Table A4 shows the instructions utilized for ChatGPT's zero-shot prompting. Through our preliminary experiments, we discovered that ChatGPT exhibits better performance when tasked with generating structured output in JSON format rather than textual output. Based on this finding, we explore additional possibilities. For the end-to-end generation approach, we experiment with modifying the instructions to a code style or providing a detailed explanation of the schema. In the case of pipeline prompting, we initially prompt ChatGPT to generate the skeleton of the output, encompassing multiple events in a competent manner. Subsequently, in the second stage, we provide the generation from the first stage and ask specific questions for each sub-argument type.

Table A5 presents the prompt employed to query ChatGPT for the generation of synthesized instances for examples with adverse events. We employ a similar prompt for the data generation of

	Flan-T5		Flan-T5 (Tr.+Aug.)		Flan-T5 (Tr. Fil.+Aug. Fil.)		ChatGPT	
	EM_F1	Token_F1	EM_F1	Token_F1	EM_F1	Token_F1	EM_F1	Token_F1
Subject	**73.11**	**82.37**	70.93	80.90	72.39	82.15	57.96	75.20
Age	**88.12**	92.07	87.21	92.55	87.50	**92.82**	86.62	90.18
Disorder	**69.80**	77.13	63.81	72.76	69.73	**77.45**	53.90	61.08
Gender	86.73	86.51	86.03	85.78	**87.15**	**87.00**	84.29	85.07
Population	74.83	75.72	72.30	73.94	**75.90**	**76.69**	49.30	42.11
Race	93.20	**93.35**	**93.29**	91.20	92.02	91.52	87.5	77.78
Treatment	**66.35**	**79.82**	66.27	79.00	65.90	79.68	57.67	73.49
Drug	**87.03**	**88.32**	85.84	87.45	86.65	87.99	80.78	82.59
Disorder	**67.19**	**73.14**	65.24	71.73	66.64	72.57	55.89	62.01
Route	**67.76**	69.34	63.55	65.55	66.37	**70.39**	56.66	63.73
Dosage	**65.95**	**76.40**	63.58	72.17	62.91	73.16	47.11	61.05
Time elapsed	61.56	71.21	54.11	61.25	**62.09**	**71.98**	40.68	51.67
Duration	60.40	**64.91**	56.12	60.42	**61.47**	58.77	47.56	56.58
Frequency	51.26	**54.37**	43.43	46.19	**53.25**	52.10	36.36	33.09
Combination.Drug	**69.77**	**71.18**	66.87	68.93	69.34	70.90	60.79	62.90
Effect	74.33	**84.73**	74.68	83.94	**74.75**	84.65	64.60	79.19

Table A3: Argument extraction results for each argument type. To accommodate space limitations, we showcase results for Flan-T5 with two augmentation strategies and ChatGPT. The Flan-T5 results represent the average score across 5-fold cross-validation, while the ChatGPT results showcase the performance of the 5-shot BM25 approach.

cases with potential therapeutic events and multiple events. Differently, we apply only the drug constraint to instances related to potential therapeutic events, as these typically do not involve a relevant effect. In addition, we refrain from imposing such constraints on multi-event instances, as doing so may complicate the preservation of event structure in synthesized samples.

G Licenses

The PHEE dataset employed in this study is subject to the MIT License. The UIE model is covered by the Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International Public License. The Flan-T5 model under the Apache License 2.0, and ChatGPT is a commercial service for which we adhere to OpenAI's terms of use. We use the dataset and tools within the scope of their intended use.

Prompting Strategy	Example
Schema	Extract event information from the following sentence and return events in json format as this: [{"event_type": event type, "arguments":[{"argument_type": argument type, "argument_span":argument extraction}]}]. Event type: adverse event, potential therapeutic event. Argument type: subject, age, gender, race, population, subject_disorder, treatment, drug, dosage, route, duration, frequency, time_elapsed, indication, combination_drug, effect. Sentence: <SENTENCE> Output:
Code	Argument = {"argument_type": str, #options: [subject, age, gender,race, population, subject_disorder, treatment, drug, dosage, route, duration, frequency, time_elapsed, indication, combination_drug, effect] "argument_span": str,} Event ={"event_type": str, #options: [adverse_event, potential_therapeutic_event] "arguments": List[Argument],} events: List[Event] = extract events in the sentence: <SENTENCE> print(json.dumps(events))
Explanation	Extract event information from the following sentence and return events in json format as this: [{"event_type": event type, "arguments":[{"argument_type": argument type, "argument_span":argument extraction}]}]. Event type: adverse event (an event shows the use of a drug or combination of drugs cause a harmful effect on the human patient), potential therapeutic event (an event shows the use of a drug or combination of drugs bring a potential beneficial effect on the human patient). Argument type: subject (overall description of the patients involved in the event), age (the concrete age or an age range of the subject), gender (the subject's gender), race (the subject's race or nationality), population (the number of patients receiving the treatment), subject_disorder (the subject's disorders), treatment (overall description of the therapy administered to the patients), drug (the drugs used as therapy in the event), dosage (the amount of the drug is given), route (the route of the drug administration), duration (how long the patient has been taking the medicine), frequency (the frequency of drug use), time_elapsed (the time elapsed after the drug was administered to the occurrence of the side effect), indication (the target disorder of the medicine administration), combination_drug (the drugs used in combination), effect (the side effect in the adverse event or the beneficial effect in the potential therapeutic event). Sentence: <SENTENCE> Output:

