Distilling Event Sequence Knowledge From Large Language Models

Somin Wadhwa[♢*] **Oktie Hassanzadeh**[♣] **Debarun Bhattacharjya**[♣]
Ken Barker[♢] **Jian Ni**[♣]
[♢]Northeastern University
[♣] IBM Research
{wadhwa.s}@northeastern.edu
{hassanzadeh, debarunb, kjbarker}@us.ibm.com

Abstract

Event sequence models have been found to be highly effective in the analysis and prediction of events. Building such models requires availability of abundant high-quality event sequence data. In certain applications, however, clean structured event sequences are not available, and automated sequence extraction results in data that is too noisy and incomplete. In this work, we explore the use of Large Language Models (LLMs) to generate event sequences that can effectively be used for probabilistic event model construction. This can be viewed as a mechanism of distilling event sequence knowledge from LLMs. Our approach relies on a Knowledge Graph (KG) of event concepts with partial causal relations to guide the generative language model for causal event sequence generation. We show that our approach can generate high-quality event sequences, filling a knowledge gap in the input KG. Furthermore, we explore how the generated sequences can be leveraged to discover useful and more complex structured knowledge from pattern mining and probabilistic event models. We release our sequence generation code and evaluation framework, as well as corpus of event sequence data.

1 Introduction

Building probabilistic models from event sequence data has numerous practical applications across different domains when plentiful high-quality event data is available. For example, in Finance, event models can be used to predict stock market trends and make informed investment decisions. In Healthcare, event models can help identify patterns and correlations in patient data to improve diagnoses and treatment plans. In the field of Cybersecurity, these models can be used to detect and prevent potential cyber attacks by analyzing the sequence of events leading up to a breach. A common characteristic of the data in these domains is that sequences are clearly associated with an entity (e.g., a company, a person, or a device). There are however other domains where such a clean association between events and entities may not be possible. One such application is news event analysis (Cekinel and Karagoz, 2022; Du et al., 2022; Hassanzadeh et al., 2022; Radinsky et al., 2012). While various news sources record and describe newsworthy events, it is often not possible to automatically put together coherent sequences of events, because each event may involve multiple topics and actors, and many correlated and unrelated events may be occurring simultaneously or in close proximity

Prior work has addressed this challenge by devising automated mechanisms for extracting narratives (Norambuena et al., 2023; Santana et al., 2023), topic detection and tracking (Allan, 2012), and timeline summarization (Gholipour Ghalandari and Ifrim, 2020). While these different categories of solutions have been successful in a range of applications, the outcome is inherently noisy and not in the form of structured event sequences useful for the construction of event models.

Large Language Models (Brown et al., 2020; Raffel et al., 2020; Wei et al., 2022) have recently become the dominant paradigm in a range of natural language processing (NLP) tasks (Chung et al., 2022) and often beat traditional approaches on a number of challenging tasks, including complex arithmetic reasoning (Imani et al., 2023) and open-domain question answering (Kamalloo et al., 2023). In this paper, our goal is to examine the capability of LLMs to generate structured event sequences useful for event analysis. Our hypothesis is that LLMs trained on large corpora have already gathered the required knowledge of plausible event sequences and therefore can be suitably guided to produce diverse and high-quality event sequences. To effectively distill this knowledge, we use event-related concepts in Wikidata (Vrandečić and Krötzsch, 2014), a comprehensive general-

[*]Work perfomed during internship at IBM Research.

domain knowledge graph, to guide the sequence generation. This can be viewed as a novel mechanism for knowledge-guided text generation (Yu et al., 2022) and symbolic knowledge distillation (West et al., 2022). We then use these generated sequences for pattern mining and learning probabilistic event models, as a way of further structuring the underlying knowledge. Figure 1 shows our overall framework along with examples from our experiments of patterns mined from an LLM-generated event sequence collection, as a well as a simple model learned from the collection.

In summary, we make the following contributions:

1. We devise a new iterative in-context prompting strategy for generating high-quality event sequences using LLMs. To the best of our knowledge, we are the first to use LLMs to generate structured event sequences for the purpose of analyzing various event models.

2. We compile high-quality event sequences using our generation mechanism, based on a curated set of high-level event concepts (classes) from Wikidata that represent newsworthy events.

3. We develop an evaluation framework and conduct experiments to show the value of LLM-powered event sequence generation on replicating and augmenting knowledge in structured representations such as knowledge graphs.

4. We further demonstrate the practical usefulness of our approach by leveraging downstream pattern mining and probabilistic event models.

Our code and generated sequence data are included in the supplementary material, and will be made publicly available for future research.

2 Knowledge-Guided Event Sequence Generation

Through utilizing LLMs, we model event prediction as a *conditional generation task* under zero and iterative few-shot settings. Our targets are linearized sequences of event concepts. We begin by prompting a large language model (with in-context exemplars) with an event trigger y_1 to generate the next concept from a defined set of labels, and repeat this process until we get a sequence of desired length. Formally, given an event trigger $\mathcal{T}$, we model the probability of generating linearized string y of length T containing N unique event concepts that follow $\mathcal{T}$ in sequence:

$$p_{\text{LM}}(y|\mathcal{T}) = \prod_{t=2}^{T} p(y_t|\mathcal{T}, y_{<t-1}) \quad (1)$$

This is the standard conditional language modeling objective. We try multiple prompting techniques and qualitatively observe optimal results with an *iterative* in-context few-shot prompting strategy (Figure 2). Specifically, we start with a set of six randomly selected examples of the form – "What usually follows event X?". This approach follows the incremental prompting procedure from (Li et al., 2023). Based on the model output (Y) from a pre-specified vocabulary, we append this same example to the original prompt *in conjunction* (i.e. "What usually follows event X and Y?") with ICL examples of the same form. As shown in Figure 2, our iterative technique serves dual purposes: (i) it leverages in-context learning; and (ii) eliminates the need for implementing complex resolvers to post-process model outputs. We repeat this process until a sequence of a desired minimum length m is achieved (in our experiments, $m = 3$) or we've exhausted a maximum number of tries ($k = 10$ in our experiments) to generate an in-domain event type.

Identifying Event Concepts With incremental prompting we curate a new dataset of high-level event concepts (classes) from Wikidata that represent newsworthy events. To do so, we query Wikidata for event concepts that have links to Wikinews articles and are instances of classes that are a subclass of the `occurrence` class, i.e., indicating they are newsworthy event classes. We gathered 50 top-level classes for these event concepts, each having multiple labels (e.g. `conflict` → `conflict (psychological)`, `dispute`, `disagreement`, `etc.`). This yielded a total of 202 unique event labels for the 50 top-level classes. Most of these event concepts have *some* causal relations (i.e. `has_cause` or `has_effect`). We use these relation pairs as in-context exemplars to create our prompts. We then generate event sequences through iterative in-context prompting (Figure 2). Full length prompts used in all our experiments are provided in the Appendix.

