Zero-Shot On-the-Fly Event Schema Induction

Rotem Dror[*], **Haoyu Wang**[*], **and Dan Roth**
Department of Computer and Information Science
University of Pennsylvania
{rtmdrr,why16gzl,danroth}@seas.upenn.edu

Abstract

What are the events involved in a pandemic outbreak? What steps should be taken when planning a wedding? The answers to these questions can be found by collecting many documents on the complex event of interest, extracting relevant information, and analyzing it. We present a new approach[1] in which large language models are utilized to generate source documents that allow predicting, given a high-level event definition, the specific events, arguments, and relations between them to construct a schema that describes the complex event in its entirety. Using our model, complete schemas on any topic can be generated on-the-fly without any manual data collection, i.e., in a zero-shot manner. Moreover, we develop efficient methods to extract pertinent information from texts and demonstrate in a series of experiments that these schemas are considered to be more complete than human-curated ones in the majority of examined scenarios. Finally, we show that this framework is comparable in performance with previous supervised schema induction methods that rely on collecting real texts and even reaching the best score in the prediction task.

1 Introduction

Event processing refers to tracking, analyzing, and drawing conclusions from streams of information about events. This event analysis aims at identifying meaningful events (such as opportunities or threats) in real-time situations and responding appropriately. Event processing can also be utilized to gain a deep understanding of the specific steps, arguments, and relations between them that are involved in a complex event. The information above can be consolidated into a graphical representation called an *event schema* (Li et al., 2021). For instance in Fig. 1, the graph representation of events

---

[*] Indicating equal contribution.
[1] https://cogcomp.seas.upenn.edu/page/publication_view/995

and participants assists in gaining an understanding of the complex event of kidnapping and could help composing a reaction plan if needed.

The NLP community has devoted much effort to understanding events that are described in a document or in a collection of documents for this purpose. These efforts include identifying event triggers (Lu and Roth, 2012; Huang et al., 2018; Wadden et al., 2019; Han et al., 2019), extracting event arguments (Punyakanok et al., 2008; Peng et al., 2016; Lin et al., 2020; Zhang et al., 2021a), and predicting the relations between events, e.g., temporal, coreferential, causal or hierarchical relations (Do et al., 2012; Lee et al., 2012; Glavaš et al., 2014; Ning et al., 2018; Wang et al., 2020; Zhang et al., 2020a; Trong et al., 2022).

Previous works on event schema induction relied on the information extracted from manually collected documents to build the schema graph. For instance, Li et al. (2020) learn an auto-regressive language model (LM) over paths in the instance graphs depicting events, arguments and relations of instances of the complex events, and then construct a schema graph by merging the top k ranked paths. Their approach, however, requires access to many documents on each topic of interest, which can be extremely laborious and time consuming to obtain.

In this paper, our goal is to allow creating schemas on-the-fly by taking as input only the name of the complex event of interest (like a "pandemic outbreak" or an "armed robbery"). To avoid manually collecting many documents on the topic of the schema, we utilize pre-trained text generators, e.g., GPT-3 (Brown et al., 2020), to obtain documents of diverse genres on the desired topic (examples presented in Fig. 2). These documents are then processed to extract pertinent information from which a schema is constructed. The fact that we do not collect any data makes our learning framework zero-shot since we do not rely on any human-collected articles or example schemas.
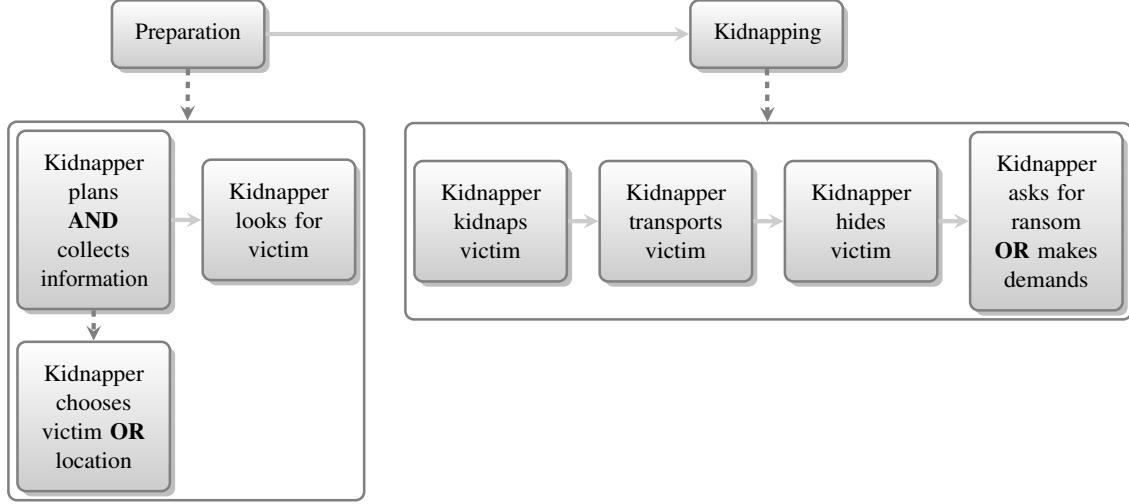
Figure 1: An example schema for the event of Kidnapping. The regular arrows represent temporal relations and the dashed arrows represent hierarchical relations (PARENT-CHILD).

In addition to eliminating the need to collect data, we also made the information extraction process faster by implementing new and efficient methods for identifying temporal and hierarchical relations between events mentioned in the text. These two steps are the most time consuming in the process of schema induction and could take up to 2 hours each using state-of-the-art models proposed by Zhou et al. (2021); Wang et al. (2021). Sending the whole text as input instead of two sentences at each time, our proposed model shortens the inference time significantly to several minutes without enduring a major loss in performance.

The process of generating texts is explained in Section §3, and the process of extracting relevant and salient information is described in Section §4, then we introduce the construction of schema graphs in Section §5. To evaluate our zero-shot schema generator we conduct experiments on a benchmark dataset for schema induction, LDC2020E25, and provide a new dataset for further evaluation called Schema-11. Additionally, we design a subject-matter expert Turing test, a.k.a. Feigenbaum test (Feigenbaum, 2003), to determine whether our algorithm could mimic experts' response. We also demonstrate that documents generated by GPT-3 are informative and useful for the task of schema induction. The experiments and results are presented in Section §6. The contributions of our work include:

1. Predicting an entire schema given the name of a complex event without collecting data.

2. Implementing a novel and efficient One-Pass

approach for identifying temporal and hierarchical relations between events.

3. Presenting a method for automatically inducing logical relations between events based on temporal relations.

4. Offering a Feigenbaum test for evaluation on a new schema dataset, Schema-11.

2 Related Work

Schema Induction: Early schema induction efforts focused on identifying the triggers and participants of atomic events without considering relations between atomic events that comprise complex schemas (Chambers, 2013; Cheung et al., 2013; Nguyen et al., 2015; Sha et al., 2016; Yuan et al., 2018). More recent work focuses on inducing schemas for pairs of events (Li et al., 2020) and multiple events (Zhang et al., 2021b; Li et al., 2021), but they require access to large corpora for the induction process. In this work, we induce schemas on-the-fly in a zero-shot manner. As is standard in state-of-the-art (SOTA) works (Li et al., 2020, 2021; Wen et al., 2021; Lawley and Schubert, 2022), we output all the essential information about relations between events and arguments extracted from the text, in addition to logical and hierarchical relations not studied previously in schema induction.