Pipeline	**Stage 1**: Extract adverse events and potential therapeutic events in the sentence, as well as the information about the subject (the patient), the treatment and the effect of the treatment involved in the event. Return the output in json format as this: [{"event_type": event type, "subject": span of subject information, "treatment": span of treatment information, "effect": span of effect information}]. Event type: adverse event, potential therapeutic event. Sentence: <SENTENCE> Output: **Stage 2**: Answer the question related to the given sentence and given event information. The answer should be a span exactly extracted from the sentence. If no answer can be found from the sentence, return N/A. Sentence: <SENTENCE> Event: Event type: <EVENT_TYPE> Subject: <SUBJECT> Treatment: <TREATMENT> Effect: <EFFECT>. <QUESTION> **Questions for each sub-argument type:** *age*: What's the age of the subject? *gender*: What's the gender of the subject? *race*: What's the race or the nationality of the subject? *population*: How many subjects are involved in the event? *subject_disorder*: What disorders do the subjects suffer from? *drug*: What drugs are administered to the subject? *dosage*: What amount of the drug is administered to the subject? *route*: What route is the drug given to the subject? *duration*: How long have the subject been taking the drug until the event occurred? *frequency*: How frequently does the subject take the drug? *time_elapsed*: How long has elapsed since the patient started or ended dosing until the event occurred? *indication*: What's the target disease of the treatment? *combination_drug*: What drugs are used in combination in the event

Table A4: Instructions for zero-shot prompting. <SENTENCE> is replaced with the query sentence. In the second stage of the pipeline prompting, <EVENT_TYPE>, <SUBJECT>, <TREATMENT>, <EFFECT> are replaced with the generated results from the first stage, and <QUESTION> is replaced with manually crafted questions for each argument type. To enhance clarity, we substitute the argument type 'treatment_disorder' in the dataset with 'indication' when querying ChatGPT.

Sentence: <SENTENCE> The events involved in the sentence are: <OUTPUT> Event type: adverse event (an event shows the use of a drug or combination of drugs cause a harmful effect on the human patient), potential therapeutic event (an event shows the use of a drug or combination of drugs bring a potential beneficial effect on the human patient). Argument type: subject (overall description of the patients involved in the event), age (the concrete age or an age range of the subject), gender (the subject's gender), race (the subject's race or nationality), population (the number of patients receiving the treatment), subject_disorder (the subject's disorders), treatment (overall description of the therapy administered to the patients), drug (the drugs used as therapy in the event), dosage (the amount of the drug is given), route (the route of the drug administration), duration (how long the patient has been taking the medicine), frequency (the frequency of drug use), time_elapsed (the time elapsed after the drug was administered to the occurrence of the side effect), indication (the target disorder of the medicine administration), combination_drug (the drugs used in combination), effect (the side effect in the adverse event or the beneficial effect in the potential therapeutic event). Generate a sentence with an adverse event which has a similar structure as the given sentence, and extract the events in the generated sentence. The drug <CONST_DRUG> must appear in the event, and the effect should be <CONST_EFFECT>. Return in the following json format: {"sentence":the generated sentence, "output": [{"event_type": event type, "event_trigger": the token indicating the existence of the event, "arguments":[{"argument_type": argument type, "argument_span":argument extraction}]}]}. Return the json output only.

Table A5: The prompt used to query ChatGPT for generating synthesized instances for ADE cases, with <SENTENCE> representing an example sentence from the training set, <OUTPUT> representing the annotation of the example sentence, <CONST_DRUG> and <CONST_EFFECT> representing a pair of sampled drug and effect from the training set.