To generate new event sequences in a zero-shot

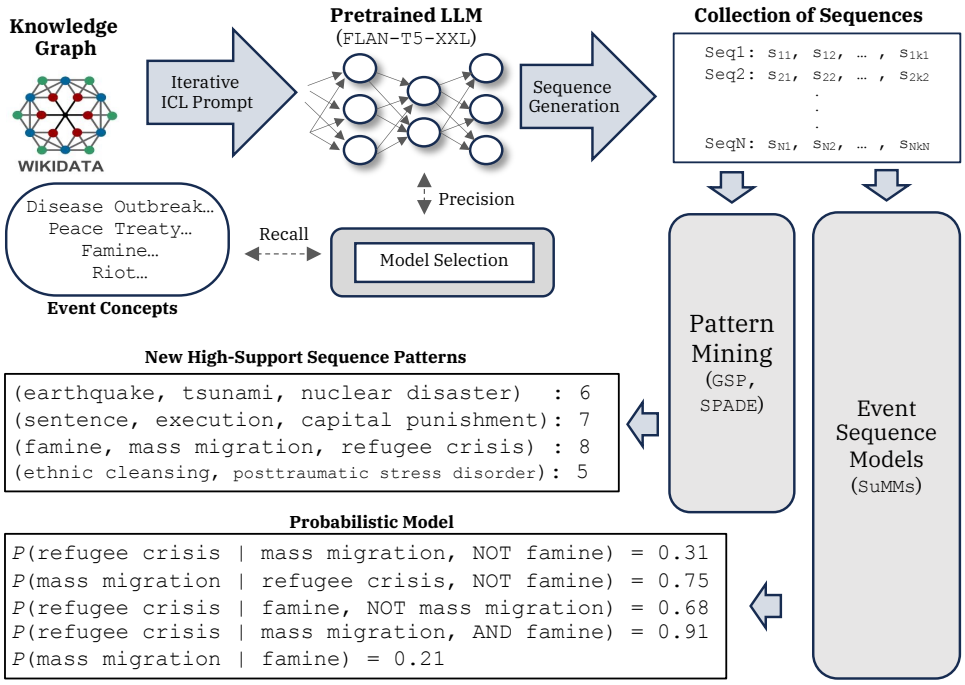

Figure 1: An overview of our framework for distilling event sequence knowledge from LLMs, along with examples portraying potential use cases. We show that by **(1)** starting with a sparse knowledge graph such as Wikidata, we can generate targeted event sequences. Owing to the inherent sparsity in the underlying KG, we can **(2)** use LLMs to carry out a portion of the evaluation (i.e. precision) to select an optimal model. On this new generated sequence dataset, we then **(3)** apply classical pattern-mining algorithms (e.g., GSP) to identify potentially interesting `has_cause` and `has_effect` event sequence chains, and **(4)** learn summary Markov models (SuMMs) to identify potential influencing events for particular event types of interest; these are both illustrations of extracting complex structured knowledge from the generated sequences.

setting, we start with an event trigger (e.g. a concept like `workplace accident`), create a prompt with instructions describing desired relationships and a few in-context exemplars (ICL prompt), and constrained the model output to the original 50 event concept labels to generate the next event in the sequence (full text of the prompt is provided in the supplementary material). We append the model output to another ICL prompt with *conjunctive* event examples (i.e. questions of the form "What happens after `X`, `Y`, and `Z`"). We repeat this process until we reach a pre-defined maximum sequence length (in this case, 10) or until the model fails to generate an in-vocabulary response in k maximum attempts (in this case, $k = 3$). In this process, we test the following two ablations –

Number of exemplars We varied the number of in-context exemplars between 1-12. We evaluated recall (i.e. proportion of references captured by the resulting output sequences) for all generated outputs in an automated way through matching lexical alignment with Wikidata references. We observed none to marginal improvements by varying the number of exemplars beyond 3 in the initial trigger prompt, and beyond 5 in the second iterative prompt.

Selection of specific exemplars Selecting in-context examples is an incredibly noisy process (Zhang et al., 2022). We started with a static set of randomly selected examples, however owing to Wikidata's inherent label imbalance (political and economic events dominate newsworthy concepts), this led to erratic results, i.e. high recall on similar concepts but not otherwise.

We then tested a dynamic selection method where every instance of the prompt would contain examples similar to `target_label`. To achieve this, we used BERT embeddings (Devlin et al., 2019) to retrieve (using cosine distance) examples from the reference set. This approach however led to a lower macro-recall and, upon manual inspection of outputs, we observed a high degree of redundancy, i.e. generated outputs were copied from the ICL exemplars.

To solve for these issues, we reverted to the static prompt examples but manually selected a mix of

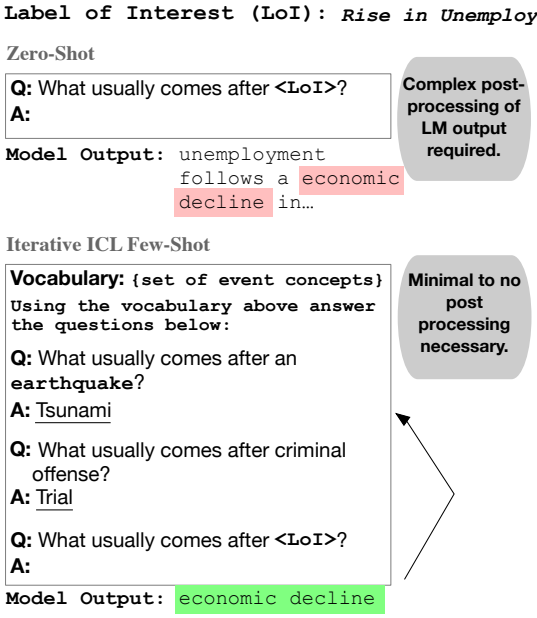

Figure 2: Illustration of our approach to elicit event sequence knowledge given a *label of interest*. Use of instructional in-context exemplars substantially reduces the need for post-processing LLM output in addition to constraining the output label space.

event topics to be included in the prompt. Our current selection of the prompt yields higher macro-recall than both of the aforementioned techniques tested. While we believe there may be better techniques to select ideal candidates for ICL exemplars (An et al., 2023; Agrawal et al., 2023) and understand how their compositionality affects specific outputs, we consider such an analysis to be beyond the scope of our work.

