Harvesting Event Schemas from Large Language Models

Jialong Tang[1,3], **Hongyu Lin**[1], **Zhuoqun Li**[1,3], **Yaojie Lu**[1],
Xianpei Han[1,2], **Le Sun**[1,2,*]
[1]Chinese Information Processing Laboratory [2]State Key Laboratory of Computer Science
Institute of Software, Chinese Academy of Sciences, Beijing, China
[3]University of Chinese Academy of Sciences, Beijing, China
{jialong2019,xianpei,sunle}@iscas.ac.cn

Abstract

Event schema provides a conceptual, structural and formal language to represent events and model the world event knowledge. Unfortunately, it is challenging to automatically induce high-quality and high-coverage event schemas due to the open nature of real-world events, the diversity of event expressions, and the sparsity of event knowledge. In this paper, we propose a new paradigm for event schema induction – knowledge harvesting from large-scale pre-trained language models, which can effectively resolve the above challenges by discovering, conceptualizing and structuralizing event schemas from PLMs. And an **E**vent **S**chema **H**arvester (**ESHer**) is designed to automatically induce high-quality event schemas via in-context generation-based conceptualization, confidence-aware schema structuralization and graph-based schema aggregation. Empirical results show that ESHer can induce high-quality and high-coverage event schemas on varying domains.

1 Introduction

Event is one of the basic units for human beings to understand and experience the world (Jackendoff, 1992). An event is a specific occurrence involving multiple participants, such as *bombing*, *election*, and *marriage*. To represent events and model the world event knowledge, event schema provides a conceptual, structural and formal language which can describe the types of events, the semantic roles (slots) of specific events, and the relations between different events. Specifically, an event schema is a frame such as "***Type***: *bombing*, ***Slots***: *perpetrator, victm, target, instrument*" (Chambers and Jurafsky, 2011), which is central in event extraction (Lu et al., 2021, 2022), event relationship understanding (Irwin et al., 2011; Li et al., 2020a, 2021), and event knowledge base construction (Zhang et al., 2020; He et al., 2022). Due to its importance, it is critical to automatically discover and construct large-scale, high-quality, and high-coverage event schemas, i.e., event schema induction.

[*]Corresponding authors.
Our source codes with corresponding experimental datasets will be openly available at `https://github.com/TangJiaLong/Event-Schema-Harvester`.

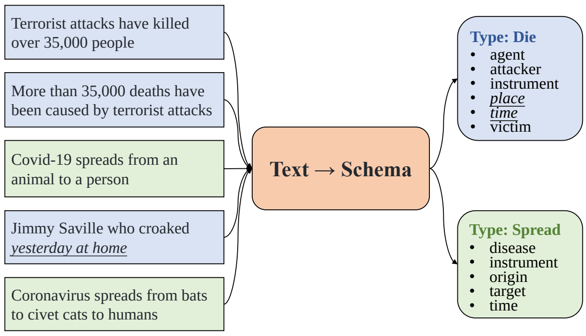

Figure 1: The event harvest paradigm for event schema induction, which induces high-quality event schemas from diverse event expressions and dispersed event knowledge on open domains, e.g., the top two texts on the left use different utterances to describe the same event while the fourth text complements the slots "*place*" and "*time*".

Event schema induction, unfortunately, is a non-trivial task due to the open nature of real-world events, the diversity of event expressions, and the sparsity of event knowledge. Firstly, in real-world applications, the size of event types is very large and new types of events are constantly emerging. To address this open problem, event schemas must be induced automatically and with a high coverage on varying domains. Secondly, as shown in Figure 1, events are usually expressed using very different natural language utterances, therefore it is critical to normalize diverse event expressions by conceptualizing and structuralizing them into formal event schemas. Finally, due to the economic principle of language (De Saussure, 2011), event expressions are mostly incomplete and many event arguments are missing. To resolve this sparsity problem, an event schema induction method must aggregate dispersed event knowledge across different expressions.

Up to recently, all most event schemas are still hand-engineered by human experts, which are expensive and labour-intensive (e.g., schemas in MUC (Chinchor et al., 1993), ACE (Doddington et al., 2004) and TAC-KBP (Ji and Grishman, 2011)). On the other hand, traditional automatic event schema induction methods still cannot overcome the open, diversity, and sparsity challenges. For instance, bottom-up concept linking methods (Huang et al., 2016; He et al., 2022) discover event types/slots by parsing and linking event expressions to external schema resources such as FrameNet (Baker et al., 1998), which are limited by the quality and the coverage of external schema resources. Top-down clustering methods (Chambers, 2013; Cheung et al., 2013; Nguyen et al., 2015; Sha et al., 2016; Ahn, 2017; Yuan et al., 2018; Liu et al., 2019b; Shen et al., 2021) cluster event expressions according to pre-defined schema templates (e.g., the 5W1H template, or templates with the predefined number of event types/slots), which are highly constrained by the pre-defined templates. To sum up, it remains a critical challenge to automatically discover schemas on open domains, normalise event schemas from diverse expressions, and aggregate dispersed knowledge from sparse descriptions.

In this paper, we propose a new paradigm for event schema induction – knowledge harvesting from large pre-trained language models (PLMs), which can effectively address the open, diversity, and sparsity challenges. The main idea is to automatically harvest open-domain and high-coverage event schemas from large PLMs and leverage the strong text generation and in-context learning abilities of PLMs for discovering, conceptualizing, and structuralizing event schemas.

Specifically, we design an **E**vent **S**chema **H**arvest**er** (**ESHer**), which automatically discovers and normalizes event types and their semantic roles via the following three components: 1) *text conceptualization via in-context generation*, which can unsupervised-ly transform diverse event expressions into conceptualized event schema candidates based on in-context demonstrations; 2) *confidence-aware schema structuralization*, which structuralizes event schemas by selecting and associating event types with their salient, reliable and consistent slots; 3) *graph-based schema aggregation*, which aggregates dispersed event schemas via graph-based clustering. In this way, the open, diversity, and sparsity challenges can be effectively resolved via schema conceptualization, structuralization and aggregation.

We conducted experiments on ERE-EN (Lin et al., 2020) and additional datasets in multiple domains including finance (ChFinAnn (Zheng et al., 2019)), pandemic (Cov-19 (Shen et al., 2021)), and daily news (New York Time and People's Daily). Empirical results show that ESHer surpasses the traditional methods in discovering high-quality and high-coverage event schemas. And the induced event schemas are close to human-annotated ones and can be quickly extended to varying domains and emerging event types.

In general, this paper's main contributions are:

- We propose a new event schema induction paradigm – knowledge harvesting from large-scale PLMs, which can effectively resolve the open, diversity, and sparsity challenges.
- We design ESHer, which can automatically induce event schemas via in-context generation-based text conceptualization, confidence-aware schema structuralization, and graph-based schema aggregation.
- Experiments show ESHer can induce high-quality and high-coverage event schemas on varying domains. And we believe the induced event schemas are valuable resources which can benefit many downstream NLP tasks.

2 Event Schema Harvester

This section describes how to discover, conceptualize, and structuralize event schemas from large pre-trained language models (PLMs) so that the open, diversity and sparsity challenges can be effectively resolved by automatically harvesting open-domain and high-coverage event schemas from large PLMs and leveraging the strong text generation and in-context learning abilities of large PLMs.

Formally, given an unlabeled corpus $\mathcal{C}$ and a PLM, our event schema induction method discovers event clusters $\mathcal{Y} = \{y_1, y_2, ..., y_N\}$, where N is the number of discovered event types. For each event cluster y, we automatically conceptualize it to a name t as well as its corresponding semantic roles $\{s_1^t, s_2^t, ...\}$, where $t \in \mathcal{T}$, $s \in \mathcal{S}$ and $\mathcal{T}/\mathcal{S}$ are open domain event type/slot names.

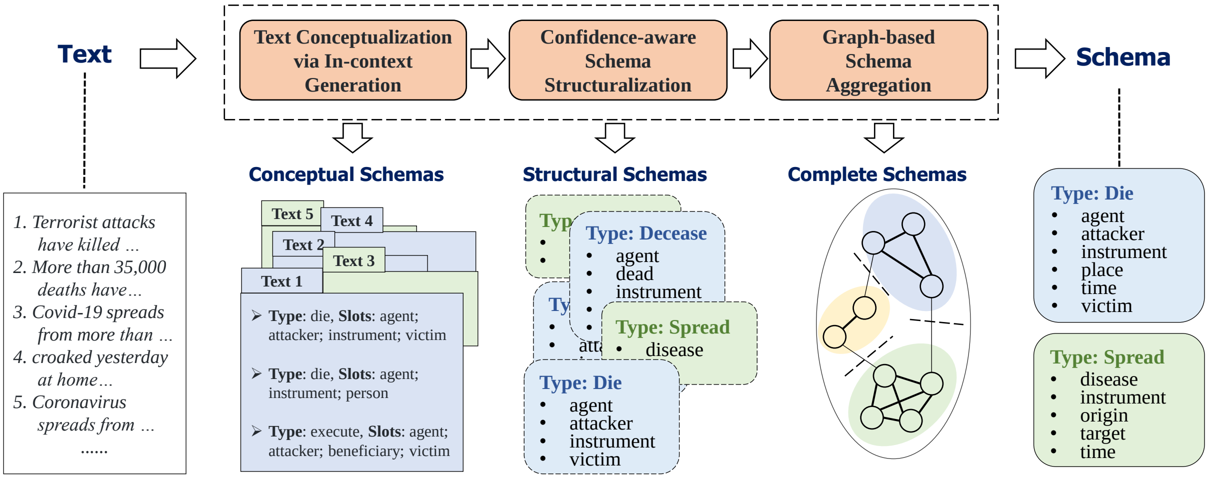

Figure 2: An overview of ESHer, which automatically discovers and normalizes event types and their semantic roles via (1) *text conceptualization via in-context generation*, (2) *confidence-aware schema structuralization* and (3) *graph-based schema aggregation*.

To this end, we design **E**vent **S**chema **H**arvest**er** (**ESHer**), and its framework is shown in Figure 2. ESHer contains three components: 1) *text conceptualization via in-context generation*, which transforms diverse event expressions into conceptualized event schemas based on in-context demonstrations; 2) *confidence-aware schema structuralization*, which structuralizes event schemas by selecting and associating event types with their salient, reliable and consistent slots; 3) *graph-based schema aggregation*, which aggregates dispersed sparse event knowledge across individual event schemas via graph-based clustering. In follows we describe these components in detail.

2.1 Text Conceptualization via In-context Generation

Events are usually expressed in diverse natural language utterances, which poses a critical challenge for schema induction. For example, "*Terrorist attacks have killed over 35,000 people*" and "*More than 35,000 deaths have been caused by terrorist attacks*" convey the same event, but with quite different words and syntactic structures. To address this challenge, we conceptualize diverse utterances into schema candidates, which can distil event schema knowledge and uniformly represent them. For example, our method will distil and represent the event types and the semantic roles in the above two examples as the same schema "***Type**: die, **Slots**: agent; attacker; instrument; victim*" (as shown in Figure 2).

To this end, this section proposes an unsupervised text-to-schema framework by leveraging the strong in-context generation ability of PLMs. Specifically, we model text conceptualization as an in-context generation process:

$$\rightarrow Schema$$

where: "*Demonstrations*" is a list of examples used to instruct PLMs how to conceptualize text to schema, and each demonstration is a <text, schema> pair represented as "*text → schema*", here "*Text*" is the event utterance we want to conceptualize, "*Schema*" is the conceptualized schema represented as "***Type**: t, **Slots**: s_1^t; s_2^t ...*", and "→" is a special token that separates the text and the event schema.

We can see that, our method is unsupervised so it can effectively resolve open and emerging events in real-world applications, and it is in-context instructed so it can be generalized to different domains/languages by instructing PLMs with appropriate in-context demonstrations.

There are many ways to select appropriate in-context demonstrations. This paper directly samples them from existing human-annotated event datasets (e.g., ACE (Doddington et al., 2004), DuEE (Li et al., 2020b), etc) and we found text conceptualization benefits from high-quality and diverse demonstrations. We believe this is because diverse demonstrations can help PLMs to better generalize to different domains, event types and event semantic roles. Furthermore, to recall more event knowledge from an instance, we generate n schema candidates $c_1, c_2, ...c_n$ for each text, where n is a hyperparameter.

2.2 Confidence-aware Schema Structuralization

The text-to-schema component distils and conceptualizes event knowledge in diverse expressions. This section describes how to structuralize these conceptualized event schemas by selecting and associating the salient, reliable, and consistent slots for each event type. For instance, we can structuralize a "*die*" event frame by evaluating the association between event type "*die*" and slots "*agent; attacker; instrument; victim*" (as shown in Figure 2).

Formally, as shown in Algorithm 1, we use $\mathcal{O}$ to denote the results of text conceptualization, in which j-th instance is $(text^j, \{c_1^j, c_2^j, ...c_n^j\})$ and $\{c_1^j, c_2^j, ...c_n^j\}$ are n generated schema candidates, and we use $SlotSet^j$ to denote the union of all generated slots of instance j by summarizing slots in $\{c_1^j, c_2^j, ...c_n^j\}$ (**Line 2-6**). To select high-quality slots for event types, we design a set of metrics to estimate the quality of slots and type-slot associations, including **Salience**, **Reliability** and **Consistency** (**Line 8-11**). We describe them as follows:

Salience - a salient slot of an event type *t* should appear frequently in the generated schemas of *t*, but less frequent in other events. For example, in Figure 2, the slots "*attacker*" and "*victim*" are more salient than "*person*" for "*die*" event. Following the TF-IDF idea, the salience of a slot *s* in j-th instance is computed as:

$$Salience(s)^j = (1 + log(freq(s)^j)^2) * log(\frac{|\mathcal{O}|}{\sum_k^{|\mathcal{O}|} freq(s)^k}) \tag{1}$$

where $freq(s)^j$ is the frequency of the slot s in $SlotSet^j$, $|\mathcal{O}|$ is the total number of instances in the outputs $\mathcal{O}$.

Reliability - a slot is reliable if it co-occurs frequently with other slots in multiple candidates of one instance. For example, in Figure 2, the slot "*agent*" is considered reliable to "*die*" event because it co-occurs with all other slots. We use PageRank algorithm (Page et al., 1999) to compute the slot reliability as follows:

$$Reliability(s)^j = \beta \sum_k^{|SlotSet^j|} \frac{Reliability(s^k)}{d(s^k)} + (1-\beta)\frac{1}{|SlotSet^j|} \tag{2}$$

Algorithm 1 : Confidence-aware Schema Structuralization.

Input: $\mathcal{O}$: the results of text conceptualization, where j-th instance of $\mathcal{O}$ is $(text^j, \{c_1^j, c_2^j, ...c_n^j\})$;
1: **for** j-th instance $\in \mathcal{O}$ **do**
2: $\quad SlotSet^j \leftarrow \emptyset$
3: $\quad$ **for** $c_i^j \in \{c_1^j, c_2^j, ...c_n^j\}$ **do**
4: $\quad\quad c_i^j =$ "***Type*** : $\hat{t}^{i,j}$ ***Slots*** : $s_1^{\hat{t}^{i,j}}; s_2^{\hat{t}^{i,j}}; ...$"
5: $\quad\quad SlotSet^j \leftarrow SlotSet^j \cup \{s_1^{\hat{t}^{i,j}}; s_2^{\hat{t}^{i,j}}; ...\}$
6: $\quad$ **end for**
7: $\quad$ **for** $s \in SlotSet^j$ **do**
8: $\quad\quad Salience(s)^j \leftarrow$ Equation (1)
9: $\quad\quad Reliability(s)^j \leftarrow$ Equation (2)
10: $\quad\quad Consistency(s)^j \leftarrow$ Equation (3)
11: $\quad\quad Score(s)^j \leftarrow$ Equation (4)
12: $\quad\quad$ **if** $Score(s)^j$ < threshold **do**
13: $\quad\quad\quad SlotSet^j$.del($s$)
14: $\quad$ **end for**
15: $\quad$ Select top-1 consistent event type $\hat{t}^j$
16: $\quad$ j-th instance $\leftarrow (text^j, \hat{t}^j, SlotSet^j)$
17: **end for**
Return: $\mathcal{O}$;

---

where β is a hyper-parameter, $|SlotSet^j|$ is the number of slots in $SlotSet^j$ and $d(s^k) = \sum_{k(s \leftrightarrow s^k)}^{|SlotSet^j|} Reliability(s^k)$, $s \leftrightarrow s^k$ means that slots s and s^k co-occur in the same candidate. We initialize the reliability score for all slots as $\frac{1}{|SlotSet^j|}$ and run PageRank T iterations or the change is less than ϵ.

Consistency – because PLMs may generate unfaithful schemas which are unfaithful to input event expressions, we also estimate the consistencies of event types and slots. Concretely, we evaluate the consistency between the generated event schemas and event expressions using semantic similarities based on WordNet (Miller, 1992), HowNet (Dong et al., 2010) and BERT (Devlin et al., 2019). And the consistency score of a slot in j-th instance is:

$$Consistency(s)^j = Sim(\hat{t}^j, text^j | s, \hat{t}^j \in c) \tag{3}$$

where $Sim(\cdot)$ is a semantic similarity function, $s, \hat{t}^j \in o$ denotes that slot s is the corresponding semantic role of the predicted event type $\hat{t}^j$ in the same schema candidate c.

The final confidence of a slot is computed by combining the salience, reliability, and consistency scores:

$$Score(s)^j = (\lambda_1 * Salience(s)^j + \lambda_2 * Reliability(s)^j) * Consistency(s)^j \tag{4}$$

where λ_1 and λ_2 are two hyperparameters.

Finally, we only retain the top-1 consistent event type for each instance and filter all slots in that instance if their confidence scores are below a certain threshold (**Line 12-16**). In this way, we obtained structuralized event schemas such as "***Type**: die, **Slots**: agent; attacker; instrument; victim*" (as shown in Figure 2).

2.3 Graph-based Schema Aggregation

As described above, event knowledge is sparse in event expressions due to the economical principle of language (De Saussure, 2011). This section describes how to address the sparsity issue by aggregating dispersed semantic roles across different schemas. For example, we can obtain a more complete schema for the "*die*" event by combining "***Type**: die, **Slots**: agent; attacker; instrument; victim*" with "***Type**: decease, **Slots**: agent; dead, instrument; place; time*".

To this end, this section proposes a graph-based clustering method which first groups individual event schemas into clusters, and then aggregates event types and slots in the same cluster. The main idea here is that event schemas are of the same event type if their original expressions describe the same kind of occurrence (text similarity), their predicted types are synonyms (type similarity) and they share many common semantic slots (slot set similarity). For example, in Figure 2, "*die*" and "*decease*" are synonyms and "*agent*" and "*instrument*" are common semantic roles, therefore they are highly likely the same event type.

Based on the above idea, given instances $\mathcal{O}$ after confidence-aware schema structuralization, in which the j-th instance is represented as $(text^j, \hat{t}^j, SlotSet^j)$, we construct a graph to model the similarities between different individual event schemas. In the graph, each node is an event schema and the similarity between two schema nodes is computed by considering the similarity between their event expressions, the similarity between their event types, and the similarity between their slot sets:

$$\begin{aligned} Graph[i][j] &= Graph[j][i] \\ &= \lambda_3 * Sim(text^i, text^j) \\ &+ \lambda_4 * Sim(\hat{t}^i, \hat{t}^j) \\ &+ \lambda_5 * Sim(SlotSet^i, SlotSet^j) \end{aligned} \tag{5}$$

where λ_3, λ_4, and λ_5 are hyper-parameters, and $Sim(\cdot)$ is the semantic similarity function defined in Equation 3.

Given the schema graph, we employ the Louvain algorithm (Blondel et al., 2008) to segment and group schemas into clusters:

$$\hat{Y} = \{\hat{y}^1, \hat{y}^2, ..., \hat{y}^{|\mathcal{O}|}\} = Louvain(Graph) \tag{6}$$

where $\hat{y}^j \in \mathcal{Y} = \{y_1, y_2, ..., y_N\}$ indicates that the j-th schema is assigned to the $\hat{y}^j$-th event cluster and each cluster representing a distinct event type.

Finally, we aggregate all individual event schemas in the same cluster to obtain a complete schema. Given a cluster y which can be represented as as a tuple (***Types**, **Slots***), with ***Types*** $= \{\hat{t}^1, \hat{t}^2, ...\}$ and ***Slots*** $= \{s^t_1, s^t_2, ...\} = SlotSet^1 \cup SlotSet^2 \cup ...$ are the predicted event types/slots by summarizing event types/$SlotSets$ from all individual schemas. An example of such a cluster in Figure 2 is "*(**Types**: {die, decease}; **Slots**: {agent; attacker; dead; instrument; place; time; victim})*".

The final event type name of this cluster is normalized by selecting the most salient prediction from $\{\hat{t}^1, \hat{t}^2, ...\}$, e.g., "*die*". For event slots, there may be synonymous slot names in ***Slots*** stand for the same semantic role, e.g., {*dead, victim*} is the synonymous set in the above example. Thus, we utilize the Louvain Algorithm (Blondel et al., 2008) again to identify synonymous event slots and then select the most salient slot to represent its synonyms, e.g., "*victim*" is chosen as the representative slot name of the synonymous set {*dead, victim*}[2]. The final slot names of this cluster are normalized by these selected slots, e.g., the aggregated complete event schema of the above example is "***Type**: die, **Slots**: agent; attacker; instrument; place; time; victim*" (as shown in Figure 2).

Summary By conceptualizing diverse event expressions, structuralizing schemas by selecting and associating event types and their slots, and aggregating dispersed event knowledge across different schemas, our knowledge harvesting method can effectively address the open, diversity, and sparsity challenges, and induce conceptual, structural, and formal event schemas from PLMs.

[2]In this step, we build a graph to model the connectivities between each slot which is similar to Equation 5. In the graph, each node is a slot and the similarity between two slot nodes is defined as $Sim(s^i, s^j)$.

ERE-EN

ESHer	Experts
Type: Transfer-Money	**Type: Transfer-Money**
giver	**giver**
recipient	**recipient**
place	**place**
currency	beneficiary
date	

ESHer	Experts
Type: Accusation	**Type: Indict**
defendant	**defendant**
prosecutor	**prosecutor**
place	**place**
	adjudicator

ESHer	Experts
Type: Convict	**Type: Convict**
defendant	**defendant**
place	**place**
adjudicator	adjudicator
sanction	
person	

ChFinAnn

ESHer	Experts
Type: 质押 (Equity Pledge)	**Type: Equity Pledge**
出质方 (pledger)	**pledger**
质押方 (pledgee)	**pledgee**
出质股数 (pledgedShares)	pledgedShares
持股数 (totalHoldingShares)	totalHoldingShares
持股比例 (totalHoldingRatio)	totalHoldingRatio
总质押股数 (totalPledgedShares)	totalPledgedShares
时间 (date)	startDate; endDate; releasedDate

ESHer	Experts
Type: 减持 (Equity Underweight)	**Type: Equity Underweight**
减持方 (equityHolder)	**equityHolder**
减持数量 (tradedShares)	**tradedShares**
持股调整后持股比例 (laterHoldingShares)	laterHoldingShares
价格 (averagePrice)	averagePrice
时间 (date)	startDate; endDate

Figure 3: Schemas induced by ESHer and annotated by experts, in which **bold black** denotes the directly matched event types/slots; black denotes recalled ground truths; teal denotes the unmatched but reasonable ones; orange denotes the missing references; red denotes the wrong predictions.

Model	ERE-EN: # of Event Types: Human	Discover	Overlap	Acceptable	# of Event Slots: Human	Discover	Overlap	Acceptable	Recall
ESHer	38	71	21.05%	85.92%	115	198	11.30%	44.95%	35.21%
ESHer (upper bound)	38	100	21.05%	93.00%	115	371	19.13%	59.30%	49.00%
Model	**ChFinAnn: # of Event Types:** Human	Discover	Overlap	Acceptable	**# of Event Slots:** Human	Discover	Overlap	Acceptable	Recall
ESHer	5	44	100.00%	72.73%	35	231	37.14%	59.31%	15.91%
ESHer (upper bound)	5	100	100.00%	96.00%	35	458	71.43%	85.81%	22.00%

Table 1: Schema Coverage Comparison on ERE-EN and ChFinAnn.

3 Experiments

3.1 Experimental Settings

Datasets. We use ERE-EN (Song et al., 2015) as our primary dataset because its event schemas are manually annotated. Furthermore, to assess event schema induction performance on different domains and languages, we further conduct experiments on various datasets including finance (ChFinAnn (Zheng et al., 2019)), pandemic (Cov-19 and Pandemic (Shen et al., 2021)) and daily news (New York Time[3] and People's Daily 1946-2001[4]).

Implementation. We use BLOOM (Scao et al., 2022) in our experiments, which is a GPT-3 (Brown et al., 2020) like large-scale PLMs but is open-accessed. For text conceptualization, we sample in-context demonstrations from ACE (Doddington et al., 2004) and DuEE (Li et al., 2020b) for both English and Chinese datasets, respectively. The running environments and all hyper-parameters are in Appendix A.

[3] https://catalog.ldc.upenn.edu/LDC2008T19
[4] http://en.people.cn

3.2 Results of Event Schema Induction

This section assesses the event schemas induced by our method. Following previous studies (Huang et al., 2016; Shen et al., 2021), we evaluate event schemas via the event type/slot matching task. Both qualitative and quantitative results show the effectiveness of the proposed ESHer.

For qualitative results, Figure 3 shows several schemas induced by ESHer from the ERE-EN and ChFinAnn datasets, and the comparison with the results of human experts is also shown. From Figure 3, we can see that ESHer can induce high-quality event schemas: 1) most of the induced event types directly match the ones annotated by experts; 2) there is a large overlap between the automatically induced slots and the manually annotated ones; 3) some unmatched slots are also reasonable through our manual checking. This also shows that

Pandemic		New York Time		People’s Daily	
Type: Occur	**Type:** Administer	**Type:** Transport	**Type:** Charge	**Type:** 号召 (Call)	**Type:** 部署 (Deploy)
disease	vaccination	origin	perpetrator	号召者 (caller)	部署者 (deployer)
agent	inoculator	destination	plaintiff	号召对象 (callee)	部署内容 (content of deploy)
instrument	administrator	place	crime	号召内容 (content of call)	部署任务 (mission)
doctor	instigator	vehicle	evidence	地点 (place)	部署范围 (scope of deploy)
patient	instrumentation	agent	penalty	时间 (time)	部署时间 (time)
person	benefit	person	place		
place	place	entity			

Figure 4: Event schemas induced on varying domains, in which black denotes the reasonable event types/slots; red denotes the rejected predictions. More results can be found in Appendix C.

it is very hard to obtain high-coverage schemas only relying on experts. 4) we found some missing golden slots have been generated in text conceptualization but dropped in the confidence-aware structuralization step, therefore we believe the performance can be further improved by introducing human-in-the-loop. 5) with appropriate in-context demonstrations, ESHer can easily extend to different languages, e.g., English for ERE-EN and Chinese for ChFinAnn.

For quantitative results, we show the performances in Table 1. We can see that: for event type discovery, ESHer recover 21.05% out of 38 event types in ERE-EN and almost all (85.92%) discovered event types are acceptable. For event slot induction, ESHer recovers 11.30% out of 115 slots, 44.95% of discovered slots can be directly accepted, and 35.21% of slots can be selected from candidates. This shows that event schema is a challenging task due to the diversity and sparsity of event slots. On ChFinAnn, a typical dataset in the finance domain, we can see that ESHer is more effective and not only recover all event types but also discover lots of reasonable new types (100% Overlap and 72.73% Acceptable). This shows that domain-specific event schemas can be better induced, we believe this may be because domain-specific events are more salient in domain corpus. To assess the performance of graph-based schema aggregation, we manually check 100 individual schemas and cluster them, and its performance is shown as ESHer (upper bound) which can be regarded as the upper bound of graph-based schema aggregation. We can see that the performance gap between ESHer and ESHer (upper bound) is not large, which verifies the effectiveness of our graph-based schema aggregation component.

Methods	ARI	NMI	BCubed-F1
Kmeans	12.51	37.65	31.01
AggClus	13.11	39.16	31.20
JCSC	17.69	43.40	37.64
Triframes-CW	5.79	25.73	33.61
Triframes-Watset	7.53	47.43	24.04
ETypeClus	10.18	36.17	28.99
ESHer	**56.59**	**67.72**	**62.43**
- Salience	32.84	57.91	52.51
- Reliability	52.54	63.86	61.47
- Consistency	37.51	66.12	50.69
only Salience	38.75	66.34	50.69
only Reliability	33.98	62.43	51.09

Table 2: Event mention clustering results on ERE-EN. All values are in percentage. We run each method 10 times and report its averaged result for each metric. Note that for ESHer and its variants, due to the huge computing cost, we only run them once.

3.3 Results on Event Mention Clustering

We also evaluate the effectiveness of ESHer via the event mention clustering task. Following Shen et al. (2021), we select 15 event types with the most mentions and cluster all candidates into several groups for ERE-EN.

Evaluation Metrics. To evaluate whether clusters (Equation 6) align well with the original types, we choose several standard metrics: 1) **ARI** (Hubert and Arabie, 1985) measures the similarity; 2) **NMI** measures the normalized mutual information; 3) **BCubed-F1** (Bagga and Baldwin, 1998) measures the aggregated precision and recall. For all the above metrics, the higher the values, the better the model performance. The math formulas of these metrics are in Appendix B.

Baselines. We compare ESHer with the following feature-based methods: *Kmeans*, *AggClus*, *JCSC* (Huang et al., 2016), *Triframes-CW* (Ustalov et al., 2018), *Triframes-Watset* (Ustalov et al., 2018) and *ETypeClus* (Shen et al., 2021). We set all hyperparameters of these clusters using the default settings of Shen et al. (2021).

Experimental Results. Table 2 shows the overall results. For our approach, we use the full ESHer and its five ablated settings: ESHer-Consistency, ESHer-Salience, ESHer-Reliability, ESHer only Reliability and ESHer only Salience, where Salience, Reliability and Consistency denote different estimations described in confidence-aware schema structuralization. We can see that:

1. ESHer outperforms all baselines on ERE-EN on all metrics. ESHer achieves state-of-the-art performance with 56.59 ARI, 67.72 NMI and 62.43 BCubed-F1. We believe this is because ESHer fully leverages the in-context learning ability of PLMs, therefore the diverse and sparse challenges can be effectively resolved.

2. The proposed salience, reliability and consistency estimations are all useful and complementary to each other. Compared with the full ESHer model, all five variants show declined performance to different degrees. ESHer outperforms ESHer-Salience 23.75 ARI, 9.81 NMI and 9.92 BCubed-F1, and this result verifies the effectiveness of the salience score for identifying good slots. ESHer outperforms ESHer-Consistency 19.08 ARI, 1.60 NMI and 11.74 BCubed-F1, this shows that consistency estimation is also indispensable. These results also verify that high-quality slot sets are beneficial for graph-based aggregation.

3.4 Results on Different Domains

This section assesses event schemas induced on different domains such as pandemic and daily news. Figure 4 shows the result schemas and we can see that ESHer is robust in different domains and can be generalized in different settings. Furthermore, the results also present challenges: 1) **the granularity alignment problem**: the slots in the same schema may have different granularities, e.g., the "*person*" and "*doctor, patient*" in Schema 1 on the pandemic domain; 2) **the polysemy problem**: event type "*administer*" in Schema 2 on pandemic domain misdirects the slot "*administrator*"; 3) **emotional expressions**: event schema knowledge should be objective but "*instigator*" conveys negative sentiment.

4 Related Work

Event Schema Induction. Traditional event schema induction methods mostly mine event schemas from raw corpora, and two main categories of methods have been proposed: Bottom-up concept linking methods (Huang et al., 2016; He et al., 2022) discover event types/slots by parsing and linking event expressions to external schema resources such as FrameNet (Baker et al., 1998); Top-down clustering methods (Chambers, 2013; Cheung et al., 2013; Nguyen et al., 2015; Sha et al., 2016; Ahn, 2017; Yuan et al., 2018; Liu et al., 2019b; Shen et al., 2021) cluster event expressions according to pre-defined schema templates (e.g., the 5W1H template, or templates with the predefined number of event types/slots). Most recently, Li et al. (2022a) train a model to predict discovered event type names. Differently, this paper induces formal event schemas in unsupervised behaviour.

There were also some other studies such as script learning (Chambers and Jurafsky, 2008, 2009; Pichotta and Mooney, 2014) and event graph schema induction (Li et al., 2021; Zhao et al., 2022; Du et al., 2022), which focus on mining event relations and narrative schemas. This paper doesn't address these issues and leaves them as future works.

Harvesting Knowledge from Large-scale Language Models. Large-scale pre-trained language models (PLMs) such as GPT-3 (Brown et al., 2020) and BLOOM (Scao et al., 2022) have been verified containing massive knowledge such as linguistic knowledge (Liu et al., 2019a; Warstadt et al., 2020), factual knowledge (Petroni et al., 2019), commonsense knowledge (Zhou et al., 2020) and reasoning knowledge (Talmor et al., 2020). Furthermore, PLMs also have shown many emergent abilities (Wei et al., 2022) such as in-context learning, chain-of-thought reasoning, etc. In recent years, researchers start to learn how to harvest resources from PLMs, such as knowledge graphs (West et al., 2022) and explanation datasets (Li et al., 2022b). In this paper, we study how to harvest open-domain, high-quality and high-coverage event schemas from PLMs by leveraging the abilities of PLMs.

5 Conclusions

In this paper, we propose a new paradigm for event schema induction – knowledge harvesting from pre-trained language models (PLMs), which can effectively resolve the open, diversity and sparsity challenges by discovering, conceptualizing and structuralizing event schemas from PLMs. And an **E**vent **S**chema **H**arvest**er** (**ESHer**) is designed to automatically induce high-quality event schemas. Empirical results show that ESHer can induce high-quality and high-coverage event schemas on different domains. Event schemas are valuable resources, we want to harvest and release an open, large-scale event schema repository to research communities.

Limitations

In this paper, we harvest event schemas from large pre-trained language models. However, the huge cost of computing resources cannot be ignored. Therefore, in future work, we want to study how to distil open-domain, in-context learning abilities from massive language models to smaller ones.

Second, some outputs of large language models may be biased, e.g., containing mentions of discriminatory or offensive language. Thus, in practice, we can introduce human-in-the-loop to avoid these concerns.

Ethics Statement

In consideration of ethical concerns, we provide the following detailed description:

- All of the datasets used in this paper come from Linguistic Data Consortium (LDC)[5], previous studies (Shen et al., 2021) and publicly available sources. LDC datasets are accessible online for academic use with corresponding licenses and other datasets are freely accessible online without copyright constraints.

- The large-scale pre-trained language model BLOOM (Scao et al., 2022) is open-accessed in Huggingface Model Collections (Wolf et al., 2020).

[5]https://www.ldc.upenn.edu/

References

Natalie Ahn. 2017. Inducing event types and roles in reverse: Using function to discover theme. In *Proceedings of the Events and Stories in the News Workshop*, pages 66–76, Vancouver, Canada. Association for Computational Linguistics.

Giusepppe Attardi. 2015. Wikiextractor. https://github.com/attardi/wikiextractor.

Amit Bagga and Breck Baldwin. 1998. Entity-based cross-document coreferencing using the vector space model. In *36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 1*, pages 79–85, Montreal, Quebec, Canada. Association for Computational Linguistics.

Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. The Berkeley FrameNet project. In *36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 1*, pages 86–90, Montreal, Quebec, Canada. Association for Computational Linguistics.

Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. 2008. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*.

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.

Nathanael Chambers and Dan Jurafsky. 2011. Template-based information extraction without the templates. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 976–986, Portland, Oregon, USA. 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.

Nancy Chinchor, Lynette Hirschman, and David D. Lewis. 1993. Evaluating message understanding systems: An analysis of the third Message Understanding Conference (MUC-3). *Computational Linguistics*, 19(3):409–450.

Ferdinand De Saussure. 2011. *Course in General Linguistics*. Columbia University Press.

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

George Doddington, Alexis Mitchell, Mark Przybocki, Lance Ramshaw, Stephanie Strassel, and Ralph Weischedel. 2004. The automatic content extraction (ACE) program – tasks, data, and evaluation. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC'04)*, Lisbon, Portugal. European Language Resources Association (ELRA).

Zhendong Dong, Qiang Dong, and Changling Hao. 2010. HowNet and its computation of meaning. In *Coling 2010: Demonstrations*, pages 53–56, Beijing, China. Coling 2010 Organizing Committee.

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

Mutian He, Tianqing Fang, Weiqi Wang, and Yangqiu Song. 2022. Acquiring and modelling abstract commonsense knowledge via conceptualization. *arXiv preprint arXiv:2206.01532*.

Lifu Huang, Taylor Cassidy, Xiaocheng Feng, Heng Ji, Clare R. Voss, Jiawei Han, and Avirup Sil. 2016. Liberal event extraction and event schema induction. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 258–268, Berlin, Germany. Association for Computational Linguistics.

Lawrence Hubert and Phipps Arabie. 1985. Comparing partitions. *Journal of classification*.

Joseph Irwin, Mamoru Komachi, and Yuji Matsumoto. 2011. Narrative schema as world knowledge for coreference resolution. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning: Shared Task*, pages 86–92, Portland, Oregon, USA. Association for Computational Linguistics.

Ray S Jackendoff. 1992. *Semantic structures*. MIT press.

Heng Ji and Ralph Grishman. 2011. Knowledge base population: Successful approaches and challenges. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1148–1158, Portland, Oregon, USA. 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. 2020a. Connecting the dots: Event graph schema induction with path language modeling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 684–695, Online. Association for Computational Linguistics.

Sha Li, Heng Ji, and Jiawei Han. 2022a. Open relation and event type discovery with type abstraction. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*.

Shiyang Li, Jianshu Chen, Yelong Shen, Zhiyu Chen, Xinlu Zhang, Zekun Li, Hong Wang, Jing Qian, Baolin Peng, Yi Mao, et al. 2022b. Explanations from large language models make small reasoners better. *arXiv preprint arXiv:2210.06726*.

Xinyu Li, Fayuan Li, Lu Pan, Yuguang Chen, Weihua Peng, Quan Wang, Yajuan Lyu, and Yong Z hu. 2020b. Duee: a large-scale dataset for chinese event extraction in real-world scenarios. In *CCF International Conference on Natural Language Processing and Chinese Computing*.

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.

Nelson F. Liu, Matt Gardner, Yonatan Belinkov, Matthew E. Peters, and Noah A. Smith. 2019a. Linguistic knowledge and transferability of contextual representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1073–1094, Minneapolis, Minnesota. Association for Computational Linguistics.

Xiao Liu, Heyan Huang, and Yue Zhang. 2019b. Open domain event extraction using neural latent variable models. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2860–2871, Florence, Italy. Association for Computational Linguistics.

Yaojie Lu, Hongyu Lin, Jin Xu, Xianpei Han, Jialong Tang, Annan Li, Le Sun, Meng Liao, and Shaoyi Chen. 2021. Text2Event: Controllable sequence-to-structure generation for end-to-end event extraction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2795–2806, Online. Association for Computational Linguistics.

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

George A. Miller. 1992. WordNet: A lexical database for English. In *Speech and Natural Language: Proceedings of a Workshop Held at Harriman, New York, February 23-26, 1992*.

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.

Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? 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 2463–2473, Hong Kong, China. 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.

Teven Le Scao, Thomas Wang, Daniel Hesslow, Lucile Saulnier, Stas Bekman, M Saiful Bari, Stella Bideman, Hady Elsahar, Niklas Muennighoff, Jason Phang, et al. 2022. What language model to train if you have one million gpu hours? *arXiv preprint arXiv:2210.15424*.

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.

Jiaming Shen, Yunyi Zhang, Heng Ji, and Jiawei Han. 2021. Corpus-based open-domain event type induction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5427–5440, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Zhiyi Song, Ann Bies, Stephanie Strassel, Tom Riese, Justin Mott, Joe Ellis, Jonathan Wright, Seth Kulick, Neville Ryant, and Xiaoyi Ma. 2015. From light to rich ERE: Annotation of entities, relations, and events. In *Proceedings of the The 3rd Workshop on EVENTS: Definition, Detection, Coreference, and Representation*, pages 89–98, Denver, Colorado. Association for Computational Linguistics.

Alon Talmor, Yanai Elazar, Yoav Goldberg, and Jonathan Berant. 2020. oLMpics-on what language model pre-training captures. *Transactions of the Association for Computational Linguistics*, 8:743–758.

Dmitry Ustalov, Alexander Panchenko, Andrey Kutuzov, Chris Biemann, and Simone Paolo Ponzetto. 2018. Unsupervised semantic frame induction using triclustering. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 55–62, Melbourne, Australia. Association for Computational Linguistics.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: The benchmark of linguistic minimal pairs for English. *Transactions of the Association for Computational Linguistics*, 8:377–392.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed Chi, Quoc Le, and Denny Zhou. 2022. Chain of thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*.

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

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

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*.

Hongming Zhang, Daniel Khashabi, Yangqiu Song, and Dan Roth. 2020. Transomcs: From linguistic graphs to commonsense knowledge. In *Proceedings of International Joint Conference on Artificial Intelligence (IJCAI) 2020*.

Yanpeng Zhao, Jack Hessel, Youngjae Yu, Ximing Lu, Rowan Zellers, and Yejin Choi. 2022. Connecting the dots between audio and text without parallel data through visual knowledge transfer. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4492–4507, Seattle, United States. Association for Computational Linguistics.

Shun Zheng, Wei Cao, Wei Xu, and Jiang Bian. 2019. Doc2EDAG: An end-to-end document-level framework for Chinese financial event 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 337–346, Hong Kong, China. Association for Computational Linguistics.

Xuhui Zhou, Yue Zhang, Leyang Cui, and Dandan Huang. 2020. Evaluating commonsense in pretrained language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

A Running Environments and Hyper-parameters

We run all experiments on a single server with 56 CPU cores and an Nvidia TITAN RTX GPU, except the BLOOM model runs on a cluster with 5 NVIDIA A100 GPUs. Our codes rely on the Huggingface Library (Wolf et al., 2020). For all corpora, we filter sentences that are too long or contain too many numerical tokens. For baselines, we dump Wikipedia 20220301 version[6] and pre-process them by WikiExtractor (Attardi, 2015) as background corpora. And we use the OntoNotes sense grouping[7] as input verb sense dictionary for all clustering baselines. Table 3 and Table 4 show the detailed hyper-parameters.

Shared Hyper-parameter	ESHer	Baselines
Max Number of Tokens	256	
Ratio of Numerical Tokens	0.25	
Min Frequency of Verbs	3	
Salient Ratio of Verbs	0.25	
Min Frequency of Arguments	3	
Salient Ratio of Arguments	0.25	
Random Seed	1234	

Table 3: Shared hyper-parameters for ESHer and other baselines in Section 3.

Specific Hyper-parameter	English	Chinese
	ESHer	
In-context Demonstrations m	8	9
Candidates n	3	
PageRank β	0.8	
PageRank T	300	
PageRank ϵ	1e-6	
λ_1	1	
λ_2	1	
λ_3	3	
λ_4	1	
λ_5	1	
Threshold of Confidence	1/3	
	JCSC (2016)	
# Iteration Epoch	100	
	Triframes (2018)	
# Neighbor	10	
Weight	0	
	ETypeClus (2021)	
Aggregation Method	Concat	
Batch Size	64	
Distribution	Softmax	
Gamma	0.02	
Hidden Dimension	[500, 500, 1000, 100]	
Learning Rate	0.001	
Pre-train Epoch	1000	
Separately Decode	False	
Sort Method	Discriminative	
Temperature	0.1	
Threshold	0.05	
Train Epoch	100	
Update Interval	100	
Use Frequency	False	

Table 4: Specific hyper-parameters for ESHer and other baselines in Section 3..

B Evaluation Metrics for Event Mention Clustering

We implement all metrics based on the Scikit-learn codebase (Pedregosa et al., 2011). Following Shen et al. (2021), we denote the ground truth clusters as C^*, the predicted clusters as C, and the total number of event mentions as N.

- **ARI** (Hubert and Arabie, 1985) measures the similarity between two cluster assignments based on the number of pairs in the same/different clusters. Let $TP(TN)$ denotes the number of element pairs in the same (different) cluster(s) in both C^* and C. Then, ARI is calculated as follows:

$$\begin{aligned} ARI &= \frac{RI - \mathbb{E}(EI)}{\max RI - \mathbb{E}(EI)} \\ RI &= \frac{TP + TN}{N} \end{aligned} \tag{7}$$

  where $\mathbb{E}(EI)$ is the expected RI of random assignments.

- **NMI** denotes the normalized mutual information between two cluster assignments. Let $MI(\cdot;\cdot)$ be the Mutual Information between two cluster assignments, and $H(\cdot)$ denotes the Entropy. Then the NMI is formulated as follows:

$$NMI = \frac{2 * MI(C^*; C)}{H(C^*) + H(C)} \tag{8}$$

- **BCubed-F1** (Bagga and Baldwin, 1998) estimates the quality of the generated cluster assignment by aggregating the precision and recall of each element. B-Cubed precision,

[6] https://dumps.wikimedia.org
[7] http://verbs.colorado.edu/html_groupings

Pandemic

Type: Infect-Infection	**Type:** Track	**Type:** Decree	**Type:** Understand	**Type:** Invent
agent	agent	adjudicator	cause	agent
instrument	place	agent	mechanism	inventor
person	staff	claim	phenomenon	instrument
place	state-health	decision	treatment	place
target		issue		virus
		place		

New York Time

Type: Report	**Type:** Ban-Embargo	**Type:** Delay	**Type:** Award	**Type:** kill
agent	agent	delayer	adjudicator	attacker
place	policy	participant	winner	instrument
reporter	product	period	place	place
statement	place	place	reward	query
				victim

People's Daily

Type: 救济 (Relief)	**Type:** 组织 (Organize)	**Type:** 实行 (Implement)	**Type:** 冲突 (Conflict)	**Type:** 投诉 (Complaint)
资金来源 (source of funds)	组织者 (Organizer)	执行方 (executor)	斗争内容 (content of conflict)	问题内容 (issue)
救济者 (reliever)	参与者 (participant)	执行对象 (target)	斗争双方 (participant)	投诉者 (complainant)
受救济者 (recipient)	目的 (purpose)	实行内容 (content of implementation)	地点 (place)	投诉对象 (target)
救济方式 (remedies)	地点 (place)	政策制定部门 (policy-making department)	时间 (time)	投诉时间 (time)
地点 (place)	时间 (time)	地点 (place)		
时间 (time)		时间 (time)		

Figure 5: Schemas induced on varying domains.

recall, and F1 are thus calculated as follows:

$$
\begin{aligned}
\textit{BCubed-P} &= \frac{1}{N}\sum_{i=1}^{N}\frac{|C(e_i)\cap C^*(e_i)|}{|C(e_i)|} \\
\textit{BCubed-R} &= \frac{1}{N}\sum_{i=1}^{N}\frac{|C(e_i)\cap C^*(e_i)|}{|C^*(e_i)|} \\
\textit{BCubed-F1} &= \frac{2}{\textit{BCubed-P}^{-1}+\textit{BCubed-P}^{-1}}
\end{aligned} \tag{9}
$$

where $C^*(\cdot)$ $(C(\cdot))$ is the mapping function from an element to its ground truth (predicted) cluster.

C More ESHer Outputs

In this section, we provide more induced schema from varying domains, including pandemic (Cov-19 and Pandemic (Shen et al., 2021)) and daily news (New York Time[8] and People's Daily 1946-2001[9]).

[8]https://catalog.ldc.upenn.edu/LDC2008T19
[9]http://en.people.cn