Script Learning: Early script learning work concentrated on chains of events with a single protagonist (Chambers and Jurafsky, 2008, 2009;

Jans et al., 2012; Rudinger et al., 2015; Granroth-Wilding and Clark, 2016) and later extended to multiple protagonists (Pichotta and Mooney, 2014; Peng and Roth, 2016; Pichotta and Mooney, 2016; Modi, 2016; Weber et al., 2018, 2020; Zhang et al., 2020b). All of these works assume there exists a single line of events that describes all occurrences within a complex event. This work does not limit itself to generating single-chained schemas. We also consider more complex graphs as schema outputs. In addition, none of these works deal with zero-shot scenarios that do not require training data.

Pre-Trained Generation Models: Large-scale pre-trained text generation models such as GPT-2 (Radford et al., 2019), GPT-3 (Brown et al., 2020), BART (Lewis et al., 2020), T5 (Raffel et al., 2020), i.a. have been used in many NLP tasks. These models are often seen as few-shot learners (Brown et al., 2020) and therefore used as inference methods. However, these text generation models are not explicitly trained to perform inference, but to produce the most likely sequence of words to proceed a certain prompt, similar to language models. In our work, we use these large pre-trained LMs as text generators. The generated documents on a particular topic are leveraged as a corpus for extracting the schema of the given topic. We rely on the intuition that the generated text will include salient and stereotypical information that is expected to be mentioned in the context of the topic (e.g., for the topic of "planning a wedding," we assume most documents will include "order catering").

3   Data Generation

The schema induction process begins with generating texts using large LMs as text generation models. These texts are joined to form a knowledge base for the schema, including all of the potential information that the schema may present. One could, of course, create this knowledge base by crawling the web for real news articles or Wikipedia entries related to a certain topic.

We argue, however, that in addition to the obvious advantages of not having to rely on the availability of data online and not having to crawl the entire web for relevant documents on each topic, the generated data from these large generative models is more efficient in reporting salient events than random events described in the news, i.e., generated texts are more likely to mention important information than real documents do.

	Generated Text	Real Text
# events / # tokens	12.52%	6.31%
# arguments / # tokens	5.45%	3.01%

Table 1: The ratio of relevant events and relevant argument roles identified in generated texts and real texts for the scenario of IED attack.

Our analysis shows that the generated stories contain a higher percentage of relevant tokens than real news articles that are used for schema induction. To demonstrate this phenomenon, we compare manually collected documents with those that are automatically generated using GPT-3 for the event of Improvised Explosive Device (IED) Attack (Li et al., 2021). To identify salient events and arguments concerning IED attacks, we adopt the DARPA KAIROS Phase 1 (v3.0) ontology[2] — a fine-grained ontology for schema learning, with 24 entity types, 67 event types, and 85 argument roles.

We calculate the number of relevant event triggers and arguments identified in the text, where a relevant mention is one whose type appears in the ontology. The results shown in Table 1 demonstrate that the quality of the generated texts in terms of conciseness and appearance of important details is higher than that of real texts. For example, the ratio of relevant events per token is more than twice as high in generated texts as it is in real texts. Hence we are able to not only generate a schema for every given topic without putting any effort in searching the web, but the information we generate is also better suited for our end task of depicting all of the important aspects of a complex event.

Given a topic for which we want to create a schema, we generate multiple texts that discuss the topic event using the OpenAI GPT-3 API[3] with the Davinci-instruct-beta-v3 model and we also experiment with the Hugging Face GPT-2 API[4]. We use three prompting methods to generate documents of diverse genres as follows:

News Articles: We begin by generating a headline using the prompt: "Write a news headline about *topic*." The output from this prompt is then used in the following prompt: "Write a news story titled *headline*." The output from the second prompt is added to the pool of generated texts. The process is repeated 30 times. See example in Fig. 2b.

---

[2]The full ontology definition can be accessed at this link: https://bit.ly/3mIWJoN.
[3]https://openai.com/blog/openai-api/.
[4]https://huggingface.co/gpt2

Prompt: What are the steps involved in holding a summit meeting? 1. **Output**: 1. Plan the agenda and objectives of the meeting. 2. Select a venue that is appropriate for the meeting's purpose and size. 3. Send out invitations and ensure all participants have the appropriate information. 4. Prepare for the meeting by creating materials and briefing the participants. 5. Hold the meeting and follow up on any action items that are generated.	**First prompt**: Write a news headline about filing a patent. **Output**: New Patent Filing Could Speed Up Drug Development. **Second prompt**: Write a news story titled "New Patent Filing Could Speed Up Drug Development". **Output**: The United States Patent and Trademark Office has just published a new patent that could speed up the drug development process. The patent, filed by a team of researchers from the DPharm Institute, covers a new drug screening process that could help identify potential new drugs much more quickly. The new process, which is based on a computer algorithm, is said to be up to 10 times faster than traditional methods. It can identify potential new drugs by screening through large databases of existing drugs and drug candidates. The DPharm Institute is already in the process of licensing the patent to several pharmaceutical companies, and is planning to commercialize the technology over the next few years.
(a) Example generated text of steps involved in the event of "holding a summit meeting".	(b) Example generation of news-like text for the event of "filing a patent".

Prompt: Describe how to 3D print an object?
Output: To 3D print an object, you need to first design the object in a 3D modeling program. Once the object is designed, you need to export it as a 3D printable file format. Next, you need to choose a 3D printer and print settings. Finally, you need to print the object.

(c) Example generation of How-To article for the event of "3D printing".

Figure 2: Examples of generated texts using different prompting methods. The highlighted tokens display relevant events that will be extracted in the information extraction step.

How-To Articles: We use the prompt: "Describe how to *topic*." to generate wikiHow-like instruction articles. The process is repeated 30 times. See example in Fig. 2c.

Direct Step-by-Step Schema: We use the prompt: "What are the steps involved in *topic*? 1."[5] to directly generate a schema. We run this process once. See example in Fig. 2a.

Generating documents of various genres enables our model to induce comprehensive schemas on any given topics. Considering that some events are more likely to be in the news (e.g., elections, pandemic outbreaks) while others are more technical in nature and are hence less newsworthy (such as earning a Ph.D. degree or planning a wedding), we generate diverse texts and then use a ranking model to choose the most relevant documents.

The ranking process includes embedding the texts and the topic with the model proposed in Reimers and Gurevych (2019), and then calculating the cosine similarity between each text and the topic embeddings. Only the 30 texts closest to the topic are selected, together with the output from the direct step-by-step schema. The following section describes the next step in generating a schema of extracting relevant information from the texts.

4 Information Extraction

For each document, we extract event triggers, arguments and relations between the events that are important and relevant to the schema topic. We do not work with a predefined ontology that defines what events and arguments are salient in advance because we allow generating a schema on any topic. Instead, we employ a statistical approach by extracting all the information and later filter it down to include just frequent items. Here are the steps involved in our information extraction pipeline:

Semantic Role Labeling (SRL): We use the SOTA SRL system[6] trained on CoNLL12 (Pradhan et al., 2012) and Nombank dataset (Meyers et al., 2004) to extract both verb and nominal event triggers and arguments.

Named Entity Recognition (NER): We employ the SOTA NER model (Guo and Roth, 2021) to extract and map entities (potential arguments of events) into entity types defined in the CoNLL 2002

---

[5]The "1." in the prompt is for the LM to automatically complete the steps.

[6] https://cogcomp.seas.upenn.edu/page/demo_view/SRLEnglish

dataset (Tjong Kim Sang, 2002) and the LORELEI project (Strassel and Tracey, 2016).

Constituency Parsing: The arguments extracted by SRL can be clauses and long phrasal nouns, hence we employ the AllenNLP[7] constituency parsing model for argument head word extraction.

Coreference Resolution: We use the SOTA model (Yu et al., 2022) for event and entity coreference resolution to identify within-document coreferential relations.

Temporal Relation Extraction: We first try to use SOTA models (Ning et al., 2019; Zhou et al., 2021) to predict the temporal relations[8] between all possible pairs of extracted events but since the SOTA models accept two sentences containing events as input, the inference time[9] for an n-event document is $\mathcal{O}(n^2)$, making the schema induction process several hours long.

One-Pass Model: We develop a One-Pass model that takes the document as input and uses the contextual representation of events to predict relations between them. A document D is represented as a sequence of tokens $D = [t_1, \cdots, e_1, \cdots, e_2, \cdots, t_n]$ where some of the tokens belong to the set of annotated event triggers, i.e., $\mathcal{E}_D = \{e_1, e_2, \cdots, e_k\}$, whereas the rest are other lexemes. We employ the transformer-based language model Big Bird (Zaheer et al., 2020) to encode a whole document and obtain the contextualized representations for all the event mentions. These representations are fed into a multi-layer perceptron in a pairwise fashion and the cross-entropy loss for each pair is calculated and accumulated for a batch of documents. As shown in Tab. 2, the inference time is shortened 63-186 times on average, while the performance of the One-Pass model is comparable to SOTA models.

Hierarchical Relation Extraction: The extremely long inference time of SOTA models for predicting hierarchical relations (PARENT-CHILD, CHILD-PARENT, COREF, NOREL) (Zhou et al., 2020; Wang et al., 2021) also impairs the efficiency of our schema induction system. Thus we use

Corpus	Model	Metrics		
		F_1 score	Speed	GPU Memory
HiEve	Zhou et al. (2020)	0.489	-	-
	Wang et al. (2021)	0.522	41.68s	4515MiB
	One-Pass model	0.472	0.65s	2941MiB
MATRES	Ning et al. (2019)	0.767	30.12s	4187MiB
	Zhou et al. (2021)	0.821	89.36s	9311MiB
	One-Pass model	0.768	0.48s	2419MiB

Table 2: Performance comparison between the One-Pass model and SOTA models for event temporal and hierarchical relation extraction. We report F_1 scores on benchmark datasets (HiEve for hierarchical relations, MATRES for temporal relations), speed (average inference time for 100 event pairs), and required GPU memory during inference. The One-Pass models are 63-186 times faster than SOTA models and take up only 26%-65% of the GPU memory required by SOTA models.

the same One-Pass methodology to extract hierarchical relations. We observe that the inference time is greatly shortened, and the One-Pass model achieves comparable results to previous models while taking up less GPU memory (see Tab. 2).

After processing the data using the procedure described above, we get a list of events, their arguments, and relations between the events. We concentrate on events and relations that frequently appear in the generated texts since we assume those are the most important to add to the schema (without any other source of information that could identify what is salient). We describe the process of building a schema in the following section.

5 Schema Induction

To consolidate the information extracted from the previous step, we build a schema as follows:

Make a list of events and relations: To compare similar event mentions in different texts, we compare the event trigger itself (whether they are the same verb or coreferential verbs[10]) and the NER types of its arguments. For example, the trigger "(take) precautions" appeared in 5 documents generated for the topic of Pandemic Outbreak. In two documents the subject of the verb phrase "take precautions" was "residents", in another two it was "people" and in the last one, it was "public". Nevertheless, the NER type is identical in all cases (PER), and thus we set the frequency of "(take) precautions" to 5. Similarly, we calculate the frequency of the temporal and hierarchical relations. We only consider relations and events that appeared

---

[7]https://demo.allennlp.org/constituency-parsing.

[8]The possible temporal relations (start-time comparison) are: BEFORE, AFTER, EQUAL and VAGUE.

[9]The inference time is mostly spent on obtaining the contextual representation of events using large fine-tuned LMs.

[10]We only consider coreferential and hierarchical relations if they appear in more than 2 documents.
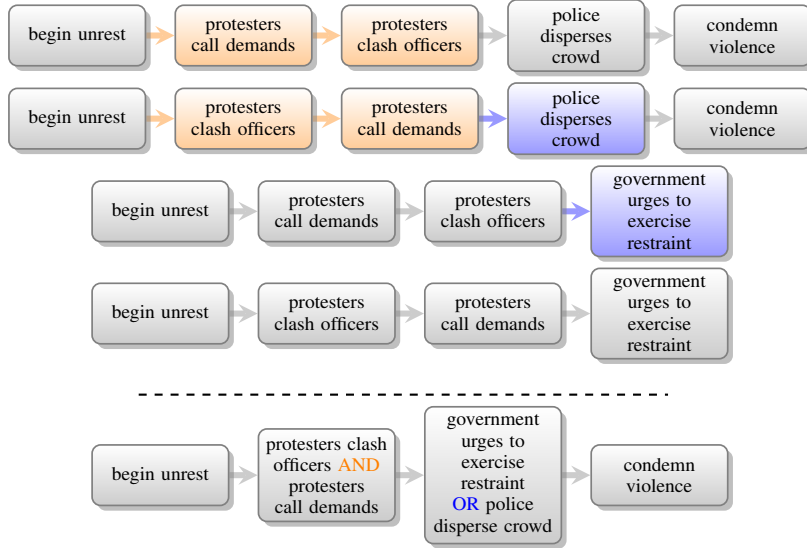
Figure 3: An example of integrating timelines and logical relations in the schema of Civil Unrest. The four upper timelines are the ones extracted from the generated texts and the lower one is their merger into a single timeline with logical relations.
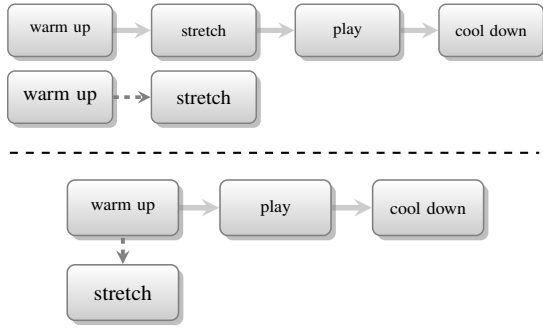

Figure 4: Example of the procedure to amend a timeline in the schema of "Sports Games". The timeline at the top that includes events from different levels ("warm up" is the parent of "stretch") is fixed below. Gray arrows mark temporal relations, and dashed arrows mark PARENT-CHILD.

in more than one document.

Construct timelines: We construct the longest timelines from the list of temporal relations. This list is a list of tuples (A, B), indicating that event A happened before event B. To construct a timeline, we search recursively for the longest chains of the following form $(A, B), (B, C), (A, C)$ and so on.

Fix timelines according to hierarchical relations: We build a hierarchy of the events using the hierarchical relation list[10] and change the timelines so that they will only include events that appear in the same level of hierarchy (see example in Fig. 4).

Add logical relations: The final step is to combine the timelines and hierarchies into a single schema

graph using logical relations (AND/OR). When observing two timelines with discrepancies between the order of events, we place a logical AND between them, since we interpret this discrepancy as both events occurring at the same time or there is no significance to the order between them. We use a logical OR to mark events that can occur simultaneously but not necessarily. See Fig. 3 for example of both logical relations.

The final output is a schema graph that contains all the events, arguments, and temporal, hierarchical and logical relations between the events. It is noteworthy that our proposed schema generation model can be easily used to *extend the scope of existing schemas* by further querying the model on more specific topics. For example, the schema in Fig. 1 does not cover the consequences of kidnapping, probably because the LM did not attend to this aspect. Hence an analyst can input another topic (e.g., consequences of kidnapping) to further develop the schema. Similarly, analysts can generate schemas for very specific events (e.g., kidnapping in a political setting). Next, we provide an in-depth experimentation for the proposed schema induction framework.

6 Experiments

6.1 Data

We conduct experiments on a dataset for general schema learning released by LDC (LDC2020E25). The corpus includes 84 types of complex events,

such as Cyber Attack, Farming and Recycling. This dataset includes ground-truth schemas created by LDC annotators. In addition, we also collected human generated schemas for 11 newsworthy scenarios[11]. The schemas were generated by four human experts who were instructed to write a schema on each topic based on their commonsense knowledge that includes a list of event triggers, event arguments and their NER types[12], and relations[13].

6.2 Evaluation Protocols

We follow Li et al. (2021) to use instance coverage and last event prediction to evaluate our method on the LDC dataset. For the Schema-11 dataset, we ask human testers to assess the completeness and soundness of both human- and automatically-generated schemas.

Coverage and Prediction A common evaluation method in schema induction and script prediction is to calculate the recall of events and relations predicted by the model, assuming the human annotations are gold labels (coverage), and to calculate the accuracy in predicting the final outcome of a scenario (prediction). For instance, the accuracy of predicting the last event type of the LDC schemas is reported in Li et al. (2021). Here we present the results of predicting the last events using event triggers instead of event types since our schemas do not use an ontology of event types.

Feigenbaum Test We show human testers two schemas on each topic in the Schema-11 dataset (see example in Appx. §A). One schema is automatically generated by our model, and the other is randomly sampled from the Schema-11 corpus[14]. Then, we ask the testers to determine which events and relations are valid to appear in the schema (soundness), and answer the following questions: which schema is more complete in the sense of including all the events needed to describe the topic, and which schema, in their opinion, was generated

by a human expert (as opposed to a machine).

6.3 Results

Coverage We calculate the intersection between events in the generated schemas and the gold schemas in two ways: (a) the matching of event triggers, and (b) the matching of event triggers and synonyms of the events in the gold schemas (synonym coverage)[15]. We believe that synonym coverage is a better evaluation metric to avoid errors such as considering different verbs describing the same action as different (e.g., "buy" and "acquire") than using a predefined ontology of event types such as the one used in Li et al. (2021). The reason is twofold: firstly, any predefined ontology is limited to certain scenarios and it may impair the variety of events extracted; and secondly the typing mechanism may also inflict errors to the schema. In the calculation of coverage of relations we only take into account relations (a, b) where both events, a and b, appear in the generated schema.

From the results in Table 3, we observe that despite the difficulty of exact matching, our model with GPT-3 covers 23.73% of the gold events, showing that generated texts are useful. If we use synonym coverage as our metric, we achieve a promising coverage of 37.84% while the SOTA *supervised* event graph model (Li et al., 2021) covers 54.84% using limited event types. In addition, we calculated an average number of 26.19 additional events that appeared in the generated schemas and not in the LDC schema, pointing to the potential of using generated documents for expending existing schemas. With the high quality event representations obtained from the One-Pass model and the proposed logical relation induction algorithm, our method can successfully cover a high percentage of multiple types of relations.

Prediction In the prediction task, our schemas are able reach SOTA performance and predict the final outcome in 63.1% of the cases for the LDC schemas (see Tab. 4). This result is extremely impressive when it is compared with Li et al. (2021) since they predict event types instead of verbs, which is a much easier task due to the fact that the set of possible answers is limited.

Schema-11 In the soundness experiments, where the testers are asked to decide which events and relations are valid to appear in the schema, it turns out

---

[11]The topics are: Bombing Attack, Business Change, Civil Unrest, Disaster and Rescue, Elections, International Conflict, Kidnapping, Mass Shooting, Pandemic Outbreak, Sports Games, and Terrorism Attack.

[12]The annotators are familiar with SRL annotations (e.g., ARG0, ARG1, etc.) and NER types (e.g., PER, ORG, etc.). See additional details in App. C

[13]No restrictions were placed for the annotators. For example, in one case, an annotator mentioned causal relations that are not covered in our framework.

[14]In some cases we combine two randomly sampled schemas because the length of the human schemas tend to be shorter than the automatically generated ones.

[15]Implemented using the NLTK WordNet Python package.

| | GPT2 | | GPT3 | | Li et al. (2021) |
	Coverage	Coverage (Syn)	Coverage	Coverage (Syn)	Coverage
Event Match	14.88	29.55	23.73	37.84	54.84
Temporal Relations	10.80	33.31	31.07	49.99	
Hierarchical Relations	33.33	33.33	11.11	13.88	-
Logical Relations	4.16	24.99	43.76	49.81	

Table 3: Coverage results for the LDC dataset. The first row presents the percentage of events that appeared in both the LDC schemas and the automatically generated schemas (out of events in LDC schemas), and the three bottom rows present the same metric for relations of different types.

Model	Accuracy
Event Language Model	49.7
Sequential Pattern Mining	47.8
Human Schema	20.5
Event Graph Model	52.0
Zero-Shot Schema GPT2	25.0
Zero-Shot Schema Synonym GPT2	45.2
Zero-Shot Schema GPT3	35.7
Zero-Shot Schema Synonym GPT3	**63.1**

Table 4: Experimental results for last event prediction in the LDC dataset. The top 4 results are from (Li et al., 2021), and the metric is HITS@1 where the events are typed based on a predefined ontology.

	S1	**S2**	**S3**	**S4**	S5	S6	S7	S8	**S9**	S10	S11
Human	4	0	1	1	1	2	1	1	0	3	1
Automatic	2	3	4	2	1	1	1	1	4	0	1

Table 5: Distribution of votes for which is the more complete schema for Schema-11 dataset.

that human-schemas contain 7.14% invalid events and 15.4% invalid relations on average. For the automatically-generated schemas, 6.06% of the events and 22.9% of the relations are considered to be invalid on average, meaning that the average percentage of legitimate events is even higher in machine-generated schemas. For the completeness results presented in Tab. 5, in 4 cases the testers agreed that the automatically generated schemas are more complete; in 3 cases they claimed that the human schemas are more complete; and a tie in the remaining 4 cases. Hence our automatically generated schemas are of comparable quality to human generated ones in the sense of completeness.

Finally, in the Feigenbaum test, where testers are asked to decide whether a schema is generated by a human or a machine, eight out of eleven times they correctly identify the human-generated schema, one incorrectly, and two ties. Some of the testers who succeeded in their guesses mentioned that it was easy to determine which schema was automatically generated since it tends to be longer and comprehensive. The full results from the Feigenbaum test are shown in Appx. §B.

Wizard of Oz Experiment There seems to be a discrepancy between the low event coverage results and the quality of generated texts that were presented in Section §3. We, therefore, conducted

another experiment to identify if the problem stems from the quality of the generated documents. In this experiment, one of the authors sampled 10 complex event names from the LDC dataset and generated, using GPT-3 text davinci-002 model, 3 texts for each scenario using the prompting methods presented in Section §3. Then, the author manually extracted all relevant events and relations from each document and built a schema based solely on those events and relations.

This experiment, in which the author pretends to be the IE and schema generator models, aims to demonstrate that if we had perfect IE and schema induction systems, then the texts generated by GPT-3 would be sufficient and even superior to other corpora collected manually. The macro-average coverage of events in this experiment is 68% and the micro-average is 74%. Furthermore, GPT-3 texts generated schemas that included, on average, 6.5 additional events not mentioned in LDC schemas but relevant to the scenario at hand. As a result, we conclude that the generated texts from GPT3 contain much of the necessary information to generate schemas in a variety of topics, and can even be used to enrich existing schemas generated by other models or humans. Two example scenarios and more details appear in Appx. §D.

7 Conclusion

We propose a method to generate schemas given the sole input of a topic. We use GPT-3 to generate texts of diverse genres and a pipeline of information extraction tools to obtain relevant information before inducing logical relations and integrating the events and relations into a schema graph. To improve the efficiency of the pipeline, we imple-

ment One-Pass models for identifying temporal and hierarchical relations that achieve comparable performances with SOTA models but require far less inference time and memory space. To evaluate our framework, we conduct experiments on a benchmark LDC dataset to show that our schemas cover a decent amount of pertinent information and display comparable ability for event prediction with supervised approaches. We observe a high percentage of valid events and relations generated for the Schema-11 dataset and the testers endorsed the completeness of our machine-generated schemas.

8 Acknowledgments

9 Limitations

The paper presents a method for building an event schema without manually collecting documents from sources such as news articles or Wikipedia. In order to generate diverse and informative documents on any topic, we rely on large pre-trained language models. Our model, which uses GPT-3, generates schemas that are comparable to those generated by manually searching the web for documents, however, when we use inferior LMs such as GPT-2, we see a decline in performance (see Tab. 3 and Tab. 4).

Our assumption is that the quality of the generated schema depends on the quality of the LM and the level of coverage of the selected topic in the LM training data. If, for instance, we were to ask our model to generate a schema for a unique topic such

as "conducting an archaeological dig in an unexplored territory" we doubt that the results would be as useful to an archaeologist as if they were looking for information themselves due to the low coverage of this topic in the corpus the model was trained on. Despite our model's reliance on pre-trained LMs, we believe the generated schemas can always serve as a good basis for further development.

10 Ethical Consideration

The proposed schema induction method does not present any direct societal implications. As is observed in Abid et al. (2021), the text generated by GPT-3 might include undesired social bias. Extracting events and relations from text with such social bias might potentially propagate the bias to the induced schemas. Besides, there are risks of malicious or unintended harmful uses of the generated schemas, for instance, the system might be used to inquire about making a bomb or contriving a terrorist attacks. Yet we believe that the proposed method can benefit various downstream NLP/NLU tasks like event prediction, task-oriented dialogue agents (Andreas et al., 2020) and risk detection (Pohl et al., 2012).

References

Abubakar Abid, Maheen Farooqi, and James Zou. 2021. Large language models associate muslims with violence. *Nature Machine Intelligence*, 3(6):461–463.

Jacob Andreas, John Bufe, David Burkett, Charles Chen, Josh Clausman, Jean Crawford, Kate Crim, Jordan DeLoach, Leah Dorner, Jason Eisner, Hao Fang, Alan Guo, David Hall, Kristin Hayes, Kellie Hill, Diana Ho, Wendy Iwaszuk, Smriti Jha, Dan Klein, Jayant Krishnamurthy, Theo Lanman, Percy Liang, Christopher H. Lin, Ilya Lintsbakh, Andy McGovern, Aleksandr Nisnevich, Adam Pauls, Dmitrij Petters, Brent Read, Dan Roth, Subhro Roy, Jesse Rusak, Beth Short, Div Slomin, Ben Snyder, Stephon Striplin, Yu Su, Zachary Tellman, Sam Thomson, Andrei Vorobev, Izabela Witoszko, Jason Wolfe, Abby Wray, Yuchen Zhang, and Alexander Zotov. 2020. Task-oriented dialogue as dataflow synthesis. *Transactions of the Association for Computational Linguistics*, 8:556–571.

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.

Nathanael Chambers. 2013. Event schema induction with a probabilistic entity-driven model. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807, Seattle, Washington, USA. Association for Computational Linguistics.

Nathanael Chambers and Dan Jurafsky. 2008. Unsupervised learning of narrative event chains. In *Proceedings of ACL-08: HLT*, pages 789–797, Columbus, Ohio. Association for Computational Linguistics.

Nathanael Chambers and Dan Jurafsky. 2009. Unsupervised learning of narrative schemas and their participants. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 602–610, Suntec, Singapore. Association for Computational Linguistics.

Jackie Chi Kit Cheung, Hoifung Poon, and Lucy Vanderwende. 2013. Probabilistic frame induction. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 837–846, Atlanta, Georgia. Association for Computational Linguistics.

Quang Do, Wei Lu, and Dan Roth. 2012. Joint inference for event timeline construction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 677–687, Jeju Island, Korea. Association for Computational Linguistics.

Edward A Feigenbaum. 2003. Some challenges and grand challenges for computational intelligence. *Journal of the ACM (JACM)*, 50(1):32–40.

Goran Glavaš, Jan Šnajder, Marie-Francine Moens, and Parisa Kordjamshidi. 2014. HiEve: A corpus for extracting event hierarchies from news stories. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 3678–3683, Reykjavik, Iceland. European Language Resources Association (ELRA).

Mark Granroth-Wilding and Stephen Clark. 2016. What happens next? event prediction using a compositional neural network model. *Proceedings of the AAAI Conference on Artificial Intelligence*, 30(1).

Ruohao Guo and Dan Roth. 2021. Constrained labeled data generation for low-resource named entity recognition. In *Findings of the Association for Computa-*

tional Linguistics: ACL-IJCNLP 2021, pages 4519–4533, Online. Association for Computational Linguistics.

Rujun Han, Qiang Ning, and Nanyun Peng. 2019. Joint event and temporal relation extraction with shared representations and structured prediction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 434–444, Hong Kong, China. Association for Computational Linguistics.

Lifu Huang, Heng Ji, Kyunghyun Cho, Ido Dagan, Sebastian Riedel, and Clare Voss. 2018. Zero-shot transfer learning for event extraction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2160–2170, Melbourne, Australia. Association for Computational Linguistics.

Bram Jans, Steven Bethard, Ivan Vulić, and Marie Francine Moens. 2012. Skip n-grams and ranking functions for predicting script events. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 336–344, Avignon, France. Association for Computational Linguistics.

Lane Lawley and Lenhart Schubert. 2022. Mining logical event schemas from pre-trained language models. *arXiv preprint arXiv:2204.05939*.

Heeyoung Lee, Marta Recasens, Angel Chang, Mihai Surdeanu, and Dan Jurafsky. 2012. Joint entity and event coreference resolution across documents. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 489–500, Jeju Island, Korea. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Manling Li, Sha Li, Zhenhailong Wang, Lifu Huang, Kyunghyun Cho, Heng Ji, Jiawei Han, and Clare Voss. 2021. The future is not one-dimensional: Complex event schema induction by graph modeling for event prediction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5203–5215, Online and Punta Cana, Dominican Republic. 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.

Ying Lin, Heng Ji, Fei Huang, and Lingfei Wu. 2020. A joint neural model for information extraction with global features. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7999–8009, Online. Association for Computational Linguistics.

Wei Lu and Dan Roth. 2012. Automatic event extraction with structured preference modeling. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 835–844, Jeju Island, Korea. Association for Computational Linguistics.

Adam Meyers, Ruth Reeves, Catherine Macleod, Rachel Szekely, Veronika Zielinska, Brian Young, and Ralph Grishman. 2004. Annotating noun argument structure for NomBank. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC'04)*, Lisbon, Portugal. European Language Resources Association (ELRA).

Ashutosh Modi. 2016. Event embeddings for semantic script modeling. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 75–83, Berlin, Germany. Association for Computational Linguistics.

Kiem-Hieu Nguyen, Xavier Tannier, Olivier Ferret, and Romaric Besançon. 2015. Generative event schema induction with entity disambiguation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 188–197, Beijing, China. Association for Computational Linguistics.

Qiang Ning, Zhili Feng, Hao Wu, and Dan Roth. 2018. Joint reasoning for temporal and causal relations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2278–2288, Melbourne, Australia. Association for Computational Linguistics.

Qiang Ning, Sanjay Subramanian, and Dan Roth. 2019. An improved neural baseline for temporal relation extraction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6203–6209, Hong Kong, China. Association for Computational Linguistics.

Haoruo Peng and Dan Roth. 2016. Two discourse driven language models for semantics. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 290–300, Berlin, Germany. Association for Computational Linguistics.

Haoruo Peng, Yangqiu Song, and Dan Roth. 2016. Event detection and co-reference with minimal supervision. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 392–402, Austin, Texas. Association for Computational Linguistics.

Karl Pichotta and Raymond Mooney. 2014. Statistical script learning with multi-argument events. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 220–229, Gothenburg, Sweden. Association for Computational Linguistics.

Karl Pichotta and Raymond Mooney. 2016. Learning statistical scripts with lstm recurrent neural networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 30(1).

Daniela Pohl, Abdelhamid Bouchachia, and Hermann Hellwagner. 2012. Automatic sub-event detection in emergency management using social media. In *Proceedings of the 21st international conference on world wide web*, pages 683–686.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. 2012. CoNLL-2012 shared task: Modeling multilingual unrestricted coreference in OntoNotes. In *Joint Conference on EMNLP and CoNLL - Shared Task*, pages 1–40, Jeju Island, Korea. Association for Computational Linguistics.

Vasin Punyakanok, Dan Roth, and Wen-tau Yih. 2008. The importance of syntactic parsing and inference in semantic role labeling. *Computational Linguistics*, 34(2):257–287.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

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. *Journal of Machine Learning Research*, 21(140):1–67.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Rachel Rudinger, Pushpendre Rastogi, Francis Ferraro, and Benjamin Van Durme. 2015. Script induction as

language modeling. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1681–1686, Lisbon, Portugal. Association for Computational Linguistics.

Lei Sha, Sujian Li, Baobao Chang, and Zhifang Sui. 2016. Joint learning templates and slots for event schema induction. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 428–434, San Diego, California. Association for Computational Linguistics.

Stephanie Strassel and Jennifer Tracey. 2016. LORELEI language packs: Data, tools, and resources for technology development in low resource languages. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 3273–3280, Portorož, Slovenia. European Language Resources Association (ELRA).

Erik F. Tjong Kim Sang. 2002. Introduction to the CoNLL-2002 shared task: Language-independent named entity recognition. In *COLING-02: The 6th Conference on Natural Language Learning 2002 (CoNLL-2002)*.

Hieu Man Duc Trong, Nghia Ngo Trung, Linh Van Ngo, and Thien Huu Nguyen. 2022. Selecting optimal context sentences for event-event relation extraction. *Association for the Advancement of Artificial Intelligence*.

David Wadden, Ulme Wennberg, Yi Luan, and Hannaneh Hajishirzi. 2019. Entity, relation, and event extraction with contextualized span representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5784–5789, Hong Kong, China. Association for Computational Linguistics.

Haoyu Wang, Muhao Chen, Hongming Zhang, and Dan Roth. 2020. Joint constrained learning for event-event relation extraction. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 696–706, Online. Association for Computational Linguistics.

Haoyu Wang, Hongming Zhang, Muhao Chen, and Dan Roth. 2021. Learning constraints and descriptive segmentation for subevent detection. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5216–5226, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Noah Weber, Niranjan Balasubramanian, and Nathanael Chambers. 2018. Event representations with tensor-based compositions. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1).

Noah Weber, Rachel Rudinger, and Benjamin Van Durme. 2020. Causal inference of script knowledge. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7583–7596, Online. Association for Computational Linguistics.

Haoyang Wen, Ying Lin, Tuan Lai, Xiaoman Pan, Sha Li, Xudong Lin, Ben Zhou, Manling Li, Haoyu Wang, Hongming Zhang, Xiaodong Yu, Alexander Dong, Zhenhailong Wang, Yi Fung, Piyush Mishra, Qing Lyu, Dídac Surís, Brian Chen, Susan Windisch Brown, Martha Palmer, Chris Callison-Burch, Carl Vondrick, Jiawei Han, Dan Roth, Shih-Fu Chang, and Heng Ji. 2021. RESIN: A dockerized schema-guided cross-document cross-lingual cross-media information extraction and event tracking system. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Demonstrations*, pages 133–143, Online. Association for Computational Linguistics.

Xiaodong Yu, Wenpeng Yin, and Dan Roth. 2022. Pairwise representation learning for event coreference. In *Proceedings of the 11th Joint Conference on Lexical and Computational Semantics*, pages 69–78, Seattle, Washington. Association for Computational Linguistics.

Quan Yuan, Xiang Ren, Wenqi He, Chao Zhang, Xinhe Geng, Lifu Huang, Heng Ji, Chin-Yew Lin, and Jiawei Han. 2018. Open-schema event profiling for massive news corpora. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pages 587–596.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. 2020. Big bird: Transformers for longer sequences. *Advances in Neural Information Processing Systems*, 33.

Hongming Zhang, Muhao Chen, Haoyu Wang, Yangqiu Song, and Dan Roth. 2020a. Analogous process structure induction for sub-event sequence prediction. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1541–1550, Online. Association for Computational Linguistics.

Hongming Zhang, Haoyu Wang, and Dan Roth. 2021a. Zero-shot Label-aware Event Trigger and Argument Classification. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1331–1340, Online. Association for Computational Linguistics.

Li Zhang, Qing Lyu, and Chris Callison-Burch. 2020b. Reasoning about goals, steps, and temporal ordering with WikiHow. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4630–4639, Online. Association for Computational Linguistics.

Yi Zhang, Sujay Kumar Jauhar, Julia Kiseleva, Ryen White, and Dan Roth. 2021b. Learning to decompose and organize complex tasks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2726–2735, Online. Association for Computational Linguistics.

Ben Zhou, Qiang Ning, Daniel Khashabi, and Dan Roth. 2020. Temporal common sense acquisition with minimal supervision. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7579–7589, Online. Association for Computational Linguistics.

Ben Zhou, Kyle Richardson, Qiang Ning, Tushar Khot, Ashish Sabharwal, and Dan Roth. 2021. Temporal reasoning on implicit events from distant supervision. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1361–1371, Online. Association for Computational Linguistics.

A Feigenbaum Test Details

The experiment took place online through filling a Google Form and involved 11 volunteer annotators. Each annotator got 3-4 scenarios to annotate. The instructions for the survey appear in Figure 5. An example scenario and the questions of the survey are presented in Fig. 6, Fig. 7, Fig. 8, and Fig. 9.

B Feigenbaum Test Results

In this section we present all the results from the experiments on the dataset Schema-11. Tab. 6 shows the distribution of answers for the question "which schema is more complete?" (same as depicted in Tab. 5), Tab. 7 presents the distribution of answers for the question "which schema was generated by a human?" together with the correct answer written in the bottom row, and Tab. 8 presents the percentage of invalid events and relations determined by the majority vote of the annotators in the automatic schema and the human schema.

C Details on Human Schema Curation

Here are the instructions that were given to the annotators that generated the human schemas for the Schema-11 dataset. All the annotators are graduate students that previously were involved in research projects that include schema induction, SRL, NER or other relevant tasks:

We are developing a system that generates schemas automatically given a topic. We want to compare our automatically-generated schema to schemas derived by people using their common-sense (without relying on texts). To do this, we need expert human annotators and would appreciate your assistance.

A schema is defined as a list of events with their argument types, and the relationships between the events. For example, here is a schema I wrote that describes the event of "armed robbery":

List of events and arguments:

- intend: arg0 - perpetrator [PER], arg1 - commit a felony

- acquire: arg0 - perpetrator [PER], arg1 - weapon [WEA]

- arrive: arg0 - perpetrator [PER], arg-loc - crime scene [LOC]

- assault: arg0 - perpetrator [PER], arg1 - [PER]

- threaten: arg0 - perpetrator [PER], arg1 - [PER]

- get: arg0 - perpetrator [PER], arg1 - money or goods

- injure: arg0 - perpetrator [PER], arg1 - [PER]

- kill: arg0 - perpetrator [PER], arg1 - [PER]

- flee: arg0 - perpetrator [PER], arg-loc - crime scene [LOC]

- call: arg0 - [PER], arg1 - police [ORG]

- chase: arg0 - police [ORG], arg1 - perpertrator[PER]

- catch: arg0 - police [ORG], arg1 - perpertrator[PER]

- manage to escape: arg0 - perpertrator[PER]

Temporal and logical relations (in the form of a timeline):

- a perpetrator (PER) **intent** to commit a felony ->

- the perpetrator (PER) **acquires** weapon (WEA) ->

- the perpetrator (PER) **arrives** at the scene (LOC) ->

- perpetrator (PER) **assault** victim (PER) with weapon (WEA) at the scene (LOC) **OR** perpetrator (PER) **threatens** a person (PER) with the weapon (WEA) at the scene (LOC) ->

- perpetrator (PER) **gets** money or goods from the person (PER) **OR** victim **injured OR** victim **killed** ->

- perpetrator **flees** the scene of the crime (LOC) **AND** someone (PER) **calls** the police (ORG) ->

- the police (ORG) are **chasing** the criminal (PER) ->

- the police (ORG) **catches** the perpetrator (PER) **XOR** the criminal (PER) **manages to escape**.

The complex events we are interested in are the following: (1) Disease Outbreak (2) IED Bombing (3) Civil Unrest (4) International Invasion (5) Disaster and Rescue (6) Terrorism Attacks (7) Election (8) Kidnapping (9) Business Change (10) Mass Shooting.

Feigenbaum Test – Scenario 11

This form mainly focuses on the evaluation of machine generated schema. Given a certain scenario, the schema includes stereotypical events and the relations between them, for instance, within scenario "acquiring a PhD degree", a schema would typically includes "publish papers," "attend conferences," "write PhD thesis" and "defend PhD thesis." And there is also a "before" relation between "write PhD thesis" and "defend PhD thesis." Besides, we also have "SuperSub" relation that means hierarchical relation between events, and "AND"/"OR" relation that means the two events must happen together/either of the events may happen.

We've asked a group of people to generate schemas from their commonsense knowledge. Given two schemas per scenario, your task is to determine whether you can distinguish the machine generated schema from the human generated one. And also provide your insights on the completeness and soundness of each schema.

For completeness, we would like you to tell us which schema is more complete.
For soundness, we would like you to tell us for each event and relation listed, whether it is valid for this scenario.
Most importantly, we would like to know which schema you think is generated by human.

Figure 5: Instructions for the Feigenbaum test.

	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11
Human	4	0	1	1	1	2	1	1	0	3	1
Automatic	2	3	4	2	1	1	1	1	4	0	1

Table 6: Completeness results. The table presents the number of votes that were recorded for which schema is more complete - the human generated schema or the automatically generated schema. The majority vote is highlighted in yellow.

	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11
A	1	1	3	0	0	2	0	2	2	1	1
B	5	2	2	3	2	1	2	0	2	2	1
Correct Answer	B	B	B	B	B	A	B	A	A	B	B

Table 7: Feigenbaum test results. The annotators guesses which schema (A or B) was generated by humans. The number of votes for each option appear along with the correct answer in the bottom row. The correct majority guesses are marked with green and incorrect with red.

	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11
Invalid Events (Auto.)	0	0	0	0	0	8.33	0	7.69	0	14.28	0
Invalid Relations (Auto.)	46.15	16.66	25	25	0	23.52	0.4	11.76	12.5	22.22	46.15
Invalid Events (Human)	0	0	14.28	14.28	0	0	0	0	0	0	0
Invalid Relations (Human)	7.69	50	15.38	15.38	0	6.25	0	11.11	0	10	7.69

Table 8: Invalidity results. The table presents the percentage of invalid events and relations determined by the human annotators for each schema and scenario.

Scenario 11: Terrorism Attack (A)

Events:
1. event: kill, arg0: {PER, ORG, VEH, WEA}, arg1: PER
2. event: injure, arg0: {PER, ORG, VEH, WEA}, arg1: PER
3. event: detonate, arg0: PER, arg1: WEA
4. event: come, arg1: attack
5. event: open, arg0: {PER, ORG}, arg1: fire
6. event: wound, arg0: {PER, ORG, VEH, WEA}, arg1: PER
7. event: strike, arg0: {PER, ORG, WEA}
8. event: claim, arg0: ORG, arg1: responsibility
9. event: leave, arg0: {PER, VEH}
10. event: attack, arg0: {PER, ORG}
11. event: choose, arg0: {PER, ORG}, arg1: {PER, ORG, GPE}
12. event: select, arg0: {PER, ORG}, arg1: method
13. event: acquire, arg0: {PER, ORG}, arg1: WEA
14. event: carry out, arg0: PER

Relations:
1. before: 3->8
2. before: 3->5
3. before: 1->4
4. before: 1->9
5. before: 2->4
6. before: 2->9
7. before: 6->4
8. before: 6->9
9. before: 11->12->13->14->10
10. OR: 8,5
11. OR: 4,9
12. AND: 1,2,6
13. supersub: 10->7->1,2

Figure 6: An example schema in the topic of Terrorism Attack. This schema was generated automatically (information that was unknown to the annotators).

Scenario 11: Terrorism Attack (B)

Events:
1. event: find, arg0: PER, arg1: ORG, arg-loc:LOC
2. event: emerge, arg0: ORG, arg1: ORG
3. event: fade, arg0: ORG, arg-tmp: TMP
4. event: reemerge, arg0: ORG, arg-tmp: TMP, arg-loc: LOC
5. event: lead, arg0: ORG, arg1: losses
6. event: lost, arg0: ORG, arg1: LOC
7. event: declare, arg0: GPE, arg1: ORG
8. event: kill, arg0: GPE, arg1: PER
9. event: plan, arg0: PER
10. event: executes, arg0: PER
11. event: injures, arg0: the attack, arg1: PER
12. event: kills, arg0: the attack, arg1: PER
13. event: damages, arg0: the attack, arg1: infrastructure
14. event: calls, arg0: PER, arg1: PER
15. event: arrive, arg0: PER
16. event: treat, arg0: PER, arg1: PER
17. event: take, arg0: PER, arg1: PER
18. event: reports, arg0: PER
19. event: claims, arg0: the group, arg1: responsibility

Relations:
1. before: 9->10
2. before: 10->11
3. before: 10->12
4. before: 10->13
5. before: 10->14
6. before: 14->15->16->17
7. before: 10->18
8. before: 10->19
9. before: 1->3
10. before: 3->4
11. AND: 1->2
12. cause: 5->6
13. cause: 8->7

Figure 7: An example schema in the topic of Terrorism Attack. This schema was generated by a human (information that was unknown to the annotators).

Which schema is more complete? *

○ (A)

○ (B)

Which one do you think is generated by human? *

○ (A)

○ (B)

Figure 8: Questions that were asked about the completeness of the schemas and the generator of the schema.

For each EVENT in schema (B), select if it is valid to appear in the scenario. *

	valid	invalid
Row 1	○	○
Row 2	○	○

Figure 9: Questions about the validity of the events appearing in one of the schemas. This question was asked on both schemas and on the relations appearing in the schemas too.

D Wizard of Oz Experiment

Two examples of the Wizard of Oz experiment for the events of invasion and remodel a facility. The examples show the three generated texts using the three prompting methods, the schema created by the author, and the LDC schema together with a matching between the events in the two schemas. The entire list of sampled topics is: print a 3D object, cyber attack, make a website, religious conversion, invasion, shipping and mailing, remodel a facility, obtain and provide medical treatment, build a drone, and nonviolent protest march.

D.1 Example 1: Invasion

Prompts and Outputs:

1. Prompt: describe how to perform an invasion.
   Output: 1. Choose a target: Identify a target population that you want to invade. This target population can be a group of people, a city, or a country. 2. Prepare your troops: Assemble your troops and supplies in preparation for the invasion. 3. Launch your attack: Attack the target population using whatever means necessary to take control. 4. Consolidate your power: Once you have taken control of the target population, consolidate your power by establishing your authority and eliminating any resistance.

2. Prompt: what are the steps involved in an invasion?
   Output: 1. The first step in an invasion is to gather intelligence about the enemy. This includes learning about their strengths, weaknesses, and their territory. 2. The next step is to plan the invasion. This includes deciding what resources will be needed and how they will be used. 3. The third step is to launch the invasion. This is when the actual fighting begins. 4. The fourth step is to consolidate the gains. This is when the invading force secures their position and prepares for the next phase of the operation.

3. Prompt: write a news headline about invasion.
   Output: U.S. troops invade Canada in surprise attack.
   Prompt: write a news article titled: "U.S. troops invade Canada in surprise attack."
   Output: In a stunning move, the United States has invaded Canada in a surprise attack. The

troops came across the border early this morning, catching the Canadians by surprise. The Canadians are currently fighting back, but it is unknown how long they will be able to hold out against the might of the United States military. This is a developing story, and more details will be released as they become available.

Schema generated based on the three outputs

1. Choose target.

2. Gather intelligence on target: learn strengths and weaknesses and territory.

3. Plan invasion: decide on resources and how the resources will be used.

4. Prepare troops: Assemble troops and supplies.

5. Launch attack: come across the border.

6. Fight or use whatever means possible.

7. Take control.

8. Consolidate power: establish authority and eliminate resistance.

9. Prepare for the next phase of operation.

LDC Schema

1. Step 1: PlanInvasion: Plan[3] for the invasion.

   (a) Step 1.1: IdentifyTargets[1]: Leadership builds a list of entry points for the invasion and selects one or more as the target(s) of the initial invasion.
   (b) Step 1.2: DevelopPlan[3]: A team develop a master invasion plan.
   (c) Step 1.3: OrderDissemination [not matched]: Planners send out detailed instructions to military forces about the invasion plan.

2. Step 2: DeployTroops[4]: Amass and deploy troops for invasion.

3. Step 3: Invasion[5]: Military force launch the invasion and enter the target territory.

4. Step 4: DisperseTroops[6]: Dispersion of invading troops across territory.

Results: The GPT3 schema covers 6 out of 7 events that are mentioned in the LDC schema. The matching between LDC events and schema events are marked in [] next to the event on the LDC schema. GPT3 schema has 7 additional events: gather intelligence, learn strengths, decide on resources, assemble troops and supplies, come across border, take control, consolidate power, establish authority, eliminate resistance, prepare for next phase.

D.2 Example 2: Remodel A Facility

Prompts and Outputs:

1. Prompt: describe how to remodel a facility.
   Output: There is no one-size-fits-all answer to this question, as the best way to remodel a facility depends on the specific needs and goals of the organization. However, some tips on how to remodel a facility include: 1. Define the scope of the project. 2. Create a budget and timeline. 3. Hire an experienced architect or designer. 4. Work with a reputable contractor. 5. Use high-quality materials. 6. Follow building codes and regulations. 7. Get input from stakeholders. 8. Test the new space before moving in.

2. Prompt: what are the steps involved in remodeling a facility?
   Output: 1. The first step in remodeling a facility is to develop a plan. This plan should include the goals of the remodel, the budget, the timeline, and the scope of work. 2. The next step is to obtain the necessary permits. Depending on the scope of work, this may require a building permit, a zoning permit, and/or a permit from the local health department. 3. Once the permits are obtained, the next step is to select a contractor. The contractor should be licensed and insured, and should have experience with the type of work that needs to be done. 4. The contractor will then begin the work, which may include demolition, construction, painting, and/or installation of new fixtures and appliances. 5. Once the work is completed, the facility will need to be inspected to ensure that it meets all applicable codes and regulations. 6. Finally, the facility can be reopened to the public.

3. Prompt: write a news headline about remodeling a facility.

Output: Facility Remodeling Underway
Prompt: write a news article titled: "Facility Remodeling Underway"
Output: In an effort to better serve the needs of the community, the local library is undergoing a remodel. The project is expected to take several months to complete, but the end result will be a more user-friendly space that meets the needs of today's patrons. Some of the changes include a new layout for the adult fiction section, additional study rooms, and updated furniture throughout the facility. The children's area will also be expanded to create a more inviting and interactive space. The library staff is excited about the changes and can't wait to see the finished product. They are confident that the updates will make the library a more popular destination for both residents and visitors.

Schema generated based on the three outputs

1. Develop a plan: define scope and goals and create budget and timeline.

2. Hire an architect or designer and obtain necessary permits.

3. Select a contractor (preconditions: contractor licensed, insured, have experience).

4. Contractor works: use high quality materials AND follow building code and regulations and demolition or construction or painting or installation.

5. Get input from stakeholders.

6. Inspect facility.

7. Test space.

8. Reopen facility.

9. Facility is user-friendly and meets needs of patrons.

LDC Schema

1. Step 1: Acquisition [not matched]: Acquire facility.

2. Step 2: Planning[1]: Research and plan expected usage, budget, changes, legal issues, dependencies.

3. Step 3: Labor[2,3]: Arrange for skills, or laborers, or both.

   (a) Step 3.1: AcquireSkills [not matched]: Acquire skills or knowledge required for remodeling.

   (b) Step 3.2: HireLaborers[2,3]: Hire skilled person or organization to perform remodeling work.

4. Step 4: AcquireMaterials[4.1]: Acquire materials and tools.

5. Step 5: Remodel[4]: Facility is remodeled.

   (a) Step 5.1: Demolition[4.2]: Deconstruction or demolition of portions of building and/or equipment installations.

   (b) Step 5.2: DebrisRemoval [not matched]: Hauling away/dumping of debris.

   (c) Step 5.3: Modification[4.2]: Modification, addition, or installation of building or systems/equipment in building.

6. Step 6: Inspection[6,7]: Inspect and/or test new portions of facility and/or new systems of facility for functionality and compliance with laws and regulations.

Results: The GPT3 schema covers 8 out of 11 events that are mentioned in the LDC schema. The matching between LDC events and schema events are marked in [] next to the event on the LDC schema. GPT3 schema has 9 additional events: contractor works, follow building code and regulations, preconditions on contractor, painting, installation, construction.