We repeat this sequence generation procedure on all 202 event concepts, generating $2,276$ event sequences with an average length of 5.7 labels per sequence.

3 Assessing the Quality of LLM-Generated Event Sequences

Open-ended text generation in a task like event sequencing, with extremely sparse reference data, poses unique challenges to the evaluation of model outputs (Wadhwa et al., 2023). The traditional scheme to evaluate discrete model outputs has been to calculate precision and recall for the generated outputs against a predefined reference test set. However, under open-world settings and particularly when a sparse KG like Wikidata is used as reference data, a missing causal relation between two event classes in a sequence may very well be a valid relation. Therefore, it is not possible to automatically derive an accurate measure of precision and recall purely using Wikidata as reference data. We take a multi-pronged evaluation approach to assess the quality and usefulness of the generated event sequences across multiple tasks. This section presents our approach in evaluating the quality of generated event sequences, along with the results of this evaluation. An evaluation of the usefulness of the generated event sequences is presented in Section 4.

3.1 Human Evaluation of Cause-Effect Prediction Accuracy of LLMs

To quantify how well model outputs correlate with human assessments, we first conduct a small-scale human evaluation on a different, independent event-commonsense reasoning dataset: ATOMIC (Hwang et al., 2021). ATOMIC consists of event-centered pairs of instances of the form `IsAfter` (`Y` comes after `X`) and `Causes` (`X` causes `Y`). We use these instances as input prompts to the model and then ask human annotators to evaluate model outputs. Specifically, we show human annotators anonymized model outputs and true references and elicit their preferences given a trigger event.

To generate outputs, we follow the same strategy as above for a set of 200 randomly selected event-centered input instances (100 each of the type 'X `Causes` Y' **and** 'Y `IsAfter` X')[1]. We then present the model output and the true reference from ATOMIC to three human annotators. For example, a typical instance presented to human evaluators was of the form:

`Input Instance:` `PersonX drops out of high school`
`Response 1:` `PersonX gets a job`
`Response 2:` `PersonX turns PersonX's life around`
`Type:` `IsAfter`

One of the *responses* above is the LLM output, while the other is the true reference. The human evaluators are then asked to answer the following questions–

- Are both responses *functionally* similar?
- Which response do you prefer?
- Which response, if any, is completely irrelevant?

[1]Complete details about ATOMIC are available in Table 1 in (Hwang et al., 2021).

We find that in an overwhelming majority of the cases, the models generate output that is semantically equivalent to the reference (even though there is no direct lexical alignment), or output that the humans prefer over the true reference. Based on the responses from human evaluators[2], we observe that humans found 65.82% of response pairs *functionally* equivalent. That is, even though not lexically aligned, they meant the same thing. In 27.64% of the instances the humans preferred the model-generated event instance over the true reference. In only 6.55% of instances did the humans prefer the true reference over the model-generated output.[3] These results reinforce our underlying assertion that LLMs are capable of event-centered reasoning, and therefore could produce high-quality event sequence collections.

3.2 Evaluation of Recall

Despite the sparsity of causal relations in Wikidata, one can still reasonably estimate recall through pairwise comparisons of generated event classes to the existing causal relations in Wikidata. We build a reference set of causal relations in Wikidata by curating a list of all the pairs of event concepts that have any of the several causal relations[4] in Wikidata in any direction, including `has_cause (P828)`, `has_immediate_cause (P1478)`, `has_contributing_factor (P1479)`, and `has_effect(P1542)`.

3.3 Evaluation of Precision

To overcome sparsity in reference data, prior work has generally relied on human evaluations (Chiang and Lee, 2023) for estimating precision, which entails looking at pairs of events and asking annotators whether or not the events in question have a causal relationship. Such a process for potentially thousands of event pairs (like in our case) can be very cost prohibitive. Furthermore, recent research (Zhao et al., 2022; He et al., 2023; **?**) indicates that pre-trained language models themselves might outperform lay human annotators such as those found on Amazon Mechanical Turk. For instance, (He et al., 2023) demonstrated that labeling data under a few-shot chain-of-thought prompt ("explain-then-annotate" setting) surpasses crowdworker annotations on relevance assessments. Following these results, we use LLMs for evaluating precision and for selecting the most optimal model. We propose evaluating precision for model selection as a binary classification task. Given two events e_1, e_2, an *evaluator* model must evaluate whether e_1 reasonably leads to e_2 under a `has_cause` or `has_effect` relationship. To do this, we start with a large instruction-tuned model (in our case, Flan-T5-XXL (11B) (Chung et al., 2022)) and adopt instructional in-context few shot prompting to classify whether or not e_1 reasonably leads to e_2, and for the model to provide a justification for its results. While we hypothesize that such an approach may yield noisy results, a small manual assessment as well as our results comparing different models with different evaluator models indicate that this approach yields reliable results for comparing different models, and a reasonable estimate of the overall precision of the model.

3.4 Results

Table 1 summarizes our results from these experiments. While prior work as proven the concept of LLM-as-a-judge (**?**) when much larger LLMs like GPT-4 are used, here we use less resource-intensive LLMs not for evaluation of the absolute quality of the generated sequences, but for comparison of the relative performance of models

Since we use our models for dual purposes – for generating *and* evaluating event sequences, owing to this inherent conflict we find it prudent to independently assess these evaluator models. To ensure robustness of our results, we use multiple evaluator models and find no significant difference between evaluated precision across different models used as evaluators with the largest model performing marginally better as a precision evaluator.

4 Knowledge Distillation Through Event Sequence Analysis

Given a high-quality collection of event sequences generated by utilizing LLMs, we use pattern mining to discover high-support sequence patterns not directly derivable from the knowledge graph, and learn probabilistic models to discover complex event sequence rules.

4.1 Mining New Patterns

In order to derive new, unseen relationships (`has_cause` or `has_effect`) between the ex-

[2]We observe strong inter-rater agreement with a Fleiss kappa, $\kappa = 0.81$; conflicting response labels were aggregated through majority vote.

[3]Additional details on these experiments are available in the supplementary material (Appendix).

