A Monte Carlo Language Model Pipeline for Zero-Shot Sociopolitical Event Extraction

Erica Cai **Brendan O'Connor**
Manning College of Information & Computer Sciences
University of Massachusetts Amherst
{ecai,brenocon}@cs.umass.edu

Abstract

Current social science efforts automatically populate event databases of "who did what to whom?" tuples, by applying event extraction (EE) to text such as news. The event databases are used to analyze sociopolitical dynamics between actor pairs (dyads) in, e.g., international relations. While most EE methods heavily rely on rules or supervised learning, *zero-shot* event extraction could potentially allow researchers to flexibly specify arbitrary event classes for new research questions. Unfortunately, we find that current zero-shot EE methods, as well as a naive zero-shot approach of simple generative language model (LM) prompting, perform poorly for dyadic event extraction; most suffer from word sense ambiguity, modality sensitivity, and computational inefficiency. We address these challenges with a new fine-grained, multi-stage instruction-following generative LM pipeline, proposing a Monte Carlo approach to deal with, and even take advantage of, nondeterminism of generative outputs. Our pipeline includes explicit stages of linguistic analysis (synonym generation, contextual disambiguation, argument realization, event modality), *improving control and interpretability* compared to purely neural methods. This method outperforms other zero-shot EE approaches and outperforms naive applications of generative LMs by at least 17 F1 percent points. The pipeline's filtering mechanism greatly improves computational efficiency, allowing it to perform as few as 12% of queries that a previous zero-shot method uses. Finally, we demonstrate our pipeline's application to dyadic international relations analysis.

1 Introduction

Event extraction (EE) infers actions and participants involved with them as structured *events* from unstructured text data, and is useful for many areas such as knowledge graph construction and intelligent question answering (Gao et al. (2016); Liu et al. (2020); Li et al. (2021); Cao et al. (2020)). However, many EE methods require training on substantial, difficult-to-collect annotated data (Li et al. (2022); Liu et al. (2016)). *Zero-shot* EE combats this challenge by providing flexibility to extract events from text without annotated data (Huang et al. (2018); Zhang et al. (2021, 2022b,a); Lyu et al. (2021); Mehta et al. (2022); Gao et al. (2016)), therefore allowing users to ask for specific new types of events (e.g. cyberwarfare, environmental incidents) that may be enormously important for specific research needs, but which do not correspond to ones defined in previous event ontologies, where annotations may have taken years to produce (e.g. in political science (Gerner et al. (2002)) or natural language processing (Doddington et al. (2004))). The input for zero-shot EE is a corpus and a user-specified set of *event classes* (types), with textual names and optionally descriptions. The output are event *instances* which contain structured details about each event class occurrence in text (Fig. 1).

Workshop on Instruction Tuning and Instruction Following at NeurIPS 2023.

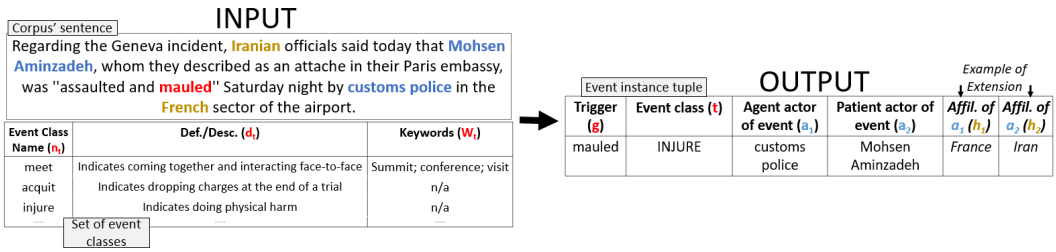


Figure 1: Input and output for zero-shot dyadic event extraction (from *New York Times*, Jan. 1, 1987 (Sandhaus (2008))).

Dyadic EE extracts action occurrences between interacting pairs of actors (*dyads*; e.g. Wasserman and Faust (1994)) which is useful for constructing knowledge graphs that depict sociopolitical relations. Earlier work has explored a variety of other methods for dyadic EE; our view of the task follows original rule-based NLP efforts in international relations (Schrodt and Gerner (1994)) that extract events between *source* and *target* actors (i.e., linguistic theory's *agent-patient* roles (Dowty (1991); Williams (2015))), using pattern-based extraction from a fixed ontology (Gerner et al. (1994, 2002); Schrodt and Gerner (2004); Boschee et al. (2013)).[1] Machine learning approaches have also explored dyadic EE (O'Connor et al. (2013); Beieler (2016); Halterman (2020); Norris et al. (2017)), with applications in conflict prediction, social networks, and misinformation (O'Brien (2010); Brandt et al. (2011); Smith et al. (2020); Stoehr et al. (2023)). In contrast to these knowledge engineering and machine learning approaches, zero-shot EE promises to use significantly less expert knowledge and data, aside from conventional pretraining outside the task.

Fig. 1 illustrates components of dyadic EE on an example sentence. Following prior (non-dyadic) EE NLP research, we divide it into subtasks of *event detection* to identify an event instance of a particular class (e.g. INJURE), and *argument extraction* to collect involved agent-patient actor pairs of people or organizations (`customs police`, `Mohsen Aminzadeh`).[2] Our main dyadic EE application also calls for identifying higher-level entities, such as countries (`France` and `Iran`) that actors are affiliated with to analyze their dynamics (Schrodt and Gerner (1994)).

We seek to apply zero-shot EE to the dyadic EE task, but unfortunately find that many current zero-shot EE methods—based on, for example, textual entailment—suffer from word sense ambiguity, modality, and efficiency issues. For example, some methods only take an event class name from a user to define the event class, but this causes ambiguity (e.g. is *charge* about money, indictment, or attack?).[3] Methods may also be sensitive to event modality, i.e., whether the event instance really occurred, or is a hypothetical, or may occur in the future. Finally, some methods are very computationally expensive, because they require making inferences exhaustively for many text spans.

To address these challenges and improve performance, we propose using generative large language models (LLMs or simply LMs), which have significant potential to capture interesting aspects of meaning and nuances of language (Raffel et al. (2020); Brown et al. (2020); Rogers et al. (2020); Ouyang et al. (2022); Piantadosi (2023)), but a simple application of them to zero-shot event detection produces poor results (Gao et al. (2023)). We further investigate by building on recent zero-shot approaches based on text entailment (TE) (Lyu et al. (2021); Yin et al. (2019)) to develop naive generative LM baselines, but find poor performance.

Instead of naive querying, we propose using a generative LM in a *fine-grained event extraction pipeline*, which allows for more explicit control of handling particular semantic properties. This follows a very widespread approach in rule-based and non-neural EE, including early work (Gildea and Jurafsky (2002); Ahn (2006); Ji and Grishman (2008)) which uses pipelines of classifiers and rule-based extractors to incrementally build and refine event structures, often building on "classic NLP pipelines" predicting linguistic structures like POS tags, parse trees, or named entities (Toutanova et al. (2005); Finkel et al. (2006); Bunescu (2008); Watanabe et al. (2008); Manning et al. (2014);

[1] While this line of work requires no annotated text, it relies on hand-engineering thousands of patterns.

[2] More broadly, arguments of interest are actors with sociopolitically relevant agency; for example, military vehicles.

[3] In the zero-shot setting, apparently only one work addresses this issue (Zhang et al. (2022a)).

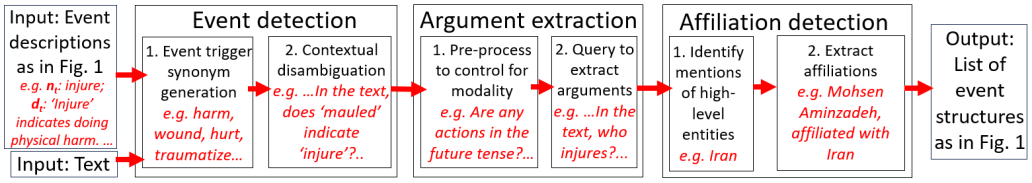


Figure 2: This work's multi-stage LM pipeline, with red text for prompts and outputs corresponding to Fig. 1's example.

Peng et al. (2015)). Instead of predicting categories from a general-purpose syntactic-semantic ontology, we query an LM for task-specific inferences.

Fig. 2 illustrates our pipeline on the event class, INJURE from Fig. 1. It implements event detection as lexical semantic stages of (1) event trigger synonym generation, and (2) contextual disambiguation, to detect if a trigger usage indicates an event instance. Next, argument extraction uses extractive QA (Du and Cardie (2020); Liu et al. (2020)) to identify arguments, and controls for event modality. Finally, we add detection of actors' affiliations with higher-level entities (e.g. countries, companies) to perform an international relations case study (§7).[4]

We propose a competitive zero-shot EE method that relies on generative LMs, with contributions:

- **Monte Carlo** – We propose probabilistically sampling many sets or single-value outputs from the LM at each stage, since LM outputs may be non-deterministic. This improves robustness, and through the temperature hyperparameter, allows the user to tune cost/performance tradeoffs in synonym generation (§5).
- **Interpretability** – Our pipeline improves interpretability compared to an all-at-once neural black box, allowing control for handling particular semantic properties.
- **Semantic challenges** – Our approach addresses significant errors from lexical ambiguity and allows control for event modality.
- **Performance/efficiency** – Our approach outperforms other recent zero-shot EE approaches on the widely-used Automatic Content Extraction (ACE) dataset (Doddington et al. (2004); Li et al. (2022)), outperforms a naive application of generative LMs by at least 17 F1 percent points, and could perform just 11.3% of queries that a previous zero-shot TE approach performs.
- **Bringing EE beyond artifical NLP datasets** – We demonstrate the flexibility of our approach for sociopolitical analysis.

We hope the flexibility and benefits of zero-shot generative LM pipelines can help support future semantic extraction applications.

2 The Zero-shot EE Task

Formally, the input of our zero-shot EE task is the corpus' sentences $\mathcal{S}$ and event descriptions $\{\langle n_t, d_t, W_t\rangle \mid t \in \mathcal{T}\}$ where $\mathcal{T}$ is a set of event classes (e.g. INJURE), n_t is an event class name (e.g. injure), d_t is a short definition or description of the event class, and W_t is an optional set of keywords that help to describe it (possibly empty). Our task does not need annotated examples.

The output are sets of event instances for each sentence in $\mathcal{S}$ as in our running example in Fig. 1. Each event instance is a tuple $\langle t, g, a_1, a_2\rangle$, where:

- $t \in \mathcal{T}$ is the *event class* (e.g. INJURE).
- g is the *event trigger*, a word in the sentence that identifies the event class (e.g. `mauled`).
- $\{a_1, a_2\}$ are actor pair *event arguments* explicitly mentioned in the sentence (e.g. `customs police`, `Mohsen Aminzadeh`).

The terminology of triggers, arguments, and classes follows the EE literature. We consider an additional variation to only extract actor participants as arguments—*who did what to whom?*—where a *dyadic* pair consists of one actor instigating the event (*agent*, a_1) and the other receiving or being

[4]As a positive example, we note the CASE ACL-IJCNLP 2021 workshop Hürriyetoğlu (2021)'s shared tasks on concrete sociopolitical event extraction tasks, such as fine-grained ACLED event detection.

	Text Entailment		Generative	
	roberta	deberta	text-davinci-003	gpt-3.5-turbo
1 Lyu et al. (2021)	33.6	31.1	37.6	40.7
2 Wording change	22.4	33.4	44.5	41.7
3 Add **def.** to [1]	18.8	4.3	35.9	29.0
4 Add **def.** to [2]	21.8	6.7	44.1	42.5

Table 1: Simple query baseline F1 performance.

affected by it (*patient*, a_2). Dyadic arguments have tremendous social scientific utility to characterize social networks and relational dynamics, and are grounded in Dowty (1991)'s semantic proto-role theory. Although dyadic EE may superficially seem similar to binary relation extraction, events mostly involve actions constrained to a specific time frame, therefore allowing analysis into how pair dynamics change over time; in contrast, relations tend to describe static associations (Agichtein and Gravano (2000); Yates et al. (2007); Cui et al. (2017)).

Further, we extend our task to add *affiliation detection*, extracting a higher level entity that each actor is affiliated with if any (see §6 and Fig. 1):

- *Additional input*, $\mathcal{C}$: The name of a higher level entity *category* (e.g. country, company).
- *Additional output*, $\langle h_1, h_2 \rangle$ s.t. $a_1 \in h_1$, $a_2 \in h_2$, and $h_1, h_2 \in \mathcal{C}$ (possibly empty). (In Fig. 1, $h_1 =$ `France`, $h_2 =$ `Iran`, and $\mathcal{C} =$ country.)

Such high-level entity actor information is useful for predicting future conflict or cooperation (Stoehr et al. (2021); Brandt et al. (2011)). For future reference, we refer to the subtasks of extracting outputs $\langle t, g \rangle$ as *event detection*, $\langle a_1, a_2 \rangle$ as *argument extraction*, and $\langle h_1, h_2 \rangle$ as *affiliation detection*.

3 Unreliability of Naive LLM Queries

Given significant potential of generative LLMs in recent zero-shot work, we naively test them for event detection by transferring promising existing zero-shot TE or QA approaches to LLMs. Variations of a TE or QA-based approach have been studied in supervised EE (Du and Cardie (2020); Liu et al. (2020)), zero-shot relation extraction (Levy et al. (2017)), zero-shot text classification (Yin et al. (2019)), and zero-shot EE (Lyu et al. (2021)). TE outputs a probability that a premise of text implies a hypothesis that the text contains an event instance (e.g. *This text is about injuring.*) (Lyu et al. (2021)) (Yin et al. (2019) explore hypothesis phrasing).

Our naive approach is brute force, aiming to detect positive event instances by exhaustively performing TE over the Cartesian product of all sentences and hypotheses enumerating each event class. We use *"This text is about..."* as the TE hypothesis (details in §10.4), converting it to a boolean question to explore generative LLM performance (e.g. *Is the text about injuring?*) (row 1, Table 1). We add short definitions to prompts and hypotheses (rows 3,4) to address ambiguity and explore effects of wording, replacing *"is about"* with *"discusses"* (row 2) (§10.4). Our evaluation is over every sentence in the same 40 documents and 33 event classes of ACE (§6) as many EE evaluations and explores roberta and deberta (large) for TE and text-davinci-003 and gpt-3.5-turbo as generative LMs for QA.

The results suggest that directly querying over long text spans may not be promising for improving performance. Table 1 shows some variation, which supports Gao et al. (2023) and Yuan et al. (2023)'s findings, and shows that adding a short definition in a TE hypothesis causes performance to plummet.

While our naive approach exhaustively queries on all pair combinations of sentences and hypotheses about event classes, queries are over entire sentences; it cannot detect multiple instances of the same event class in one sentence. Less naive approaches could instead query over substrings of sentences based on SRL model outputs, but this involves more computation and does not achieve significantly better performance (Lyu et al. (2021)).

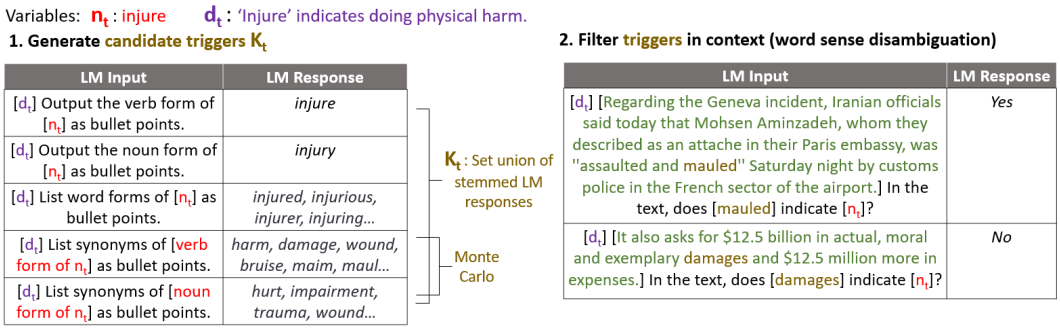


Figure 3: Prompt-based pipeline for event detection (§4.1) on the running example from Fig. 1.

4 Event Extraction Pipeline with Generative Language Models

Given unreliable and poor performance of *naive*, exhaustive generative LM prompting, we propose a pipelined instruction-following approach to LM-based, zero-shot dyadic event extraction, which includes (to our knowledge) the *first competitive zero-shot event detection method using generative LMs*. It involves separate steps for event detection—with synonym generation, filtering, and disambiguation (§4.1)—then argument extraction (§4.2) and later affiliation detection (§7). Throughout, it uses a Monte Carlo (MC) sampling method (proposed in §5) to improve robustness, and to control size and diversity of candidate trigger synonym sets.

4.1 Event Detection

We propose a generative LM method for event detection which queries over words and phrases instead of over entire sentences, and illustrate the approach on our running example in Fig. 3. Our method is more selective about making queries than the naive exhaustive LM querying approach, and this greatly improves efficiency (§6). Our method finds a trigger in the following phases:

Step 1. Generate a set of candidate trigger word stems K_t for each event class t given $\langle n_t, d_t, W_t \rangle$.

Step 2. Identify if a candidate trigger stem $k_t \in K_t$, if in a sentence, is a stem of an actual trigger word for an event instance (disambiguation).

The event detection output is sets of event type and trigger tuples $\langle t, g \rangle$ corresponding to each sentence.

Step 1: Generate candidate trigger terms. *For each event class t, generate a (possibly overcomplete) set of candidate trigger words and phrases, to identify event instances of the class, as K_t.*

K_t is populated by expanding n_t to many more lexical items, generating a set of its **inflections**, **noun** and **verb** forms of n_t, and their respective **synonym sets** (3). This expansion is also performed for each $w_t \in W_t$; K_t is the union of these expansions, all generated by LM prompting (Fig. 3; §10.5), then stemmed (Porter (2001)) and deduplicated. Each query includes definition d_t, helping the model use an appropriate word sense of n_t. Synonym sets are generated with a Monte Carlo (MC) method (§5).

For example, n_t=*injure* yields 68 word stems in K_t, including near-synonyms (*hurt*), many hyponyms (*wound*, *maim*), and some only moderately similar terms (*torment*, *loss*). We prefer to possibly overgenerate, since the next step removes spurious matches. While we explore using an LLM for this lexical expansion, alternative resources such as word embeddings (e.g. GLOVE; Pennington et al. (2014)) or hand-built lexical databases (e.g. WordNet; Miller (1995)), could replace this step in future work. Possible LLM advantages include accommodation of multiword n_t and w_t (e.g. *start organization*), and flexibly distant temperature-based MC control over synonym set size (§5).

Step 2: Filter triggers (disambiguation). *Within a sentence context, determine if candidate trigger stem k_t actually identifies the event.*

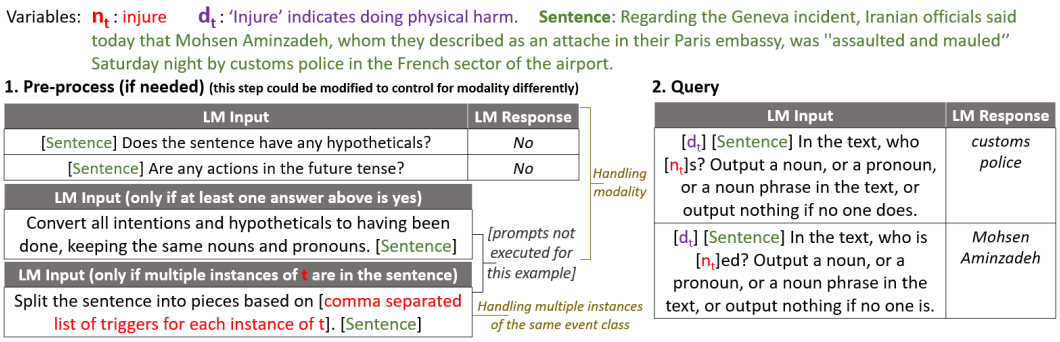


Figure 4: Prompt-based pipeline for argument extraction (§4.2) on the running example from Fig. 1.

Our method analyzes all sentences $s \in \mathcal{S}$ for any stems $k_t \in K_t$ of a class $t \in \mathcal{T}$.[5] Each match is disambiguated with a generative model, asking if the word that has k_t as a substring indicates class t (in the form of n_t or w_t; step 2 of Fig. 3). The query includes definition d_t. A *yes* answer indicates successful event detection, while *no*, otherwise. