Distilling Event Sequence Knowledge From Large Language Models

Somin Wadhwa[1,2(✉)], Oktie Hassanzadeh[1], Debarun Bhattacharjya[1], Ken Barker[1], and Jian Ni[1]

[1] IBM Research, Yorktown Heights, NY, USA
{hassanzadeh,debarunb,kjbarker}@us.ibm.com
[2] Northeastern University, Boston, USA
wadhwa.s@northeastern.edu

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.

Keywords: Knowledge Graphs · Large Language Models · Knowledge Distillation

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

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-77844-5_13.

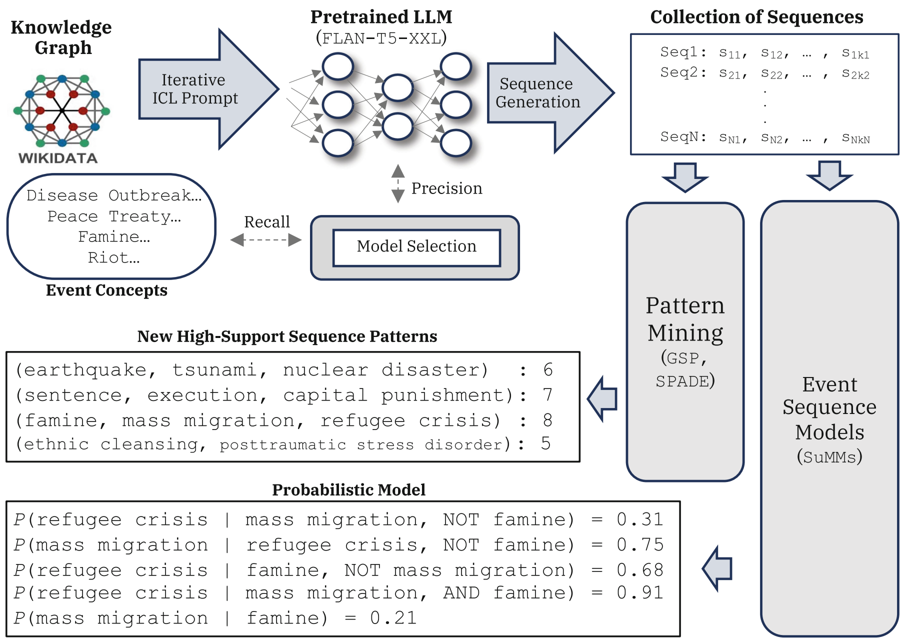
be used to detect and prevent potential cyber attacks by analyzing the sequence of events leading up to a breach. A common characteristic of the data in these domains is that sequences are clearly associated with an entity (e.g., a company, a person, or a device). There are however other domains where such a clean association between events and entities may not be possible. One such application is news event analysis [8,13,18,30]. 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 [29,33], topic detection and tracking [2], and timeline summarization [16]. 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. Prior work has shown little success in automatically turning narratives into structured sequences. In this paper, we explore a novel mechanism for creating structured event sequences in such domains, using generative language models.

Large Language Models [7,31,38] have recently become the dominant paradigm in a range of natural language processing (NLP) tasks [11] and often beat traditional approaches on a number of challenging tasks, including complex arithmetic reasoning [22] and open-domain question answering [23]. In this paper, our goal is to examine the capability of LLMs to generate structured event sequences useful for event analysis. Our aim is to use LLMs directly for generation of event sequences, as opposed to improving prior extraction methods that would rely on high-quality and comprehensive sources describing detailed event sequences for a wide variety of events. 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 [36], a comprehensive general-domain knowledge graph, to guide the sequence generation. This can be viewed as a novel mechanism for knowledge-guided text generation [42] and symbolic knowledge distillation [39]. 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.


Knowledge Graph
WIKIDATA
Iterative ICL Prompt
Pretrained LLM (FLAN-T5-XXL)
Sequence Generation
Collection of Sequences
Seq1: s_{11}, s_{12}, … , s_{1k1}
Seq2: s_{21}, s_{22}, … , s_{2k2}
SeqN: s_{N1}, s_{N2}, … , s_{NkN}
Disease Outbreak…
Peace Treaty…
Famine…
Riot…
Event Concepts
Recall
Precision
Model Selection
Pattern Mining (GSP, SPADE)
Event Sequence Models (SuMMs)
New High-Support Sequence Patterns
Probabilistic Model

```
(earthquake, tsunami, nuclear disaster)   : 6
(sentence, execution, capital punishment): 7
(famine, mass migration, refugee crisis) : 8
(ethnic cleansing, posttraumatic stress disorder): 5
```

```
P(refugee crisis | mass migration, NOT famine)  = 0.31
P(mass migration | refugee crisis, NOT famine)  = 0.75
P(refugee crisis | famine, NOT mass migration)  = 0.68
P(refugee crisis | mass migration, AND famine)  = 0.91
P(mass migration | famine) = 0.21
```

Fig. 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.

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 available at https://doi.org/10.5281/zenodo.13308920.

2 Related Work

News Event Analysis. The primary applications of our work are around news event analysis and forecasting. Liang [45] 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. [30] 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 [8], with graph mining used as a mechanism of extracting useful relations from large and noisy outputs of extraction. The application of building event sequence models over such outputs has not been explored. Our paper explores an alternative approach of event sequence generation that is capable of generating longer and potentially higher-quality sequences.

Event Sequence Extraction from Text. There is a wealth of literature on different methods of extracting sequences from textual corpora. Norambuena et al. [29] 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 itself is a highly challenging task and the topic of extensive research [9,24,41], which has its root in the Topic Detection and Tracking (TDT) task [33]. This line of work started out as a DARPA-sponsored initiative with the same name [3]. Another closely related task is news timeline summarization [16]. While pattern mining algorithms have been applied to the output of such extractions, e.g. for creation of "domain templates" [14], we are not aware of any attempts to use the extractions to construct event models for analysis or prediction. This is mainly due to the fact that the output of such methods often result in very short and noisy sequences, not suitable for the majority of event sequence models.

Sequential Pattern Mining and Event Sequence Models. Sequential pattern mining and related approaches have been the subject of extensive research [15,26–28]. 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. There is also a long line of work on statistical and probabilistic models for analysis of different kinds of event sequences. Our focus in this paper is on multivariate event sequences, i.e., sequences of various event types without timestamps. Markov models for sequences [5,32] and long short-term memory (LSTM) [20] models are examples of prediction models. We leverage a more recent family of models – Summary Markov models [6] – for some of the experiments in this paper that involve analyzing generated event sequences.

3 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:

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

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 (Fig. 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 [25]. 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 Fig. 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 (Fig. 2). Full length prompts used in all our experiments are provided in the Appendix.

To generate new event sequences in a zero-shot 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

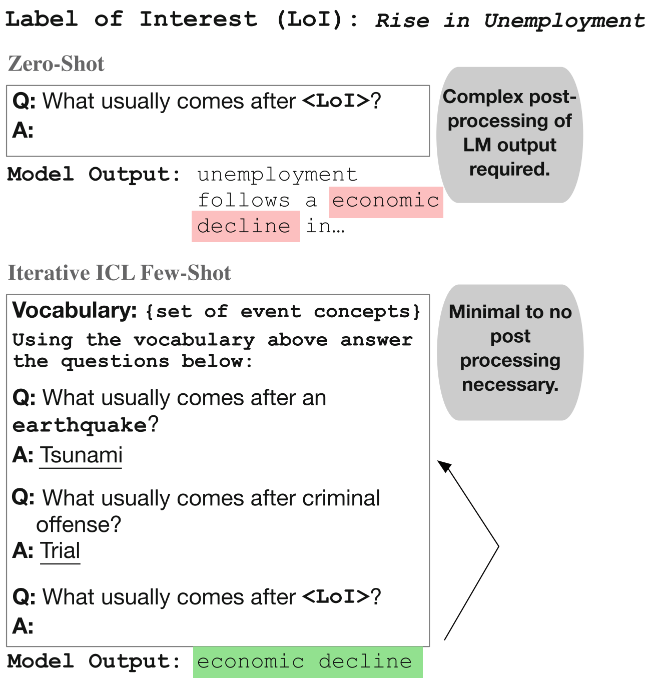

Label of Interest (LoI): Rise in Unemployment
Zero-Shot
Q: What usually comes after <LoI>?
A:
Model Output: unemployment follows a economic decline in…
Complex post-processing of LM output required.
Iterative ICL Few-Shot
Vocabulary: {set of event concepts}
Using the vocabulary above answer the questions below:
Q: What usually comes after an earthquake?
A: Tsunami
Q: What usually comes after criminal offense?
A: Trial
Q: What usually comes after <LoI>?
A:
Model Output: economic decline
Minimal to no post processing necessary.

Fig. 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.

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. This is in addition to the zero-shot experiments described in the main paper. 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 [44]. 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 [12] 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 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 [1,4] 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.

4 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 [37]. 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 Sect. 5.

4.1 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 [21]. 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?

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.

4.2 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)`. Our results show that this is an effective mechanism of comparing recall across methods, as larger models that are expected to have better accuracy yield higher recall scores.

[1] Complete details about ATOMIC are available in Table 1 in [21].
[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.

4.3 Evaluation of Precision

To overcome sparsity in reference data, prior work has generally relied on human evaluations [10] 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 [19,46,47] indicates that pre-trained language models themselves might outperform lay human annotators such as those found on Amazon Mechanical Turk. For instance, [19] 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) [11]) 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. Note that the evaluator performs only a simple binary classification task that has a very high accuracy even in smaller models.

4.4 Settings

We performed all our model inference related experiments on two NVIDIA V100 GPUs. We used the Huggingface library v4.26.1 [40] and publicly available model checkpoints.[5]. Classical pattern mining algorithms (GSP, SPADE) were implemented in Python through the use of `spmf`[6] 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 [6] 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.

4.5 Results

Table 1 summarizes our results from these experiments. While prior work as proven the concept of LLM-as-a-judge [47] when much larger LLMs like GPT-4 are used, here we use less resource-intensive LLMs not for evaluation of the

[5] https://huggingface.co/docs/transformers/model_doc/flan-t5
[6] pypi.org/project/spmf/.

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.

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.

	Precision Evaluator Model (P)			R	F-1
	Flan-T5-Large	*Flan-T5-XL*	*Flan-T5-XXL*		
Flan-T5-Large (783M) [11]	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) [35]	0.70	0.69	0.72	0.45	0.55
Flan-T5-XXL (11B)	**0.79**	**0.79**	**0.81**	**0.54**	**0.65**

4.6 Discussion

We observed in our analysis that some event triggers did not produce any sequences. Out of the 202 unique input event triggers in our data, 8 event triggers yielded sequences of length 1, i.e., no sequences. This means that the underlying model could not select a relevant event concept from the provided vocabulary that might follow the given trigger.

5 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.

5.1 Mining New Patterns

In order to derive new, unseen relationships (`has_cause` or `has_effect`) between the extracted event classes, we use the generated event sequence collection followed by classical frequent itemset mining algorithms like GSP [34] and SPADE

[43] to derive new high support patterns. This aspect of the overall workflow is illustrated in the middle section of Fig. 1, where high-support patterns are mined from event sequences.

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

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

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. The algorithm organizes the search space into equivalence classes, and avoids multiple scans of the sequences.

Algorithm 1. Generalized Sequential Pattern (GSP) Mining Algorithm

```
1: **Input:** Database of sequences $D$, minimum support threshold min_sup
2: **Output:** Set of all sequential patterns $SP$
3: $SP \leftarrow \emptyset$
4: $C_1 \leftarrow$ set of all individual items in $D$ that meet min_sup
5: $L_1 \leftarrow$ filter candidates in $C_1$ by min_sup
6: $k \leftarrow 2$
7: while $L_{k-1} \neq \emptyset$ do
8:     $C_k \leftarrow$ generate_candidates($L_{k-1}$)
9:     for each sequence $s \in D$ do
10:        for each candidate $c \in C_k$ do
11:            if $c$ is contained in $s$ then
12:                increment support count of $c$
13:            end if
14:        end for
15:    end for
16:    $L_k \leftarrow$ filter candidates in $C_k$ by min_sup
17:    $SP \leftarrow SP \cup L_k$
18:    $k \leftarrow k + 1$
19: end while
20: return $SP$
```

Examples of Mined Patterns. Following are example outputs from application of GSP on LLM outputs.

Mined Pattern Example 1.

`Pattern: Civil Disorder` *(Q686984)* $\rightarrow$ `Democratization` *(Q1064441)* $\rightarrow$ `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.[7]. 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)* $\rightarrow$ `Refugee Crises` *(Q20898283)* $\rightarrow$ `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.[8]. 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.

5.2 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 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) [6] 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 Sects. 3.3 and 3.4 in [6] 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

[7] https://en.wikipedia.org/wiki/South_African_energy_crisis
[8] https://en.wikipedia.org/wiki/Great_Famine_(Ireland)

occurrences h, *influencing* and *non-influencing* event sets can be formally defined as label sets $\mathbf{U} = \mathcal{L}\backslash\ \mathbf{U}$ 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.

Evaluation: SuMMs Vs LSTMs. Following the evaluation strategy implemented in [6], 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.

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.

(↓) Data Generator Model	BSuMMs	OSuMMs	LSTM
Flan-T5-Large (783M) [11]	-63.49	-84.29	-109.28
Flan-T5-XL (3B)	-63.20	-92.65	-121.23
Open-Research LLaMA (3B) [35]	-110.57	-101.29	-190.68
Flan-T5-XXL (11B)	**-57.99**	**-78.64**	**-108.89**

Qualitative Assessment. Fig. 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. Overall, these results show the high quality and usefulness of the distilled event sequence 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)
```

- $P(\texttt{Civil Disorder}|\texttt{NO Hate Crime}) = 0.12$
- $P(\texttt{Civil Disorder}|\texttt{Hate Crime}) = 0.55$

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)
```

- $P(\texttt{TR}|\texttt{NO Outbreak, NO Lockdown}) = 0.0.0001$
- $P(\texttt{TR}|\texttt{Lockdown, NO Outbreak}) = 0.0.29$
- $P(\texttt{TR}|\texttt{NO Lockdown, Outbreak}) = 0.26$

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).

Error Analysis. We observed that in some cases, the identified influencing sets for some event concepts were not correct or, at times, illogical. Qualitatively, we observe that this occurs with rare event types that do not appear often either in the underlying knowledge graph, or in the LLM-generated set of event sequences. Consequently, SuMMs fail to identify logical influencing sets. For example, for the event type `United States Presidential Impeachment`, BSuMM model identified `unequal treaty` as the influencing event.

6 Conclusions and Future Work

In this paper, we presented methods of distilling event sequence knowledge from large language models. We first presented an approach of generating high-quality event sequences through knowledge-guided generation using LLMs. We then conducted a three-pronged quantitative and qualitative evaluation of our results. First, we evaluated LLM-generated event sequences in absolute terms through manual evaluation and automated evaluation using standard accuracy metrics. Second, we looked at some qualitative examples of *newly* identified high support sequences not already present in our base knowledge graph i.e. Wikidata. Finally, we gauge how probabilistic models like SuMMs fare when trained on LLM-generated data. We found that through our carefully-designed elicitation procedure, LLMs are capable of producing high-quality event sequences that can be used for the downstream tasks of pattern mining and the construction of probabilistic event models such as SuMMs.

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. Here, we outline a number of directions for future work:

– In this paper, we used Wikidata as our base source of knowledge. The event concepts found on Wikidata often appear in the pre-training 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.
– 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 for evaluating model performance. However, an ideal and accurate evaluation of model outputs should involve the use of human annotators and a validation of the use of larger LLMs as a part of LLMs-as-a-judge [47] evaluation strategy. Furthermore, our use of smaller models was justified in part due to the simplicity of the precision evaluation task, which can be posed as a binary classification task. It will be interesting to evaluate the correctness of post-hoc explanations generated by models for their corresponding output during precision evaluation, as such explanations are key to the usefulness of the distilled knowledge, particularly in critical decision support applications.
– In the context of our domain of interest in this paper (news event analysis), an exciting avenue for future research is to study the effect of utilizing structured knowledge distilled from LLMs in a pipeline for future event prediction, and a comparison with the performance of human forecasters. Recent work [17] claims that LLMs, without symbolic knowledge elicitation, come close and in some cases surpass the ability of a "crowd aggregate of competitive forecasters" in making accurate forecasts in the form of answering forecast questions on online platforms such as the Good Judgment Open ((https://www.gjopen.com)). Two research questions arise: 1) Could a neurosymbolic approach utilizing distilled symbolic knowledge complement or outperform a purely LLM-based solution? 2) Would distilled high-quality structured knowledge provide better explainability and therefore be a more reliable tool as a part of a human-AI collaboration mechanism for event forecasting?

Supplemental Material Statement: Our supplementary material is available at https://doi.org/10.5281/zenodo.13308920 and includes code and prompts used for event sequence generation and benchmarking, as well as the base KG and sample output files. `README.md` has all the details. `Appendix.pdf` contains our prompts and human evaluation details.

References

1. Agrawal, S., Zhou, C., Lewis, M., Zettlemoyer, L., Ghazvininejad, M.: In-context examples selection for machine translation. In: Findings of the Association for Computational Linguistics: ACL 2023, pp. 8857–8873. Association for Computational Linguistics, Toronto, Canada (2023). https://doi.org/10.18653/v1/2023.findings-acl.564
2. Allan, J.: New York, 1st edn. Springer (2012). https://doi.org/10.1007/978-1-4615-0933-2
3. Allan, J., Carbonell, J.G., Doddington, G., Yamron, J., Yang, Y.: Topic detection and tracking pilot study final report. In: Proceedings of the DARPA Broadcast News Transcription and Understanding Workshop, February, 1998 (1998). https://doi.org/10.1184/R1/6626252.v1
4. An, S., et al.: 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, pp. 11027–11052. Association for Computational Linguistics, Toronto, Canada (2023). https://doi.org/10.18653/v1/2023.acl-long.618
5. Begleiter, R., El-Yaniv, R., Yona, G.: On prediction using variable order Markov models. J. Artif. Intell. Res. **22**, 385–421 (2004). https://doi.org/10.1613/jair.1491
6. Bhattacharjya, D., Sihag, S., Hassanzadeh, O., Bialik, L.: Summary Markov models for event sequences. In: Raedt, L.D. (ed.) Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22, pp. 4836–4842. International Joint Conferences on Artificial Intelligence Organization (2022). https://doi.org/10.24963/ijcai.2022/670
7. Brown, T., et al.: Language models are few-shot learners. In: Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., Lin, H. (eds.) Advances in Neural Information Processing Systems, vol. 33, pp. 1877–1901. Curran Associates, Inc. (2020). https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf
8. Cekinel, R.F., Karagoz, P.: Event prediction from news text using subgraph embedding and graph sequence mining. World Wide Web **25**, 2403–2428 (2022). https://doi.org/10.1007/s11280-021-01002-1
9. Chen, M., et al.: Event-centric natural language processing. In: ACL (2021)
10. Chiang, C.H., Lee, H.y.: 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, pp. 15607–15631. Association for Computational Linguistics, Toronto, Canada (2023).https://doi.org/10.18653/v1/2023.acl-long.870
11. Chung, H.W., et al.: Scaling instruction-finetuned language models (2022)
12. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: 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, pp. 4171–4186. Association for Computational Linguistics, Minneapolis, Minnesota (2019). https://doi.org/10.18653/v1/N19-1423
13. Du, X., et al.: 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, pp. 54–63. Association for Computational Linguistics, Seattle, Washington (2022). https://doi.org/10.18653/v1/2022.naacl-demo.7

14. Filatova, E., Hatzivassiloglou, V., McKeown, K.: Automatic creation of domain templates. In: Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pp. 207–214. Association for Computational Linguistics, Sydney, Australia (2006). https://aclanthology.org/P06-2027
15. Fournier-Viger, P., Lin, J.C.W., Kiran, R.U., Koh, Y.S.: A survey of sequential pattern mining. Data Sci. Pattern Recogn. **1**(1), 54–77 (2017)
16. Gholipour Ghalandari, D., Ifrim, G.: Examining the state-of-the-art in news timeline summarization. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 1322–1334. Association for Computational Linguistics (2020). https://doi.org/10.18653/v1/2020.acl-main.122
17. Halawi, D., Zhang, F., Yueh-Han, C., Steinhardt, J.: Approaching human-level forecasting with language models (2024)
18. Hassanzadeh, O., et al.: 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, pp. 5904–5907 (2022). https://doi.org/10.24963/ijcai.2022/850
19. He, X., et al.: AnnoLLM: making large language models to be better crowdsourced annotators (2023)
20. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997). https://doi.org/10.1162/neco.1997.9.8.1735
21. Hwang, J.D., et al.: COMET-ATOMIC 2020: on symbolic and neural commonsense knowledge graphs. In: AAAI (2021)
22. Imani, S., Du, L., Shrivastava, H.: MathPrompter: mathematical reasoning using large language models. In: Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics, Volume 5: Industry Track, pp. 37–42. Association for Computational Linguistics, Toronto, Canada (2023). https://doi.org/10.18653/v1/2023.acl-industry.4
23. Kamalloo, E., Dziri, N., Clarke, C., Rafiei, D.: 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, pp. 5591–5606. Association for Computational Linguistics, Toronto, Canada (2023). https://doi.org/10.18653/v1/2023.acl-long.307
24. Li, M., et al.: 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), pp. 684–695. Association for Computational Linguistics (2020). https://doi.org/10.18653/v1/2020.emnlp-main.50
25. Li, S., Zhao, R., Li, M., Ji, H., Callison-Burch, C., Han, J.: 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, pp. 5677–5697. Association for Computational Linguistics, Toronto, Canada (2023). https://doi.org/10.18653/v1/2023.acl-long.312
26. Mabroukeh, N.R., Ezeife, C.I.: A taxonomy of sequential pattern mining algorithms. ACM Comput. Surv. **43**(1) (2010). https://doi.org/10.1145/1824795.1824798
27. Mannila, H., Toivonen, H., Verkamo, A.I.: Discovery of frequent episodes in event sequences. Data Min. Knowl. Disc. **1**, 259–289 (1997)
28. Mooney, C.H., Roddick, J.F.: Sequential pattern mining – approaches and algorithms. ACM Comput. Surv. **45**(2) (2013). https://doi.org/10.1145/2431211.2431218

29. Norambuena, B.K., Mitra, T., North, C.: A survey on event-based news narrative extraction. ACM Comput. Surv. **55**(14s) (2023). https://doi.org/10.1145/3584741
30. Radinsky, K., Davidovich, S., Markovitch, S.: Learning causality for news events prediction. In: Proceedings of the 21st World Wide Web Conference 2012, WWW 2012, Lyon, France, April 16-20, 2012, pp. 909–918. ACM (2012). https://doi.org/10.1145/2187836.2187958
31. Raffel, C., et al.: Exploring the limits of transfer learning with a unified text-to-text transformer. J. Mach. Learn. Res. **21**(1) (2020)
32. Raftery, A.: A model for high-order Markov chains. J. Roy. Stat. Soc. B **47**(3), 528–539 (1985)
33. Santana, B.S., Campos, R., Amorim, E., Jorge, A., Silvano, P., Nunes, S.: A survey on narrative extraction from textual data. Artif. Intell. Rev. **56**(8), 8393–8435 (2023). https://doi.org/10.1007/s10462-022-10338-7
34. Srikant, R., Agrawal, R.: Mining sequential patterns: generalizations and performance improvements. In: Apers, P., Bouzeghoub, M., Gardarin, G. (eds.) EDBT 1996. LNCS, vol. 1057, pp. 1–17. Springer, Heidelberg (1996). https://doi.org/10.1007/BFb0014140
35. Touvron, H., et al.: LLaMA: open and efficient foundation language models. arXiv preprint arXiv:2302.13971 (2023)
36. Vrandečić, D., Krötzsch, M.: Wikidata: a free collaborative knowledgebase. Commun. ACM **57**(10), 78–85 (2014). https://doi.org/10.1145/2629489
37. Wadhwa, S., Amir, S., Wallace, B.: 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, pp. 15566–15589. Association for Computational Linguistics, Toronto, Canada (2023). https://doi.org/10.18653/v1/2023.acl-long.868
38. Wei, J., et al.: Finetuned language models are zero-shot learners. In: International Conference on Learning Representations (2022). https://openreview.net/forum?id=gEZrGCozdqR
39. West, P., et al.: 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, pp. 4602–4625. Association for Computational Linguistics, Seattle, United States (2022). https://doi.org/10.18653/v1/2022.naacl-main.341
40. Wolf, T., et al.: Transformers: state-of-the-art natural language processing. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, pp. 38–45. Association for Computational Linguistics (2020). https://doi.org/10.18653/v1/2020.emnlp-demos.6
41. Xiang, W., Wang, B.: A survey of event extraction from text. IEEE Access **7**, 173111–173137 (2019). https://doi.org/10.1109/ACCESS.2019.2956831
42. Yu, W., et al.: A survey of knowledge-enhanced text generation. ACM Comput. Surv. **54**(11s) (2022). https://doi.org/10.1145/3512467
43. Zaki, M.J.: SPADE: an efficient algorithm for mining frequent sequences. Mach. Learn. **42**, 31–60 (2004). https://api.semanticscholar.org/CorpusID:5387869
44. Zhang, Y., Feng, S., Tan, C.: Active example selection for in-context learning. In: Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, pp. 9134–9148. Association for Computational Linguistics, Abu Dhabi, United Arab Emirates (2022). https://doi.org/10.18653/v1/2022.emnlp-main.622
45. Zhao, L.: Event prediction in the big data era: a systematic survey. ACM Comput. Surv. **54**(5) (2021). https://doi.org/10.1145/3450287

46. Zhao, M., et al.: LMTurk: few-shot learners as crowdsourcing workers in a language-model-as-a-service framework. In: Findings of the Association for Computational Linguistics: NAACL 2022, pp. 675–692. Association for Computational Linguistics, Seattle, United States (2022). https://doi.org/10.18653/v1/2022.findings-naacl.51
47. Zheng, L., et al.: Judging LLM-as-a-judge with MT-bench and chatbot arena (2023)

□