[4]www.wikidata.org/wiki/Wikidata:List_of_properties/causality

	Precision Evaluator Model (P)			R	F-1
	Flan-T5-Large	*Flan-T5-XL*	*Flan-T5-XXL*		
Flan-T5-Large (783M) (Chung et al., 2022)	0.73	0.72	0.72	0.47	0.57
Flan-T5-XL (3B)	0.75	0.75	0.78	0.49	0.60
Open-Research LLaMA (3B) (Touvron et al., 2023)	0.70	0.69	0.72	0.45	0.55
Flan-T5-XXL (11B)	**0.79**	**0.79**	**0.81**	**0.54**	**0.65**

Table 1: Evaluation of LLM-generated sequences. For the purpose of evaluating recall (R), we count event pairs (e_1, e_2) if such a pair exists in Wikidata. For evaluating precision (P), we treat correctness of all generated event pairs (e_1, e_2) as a standard classification task. Best-performing model scores are in bold.

Algorithm 1 Generalized Sequential Pattern (GSP) Mining Algorithm

Input: Database of sequences D, minimum support threshold min_sup
Output: Set of all sequential patterns SP
$SP \leftarrow \emptyset$
$C_1 \leftarrow$ set of all individual items in D that meet min_sup
$L_1 \leftarrow$ filter candidates in C_1 by min_sup
$k \leftarrow 2$
while $L_{k-1} \neq \emptyset$ **do**
  $C_k \leftarrow$ generate_candidates(L_{k-1})
  for each sequence $s \in D$ **do**
    for each candidate $c \in C_k$ **do**
      if c is contained in s **then**
        increment support count of c
      end if
    end for
  end for
  $L_k \leftarrow$ filter candidates in C_k by min_sup
  $SP \leftarrow SP \cup L_k$
  $k \leftarrow k + 1$
end while
return SP

tracted event classes, we use the generated event sequence collection followed by classical frequent itemset mining algorithms like GSP (Srikant and Agrawal, 1996) and SPADE (Zaki, 2004) to derive new high support patterns.

The classical pattern mining algorithms we use to mine for new event patterns are both Apriori-based approaches. Under both methods, given two sequences of the same event concept class

$$\alpha =< a_1, a_2, .., a_n > \text{ and } \beta =< b_1, b_2, .., b_n > \quad (2)$$

α is a subsequence of β denoted as $\alpha \subseteq \beta$ iff there exists a set of values $1 \leq j_1 < j_2 < ... < j_n \leq m$ such that $a_1 \subseteq b_{j1}, a_2 \subseteq b_{j2}, ..., a_n \subseteq b_{jn}$,; then β is a supersequence of α. Given multiple such sequences and a support threshold, the task here is to find a set of frequent event subsequences. GSP is depicted in Algorithm 1). SPADE uses a "vertical database format", which stores sequences as lists of itemsets associated with their IDs (referred to as "id-lists"), which allows faster support counting.

4.1.1 Examples of Mined Patterns

Following are example outputs from application of GSP on LLM outputs.

Mined Pattern Example 1

Pattern: Civil Disorder (Q686984) → Democratization (Q1064441) → Energy Crises (Q8413663) **Support:** 5

The pattern above occurs with a support value of 5 i.e. at least supported by 5 super-sequences. Empirically, we can find support for such a pattern in history.[5] In 1994, the country of South Africa democratized post civil disorder. This led to an increased energy demand over the following decade, eventually culminating in a full blown energy crises starting 2003.

Mined Pattern Example 2

Pattern: Famine (Q168247) → Refugee Crises (Q20898283) → Post Traumatic Stress Disorder(Q202387)
Support: 5

The pattern above again occurs with a support value of 5 i.e. at least supported by 5 super-sequences. Similar to the previous example, evidence supporting such an event tranisiton can be found in real life.[6] During the Great Irish Famine people were forced to relocate and flee Ireland causing a refugee crises. A great number of these individuals suffered from mental health crises (e.g. PTSD) due to the events directly associated with the famine.

4.2 Identifying Influencing Sets through Summary Markov Models

Learning about potential *influencing events* that lead to a given event type in a large set of event sequences is an important aspect of analyzing multivariate event sequences, i.e. events *without time*

[5] wikipedia.org/wiki/South_African_energy_crisis
[6] wikipedia.org/wiki/Great_Famine_(Ireland)

stamps, like the ones generated by our models. Classic k_{th} order Markov chains capture these dynamics by modeling the probability of observing a particular event type given the preceding k events in-sequence. The recent family of summary Markov models (SuMMs) (Bhattacharjya et al., 2022) generalize other well known Markov models for event sequences by leveraging a function that summarizes historical event occurrences, and identifying the subset of event types that affect the probability of occurrence of event types of interest; this forms the influencing set.

We use the LLM-generated sequences to learn two types of SuMMs: binary SuMMs (BSuMMs) and ordinal SuMMs (OSuMMs). In BSuMMs, it is only the presence or absence of an event in the relevant history that has an effect on the occurrence of other events, while in OSuMMs the order of the events is also taken into account. We refer the reader to Sections 3.3 and 3.4 in (Bhattacharjya et al., 2022) for complete formal definitions and methods for learning SuMMs over event sequence collections. Briefly, given a subset of event labels of interest $\mathbf{X} \in \mathcal{L}$ and a set of parameters $\Theta_X = \{\theta_{x|h}\}$, where $\theta_{x|h}$ is the probability of a given event label $x \in \mathbf{X}$ occurring at any position in the sequence given the historical event occurrences h, *influencing* and *non-influencing* event sets can be formally defined as label sets $\mathbf{U} = \mathcal{L} \setminus \mathbf{U}$ are influencing sets for event labels $\mathbf{X}$ such that they minimally determine the probability of observing any particular label of interest x_i for a given position i in the sequence.

4.2.1 Evaluation: SuMMs vs LSTMs

Following the evaluation strategy implemented in (Bhattacharjya et al., 2022), we focus on individual labels of interest and conduct an evaluation around probabilistic prediction. Consequently, we select negative log loss as the evaluation metric. Table 2 summarizes our results on the test set for both BSuMMs and OSuMMs, along with a simple LSTM baseline. We observe that models trained on event sequences generated from larger models (e.g. Flan-T5-XXL) fare better than the ones generated from their smaller counterparts (e.g. Flan-T5-Large). We treat this observation as a proxy for generated event sequence quality. That is, better quality sequences lead to better predictive models.

4.2.2 Qualitative Assessment

Figure 1 shows examples of learned influencing sets using BSuMMs for `refugee crisis` and `mass migration` events. Two events `mass migration` and `famine` are identified as influencing events for `refugee crisis`. The model indicates that the occurrence of both `mass migration` and `famine` together has a 0.91 probability of resulting in `refugee crisis` as a part of a sequence of events. On the other hand the occurrence of `mass migration` in the absence of `famine` has a 0.31 probability of resulting in `refugee crisis`. BSuMMs and OSuMMs deploy a greedy score-based forward and backward search strategy to efficiently discover the minimal influencing sets. Overall, the discovery of influencing sets from LLM-generated data provides a mechanism of distilling complex symbolic knowledge from the output of LLMs.

We provide two more examples below of influencing sets derived from the application of SuMMs. For each example, we show evidence supporting the accuracy of the extracted knowledge.

Influencing Set Example 1 The following example identifies "Hate Crimes" as a predecessor to the occurrence of "Civil Disorder" related events.

X: `Civil Disorder (Q686984)`
Parent: `Hate Crime (Q459409)`

- $P($`Civil Disorder`$|$`NO Hate Crime`$) = 0.12$
- $P($`Civil Disorder`$|$`Hate Crime`$) = 0.55$

In the example above, we see that the likelihood of a civil disorder is greatly influenced by the occurrence of a hate crime.

Influencing Set Example 2 The following example identifies "Disease Outbreaks" and "Lockdowns" as precursors to the institution of "Travel Restrictions".

X: `Travel Restriction (Q87745167)`
Parents: `Disease Outbreak (Q3241045), Lockdown (Q6665312)`

- $P($`TR`$|$`NO Outbreak, NO Lockdown`$) = 0.0.0001$
- $P($`TR`$|$`Lockdown, NO Outbreak`$) = 0.0.29$
- $P($`TR`$|$`NO Lockdown, Outbreak`$) = 0.26$

Quite intuitively, we see here the likelihood of Travel Restrictions directly correlate with the institution of Disease Outbreaks and Lockdowns (since the latter have been lately associated with the former).

(↓) Data Generator Model	BSuMMs	OSuMMs	LSTM
Flan-T5-Large (783M) (Chung et al., 2022)	-63.49	-84.29	-109.28
Flan-T5-XL (3B)	-63.20	-92.65	-121.23
Open-Research LLaMA (3B) (Touvron et al., 2023)	-110.57	-101.29	-190.68
Flan-T5-XXL (11B)	**-57.99**	**-78.64**	**-108.89**

Table 2: Negative log likelihood (*lower magnitude is better*) averaged over interest labels from LLM-generated (Flan-T5-XXL) event sequences. Lookback window (k) for LSTM was fixed to 5. Best-performing data generator model scores are in bold.

5 Related Work

News Event Analysis The primary applications of our work are around news event analysis and forecasting. Liang (Zhao, 2021) presents a taxonomy of different flavors of event prediction in the literature. Our target event prediction applications fall under the "Semantic Prediction" category, with time and location details not being of interest, and the primary goal being the prediction of "event profiles" such as event types. Seminal work in this area is the work of Radinsky et al. (Radinsky et al., 2012) where causal relations between past events are extracted from text and then a knowledge graph is utilized to generalize the extracted relations in order to make predictions. More recent work has explored the use of graph sequence mining over a graph structure representation of events extracted from text (Cekinel and Karagoz, 2022), with graph mining used as a mechanism of extracting useful relations from large and noisy outputs of extraction.

Event Sequence Extraction from Text There is a wealth of literature on different methods of extracting sequences from textual corpora. Norambuena et al. (Norambuena et al., 2023) present a comprehensive survey of automated methods of narrative extraction. Narratives contain several elements including events, participants (actors/protagonists), time, and space. The task of event detection is highly challenging and the topic of extensive research (Chen et al., 2021; Xiang and Wang, 2019; Li et al., 2020), which has its root in the Topic Detection and Tracking (TDT) task (Santana et al., 2023). This line of work started out as a DARPA-sponsored initiative with the same name (Allan et al., 1998). Another closely related task is news timeline summarization (Gholipour Ghalandari and Ifrim, 2020). While pattern mining algorithms have been applied to the output of such extractions, e.g. for creation of "domain templates" (Filatova et al., 2006), we are not aware of any attempts to use the extractions to construct event models for analysis or prediction.

Sequential Pattern Mining and Event Sequence Models Sequential pattern mining and related approaches have been the subject of extensive research (Mannila et al., 1997; Mabroukeh and Ezeife, 2010; Mooney and Roddick, 2013; Fournier-Viger et al., 2017). These algorithms take in a set of sequences (or "sequential records") with a set of unique events (or "items") and often a "minimum support" threshold, and return as output a ranked set of all frequent sequences in a given sequence collection (or database) meeting the minimum support threshold. Our focus in this paper is on multivariate event sequences, i.e., sequences of various event types without timestamps. Markov models for sequences (Raftery, 1985; Begleiter et al., 2004) and long short-term memory (LSTM) (Hochreiter and Schmidhuber, 1997) models are examples of prediction models. We leverage a more recent family of models – SUMMs (Bhattacharjya et al., 2022) – for some of the experiments in this paper that involve analyzing generated sequences.

6 Conclusions

In this paper, we presented methods of distilling event sequence knowledge from large language models. This work demonstrates the utility of large language models for extracting event-related information and helps researchers better navigate the complex interaction between event-related entities. While we have demonstrated some use-cases for the resulting dataset of event sequences – mining logical rules/patterns and extracting influencing event types – the resulting datasets in this work and those from other domains could themselves be a useful resource for researchers, as one may leverage them suitably to inform or justify decision-making. We make our code and datasets publicly available for future research.

Limitations

We have proposed new methods of generating event sequences and establishing associations between the events. This work has several important limitations, and subsequently, opportunities for future work.

Limited Data First, we considered a dataset (Wikidata) of only *newsworthy* event concepts – and consequently generated sequences of the same type. Such generic concepts often appear in the pretraining datasets of large language models. We did not consider more complex, domain-specific datasets of non-timestamped events (e.g. Healthcare, Finance). Applying LLMs to generate domain specific events may require considerable amounts of data to fine-tune these models, and collecting such supervision to train these models at scale may be prohibitively expensive. We leave such analysis for future work.

Noisy Evaluation Second, a key element of our evaluation strategy (precision) involves the use of the same language models that were used to generate the sequences being evaluated in the first place. Through qualitative examples and past research on using LLMs as annotators, we observe that such a strategy yields a noisy but useful signal on evaluating model performance. However, an ideal and accurate evaluation of model outputs should involve use of human annotators. We also do not attempt to evaluate the correctness of post-hoc explanations generated by models for their corresponding output during precision evaluation.

Use of Even Larger Models Third, a methodological limitation of our work is that we did not experiment with GPT-$*$ models (OpenAI, 2023). We could use OpenAI's API for the task we've proposed, which might yield improved results. However, our primary goal was to frame this task as a language modeling problem; the specific choice of an underlying LLM is a secondary consideration. Further, opting for open-source models ensures transparency allowing us to release the necessary code and other details for reproducibility.

English-Only Experiments Finally, we only conducted experiments with data in English and therefore, we do not know what issues may occur replicating our work with data in other languages.

Ethical Considerations

We believe our work has potential applications that can significantly contribute to human well being through the development of probabilistic event models on real-world data. Realizing these potential applications, however, raises concerns about how such models may be used to influence policy decisions and broader viewpoints of those interpreting the data being generated. In light of these ethical considerations, we emphasize that any resulting event models created through the use of model-generated data should be *thoroughly* evaluated by humans, including a comprehensive evaluation of the generated data itself to adjust for risk of bias and its resulting effects.

References

Sweta Agrawal, Chunting Zhou, Mike Lewis, Luke Zettlemoyer, and Marjan Ghazvininejad. 2023. In-context examples selection for machine translation. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 8857–8873, Toronto, Canada. Association for Computational Linguistics.

James Allan. 2012. *Topic Detection and Tracking: Event-Based Information Organization*. Springer Publishing Company, Incorporated.

James Allan, Jaime G. Carbonell, George Doddington, Jonathan Yamron, and Yiming Yang. 1998. Topic Detection and Tracking Pilot Study Final Report. In *Proceedings of the DARPA Broadcast News Transcription and Understanding Workshop, February, 1998.*

Shengnan An, Zeqi Lin, Qiang Fu, Bei Chen, Nanning Zheng, Jian-Guang Lou, and Dongmei Zhang. 2023. How do in-context examples affect compositional generalization? In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11027–11052, Toronto, Canada. Association for Computational Linguistics.

Ron Begleiter, Ran El-Yaniv, and Golan Yona. 2004. On prediction using variable order markov models. *J. Artif. Intell. Res.*, 22:385–421.

Debarun Bhattacharjya, Saurabh Sihag, Oktie Hassanzadeh, and Liza Bialik. 2022. Summary markov models for event sequences. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, pages 4836–4842. International Joint Conferences on Artificial Intelligence Organization. Main Track.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda

Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Recep Firat Cekinel and Pinar Karagoz. 2022. Event prediction from news text using subgraph embedding and graph sequence mining. *World Wide Web*, 25(6):2403–2428.

Muhao Chen, Hongming Zhang, Qiang Ning, Manling Li, Heng Ji, Kathleen McKeown, and Dan Roth. 2021. Event-centric natural language processing. In *ACL*.

Cheng-Han Chiang and Hung-yi Lee. 2023. Can large language models be an alternative to human evaluations? In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15607–15631, Toronto, Canada. Association for Computational Linguistics.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. 2022. Scaling instruction-finetuned language models.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Xinya Du, Zixuan Zhang, Sha Li, Pengfei Yu, Hongwei Wang, Tuan Lai, Xudong Lin, Ziqi Wang, Iris Liu, Ben Zhou, Haoyang Wen, Manling Li, Darryl Hannan, Jie Lei, Hyounghun Kim, Rotem Dror, Haoyu Wang, Michael Regan, Qi Zeng, Qing Lyu, Charles Yu, Carl Edwards, Xiaomeng Jin, Yizhu Jiao, Ghazaleh Kazeminejad, Zhenhailong Wang, Chris Callison-Burch, Mohit Bansal, Carl Vondrick, Jiawei Han, Dan Roth, Shih-Fu Chang, Martha Palmer, and Heng Ji. 2022. RESIN-11: Schema-guided event prediction for 11 newsworthy scenarios. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: System Demonstrations*, pages 54–63, Hybrid: Seattle, Washington + Online. Association for Computational Linguistics.

Elena Filatova, Vasileios Hatzivassiloglou, and Kathleen McKeown. 2006. Automatic creation of domain templates. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, pages 207–214, Sydney, Australia. Association for Computational Linguistics.

Philippe Fournier-Viger, Jerry Chun-Wei Lin, Rage Uday Kiran, and Yun Sing Koh. 2017. A survey of sequential pattern mining. *Data Science and Pattern Recognition*, 1(1):54–77.

Demian Gholipour Ghalandari and Georgiana Ifrim. 2020. Examining the state-of-the-art in news timeline summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1322–1334, Online. Association for Computational Linguistics.

Oktie Hassanzadeh, Parul Awasthy, Ken Barker, Onkar Bhardwaj, Debarun Bhattacharjya, Mark Feblowitz, Lee Martie, Jian Ni, Kavitha Srinivas, and Lucy Yip. 2022. Knowledge-based news event analysis and forecasting toolkit. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI 2022, Vienna, Austria, 23-29 July 2022*, pages 5904–5907. ijcai.org.

Xingwei He, Zhenghao Lin, Yeyun Gong, A-Long Jin, Hang Zhang, Chen Lin, Jian Jiao, Siu Ming Yiu, Nan Duan, and Weizhu Chen. 2023. Annollm: Making large language models to be better crowdsourced annotators.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.*, 9(8):1735–1780.

Jena D. Hwang, Chandra Bhagavatula, Ronan Le Bras, Jeff Da, Keisuke Sakaguchi, Antoine Bosselut, and Yejin Choi. 2021. Comet-atomic 2020: On symbolic and neural commonsense knowledge graphs. In *AAAI*.

Shima Imani, Liang Du, and Harsh Shrivastava. 2023. MathPrompter: Mathematical reasoning using large language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 5: Industry Track)*, pages 37–42, Toronto, Canada. Association for Computational Linguistics.

Ehsan Kamalloo, Nouha Dziri, Charles Clarke, and Davood Rafiei. 2023. Evaluating open-domain question answering in the era of large language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5591–5606, Toronto, Canada. Association for Computational Linguistics.

Manling Li, Qi Zeng, Ying Lin, Kyunghyun Cho, Heng Ji, Jonathan May, Nathanael Chambers, and Clare

Voss. 2020. Connecting the dots: Event graph schema induction with path language modeling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 684–695, Online. Association for Computational Linguistics.

Sha Li, Ruining Zhao, Manling Li, Heng Ji, Chris Callison-Burch, and Jiawei Han. 2023. Open-domain hierarchical event schema induction by incremental prompting and verification. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5677–5697, Toronto, Canada. Association for Computational Linguistics.

Nizar R. Mabroukeh and C. I. Ezeife. 2010. A taxonomy of sequential pattern mining algorithms. *ACM Comput. Surv.*, 43(1).

H. Mannila, H. Toivonen, and A. I. Verkamo. 1997. Discovery of frequent episodes in event sequences. *Data Mining and Knowledge Discovery*, 1:259–289.

Carl H. Mooney and John F. Roddick. 2013. Sequential pattern mining – approaches and algorithms. *ACM Comput. Surv.*, 45(2).

Brian Keith Norambuena, Tanushree Mitra, and Chris North. 2023. A survey on event-based news narrative extraction. *ACM Comput. Surv.*, 55(14s).

OpenAI. 2023. Gpt-4 technical report. *ArXiv*, abs/2303.08774.

Kira Radinsky, Sagie Davidovich, and Shaul Markovitch. 2012. Learning causality for news events prediction. In *Proceedings of the 21st World Wide Web Conference 2012, WWW 2012, Lyon, France, April 16-20, 2012*, pages 909–918. ACM.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(1).

A. Raftery. 1985. A model for high-order Markov chains. *Journal of the Royal Statistical Society, Series B*, 47(3):528–539.

Brenda Salenave Santana, Ricardo Campos, Evelin Amorim, Alípio Jorge, Purificação Silvano, and Sérgio Nunes. 2023. A survey on narrative extraction from textual data. *Artif. Intell. Rev.*, 56(8):8393–8435.

Ramakrishnan Srikant and Rakesh Agrawal. 1996. Mining sequential patterns: Generalizations and performance improvements. In *Advances in Database Technology — EDBT '96*, pages 1–17, Berlin, Heidelberg. Springer Berlin Heidelberg.

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*.

Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: A free collaborative knowledgebase. *Commun. ACM*, 57(10):78–85.

Somin Wadhwa, Silvio Amir, and Byron Wallace. 2023. Revisiting relation extraction in the era of large language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15566–15589, Toronto, Canada. Association for Computational Linguistics.

Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V Le. 2022. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*.

Peter West, Chandra Bhagavatula, Jack Hessel, Jena Hwang, Liwei Jiang, Ronan Le Bras, Ximing Lu, Sean Welleck, and Yejin Choi. 2022. Symbolic knowledge distillation: from general language models to commonsense models. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4602–4625, Seattle, United States. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Wei Xiang and Bang Wang. 2019. A survey of event extraction from text. *IEEE Access*, 7:173111–173137.

Wenhao Yu, Chenguang Zhu, Zaitang Li, Zhiting Hu, Qingyun Wang, Heng Ji, and Meng Jiang. 2022. A survey of knowledge-enhanced text generation. *ACM Comput. Surv.*, 54(11s).

Mohammed J. Zaki. 2004. Spade: An efficient algorithm for mining frequent sequences. *Machine Learning*, 42:31–60.

Yiming Zhang, Shi Feng, and Chenhao Tan. 2022. Active example selection for in-context learning. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9134–9148, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Liang Zhao. 2021. Event prediction in the big data era: A systematic survey. *ACM Comput. Surv.*, 54(5).

Mengjie Zhao, Fei Mi, Yasheng Wang, Minglei Li, Xin Jiang, Qun Liu, and Hinrich Schuetze. 2022. LM-Turk: Few-shot learners as crowdsourcing workers in a language-model-as-a-service framework. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 675–692, Seattle, United States. Association for Computational Linguistics.

Appendix

This section constitutes technical appendix (i.e. supplementary material) for the submission titled "Distilling Event Sequences from Large Language Models".

A Experimental Settings

We performed all our model inference related experiments on two NVIDIA V100 GPUs. We used the Huggingface library v4.26.1 (Wolf et al., 2020) and publicly available model checkpoints.[7] Classical pattern mining algorithms (GSP, SPADE) were implemented in Python through the use of `spmf`[8] library v1.4. For event sequence generation, we use in-context learning under zero shot settings for all LLMs, with `top-k` sampling ($k = 50$, in conjunction with `top-p`, where $p = 0.95$). To identify influencing events from Summary Markov Models (SuMMs), we use implementations from (Bhattacharjya et al., 2022) and split the dataset (generated from LLMs) into train/dev/test sets ($70\%/15\%/15\%$) to generate results reported in Table 2. BSuMMs and OSuMMs were learned with hyperparameters of $\alpha = 0.1$, $\gamma = 0.5$ and a look-back (κ) window of 4.

B Supplemental Material Statement

Our supplementary material (included as a zip file in our submission) includes code and the prompts used for event sequence generation and benchmarking, as well as the base KG, and sample output files. `README.md` has all the details.

C Event Sequence Prompts

We now describe the specific prompt-types used (including pseudo-code for their use) to generate event sequences given individual Wikidata event concepts. We start a Wikidata event concept label and feed it into the following prompt with 3 ICL exemplars. The prompt also includes a label space of other Wikidata concepts from which we i nstruct the model to choose a concept. In case of an out-of-domain generated output ($\approx 15\%$ of total outputs), we discard those outputs.

```
def build_prompt1(vocab, target_label):
    prompt = "Use the following vocabulary to respond to the questions: " + \
            f"{' '.join(label_space)}\n" + \
            f"Question: what usually happens after earthquake?\n" +\
            f"Answer: tsunami\n" + \

            f"Question: what usually happens after economic crises?\n" +\
            f"Answer: unemployment\n" + \

            f"Question: what usually happens after bomb attack?\n" +\
            f"Answer: injury\n" + \

            f"Question: what usually happens after {target_label}?\n" +\
            f"Answer:"

    return prompt
```

Listing 1: Initial Trigger Prompt

Following the output from the above prompt trigger, we feed the in-domain outputs to the following ICL iterative prompt with *conjunctive* event exemplars (i.e. `X and Y`). We then successively use this prompt by feeding model generated outputs as new event concepts. Restricting the generated vocabulary to existing Wikidata concepts allows for a reasonable recall evaluation even from a sparse reference set. We illustrate this approach in Figure 2 in the main paper.

```
def build_prompt2(vocab, target_labels):

    prompt = "Use the following vocabulary to respond to the questions: " + \
            f"{' '.join(label_space)}\n" + \
            f"Question: what usually happens after earthquake?\n" +\
            f"Answer: tsunami\n" + \

            f"Question: what usually happens after earthquake and tsunami?\n" +\
            f"Answer: nuclear disaster\n" + \

            f"Question: what usually happens after economic crises and wage decline and unemployment?\n" +\
            f"Answer: legislation\n" + \

            f"Question: what usually happens after military conflict?\n" +\
            f"Answer: war\n" + \

            f"Question: what usually happens after military conflict and war?\n" +\
```

[7] huggingface.co/docs/transformers/model_doc/flan-t5

[8] pypi.org/project/spmf/

```
            f"Answer: peace treaty\n" + \

            f"Question: what usually
    happens after {' and '.join(
    target_labels)}?\n" +\
            f"Answer:"

    return prompt
```

Listing 2: Iterative ICL Prompt with Conjunctive Examples

D Evaluator Models (Example Outputs and Prompts)

For a model generated event sequence $\alpha = a_1, a_2, .., a_n$, we consider all possible pairs of events (a_i, a_{i+1}) as (trigger, consequence) pairs where a_0 is the initial trigger sequence and $i \in 1, .., n$ are all events generated by the model that follow a_0. Then we use the following prompt to evaluate correctness of all such possible pairs and use this as proxy for a true precision evaluation of the generated sequences.

```
def build_precision_eval(trigger,
    consequence):
    prompt = "Respond to the questions
    below with a (YES/NO) with a
    historical example:" + \
            f"Question: Can economic
    crises cause a landslide?\n" +\
            f"Answer: NO. There is no
    historical example of an economic
    crisis causing a landslide, which is
     natural disaster.\n" + \
            f"Question: Can earthquake
    cause a tsunami?\n" +\
            f"Answer: YES. In 2011,
    Japan experienced an earthquake in
    tohoku that caused a tsunami. \n" +
    \
            f"Question: Can mass
    shooting cause a condensation cloud
    ?\n" +\
            f"Answer: NO. A condensation
     cloud is a weather phenomenon, not
    a mass shooting.\n" + \
            f"Question: Can accident
    cause a stock market crash?\n" +\
            f"Answer: NO. The stock
    market crash of 1929 was caused by a
     series of events, not an accident.\
    n" + \
            f"Question: Can disease
    outbreak cause a inventory shrinkage
    ?\n" +\
            f"Answer: YES. The bubonic
    plague outbreak in Europe in 1348
    caused a massive inventory shrinkage
    .\n" + \
            f"Question: Can fraud cause
    a travel ban?\n" +\
            f"Answer: YES. Travel bans
    are a form of punishment for
    immigration fraud.\n" + \
            f"Question: Can {trigger}
    cause a {consequence}?\n" + "Answer:
     "
    return prompt
```

Listing 3: Prompt used to evaluate correctness of a given event pair

As mentioned in the limitations section, a key shortcoming of our approach here is that we do not evaluate the correctness of the post-hoc explanations generated by the model for it's classification label. We leave that analysis for future work.

E ATOMIC Human Evaluations

Amazon Mechanical Turk (AMT) is a platform for non-expert works to perform microtasks, in our case – human annotations. Three authors with graduate degrees in computer science carried out these human evaluations. Figure 3 shows the interface provided to these human annotators. The human evaluators were asked to answer the following questions–

- Are both responses *functionally* similar?
- Which response do you prefer?
- Which response, if any, is completely irrelevant?

Functionally Similar Responses For responses to be functionally similar, they *must* convey the same meaning even if there is major lexical mismatch between the actual tokens.

Human Preferences The preferences elicited here are based on a humans degree of reasonableness given an event trigger. We observed a high Fleiss $\kappa = 0.81$ indiciating a high degree of agreement between the annotators.

Irrelevant Responses Here, again the annotators were asked to exercise judgment in what they may find a *completely unreasonable* response to a given event trigger.

We release the results from these human annotations for a comprehensive analysis and any additional findings that readers may infer.

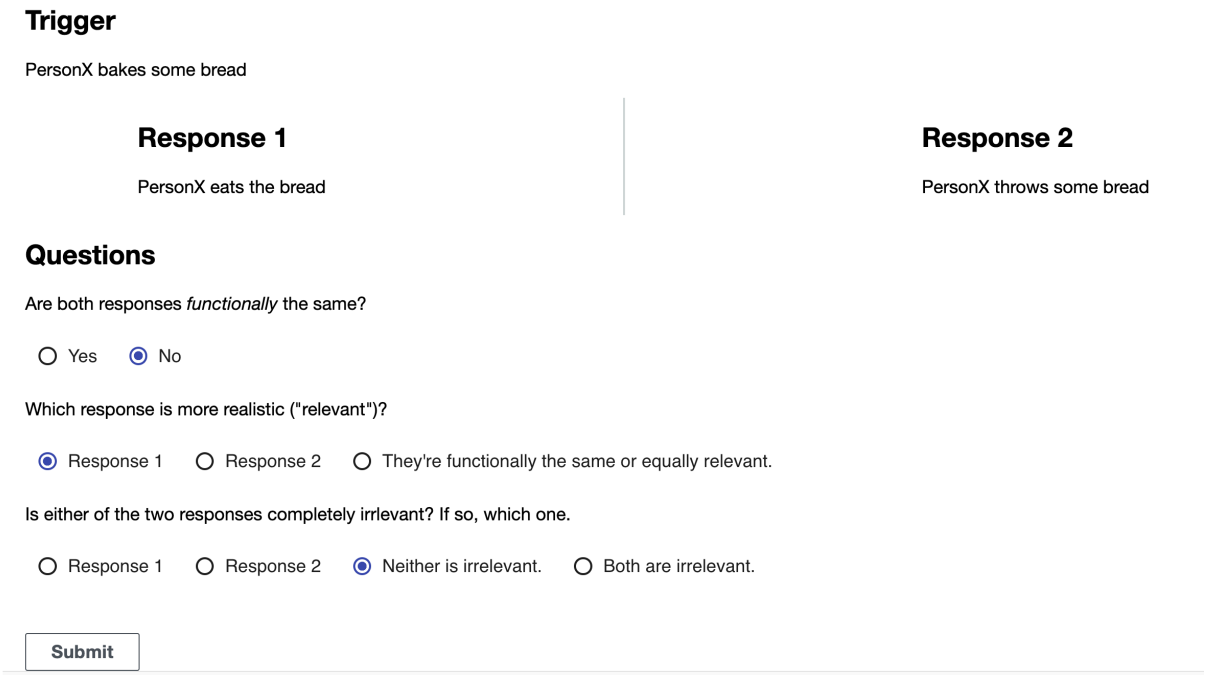

Figure 3: Annotation interface for classifying model generated response vs reference (anonymized) for the ATOMIC event descriptions.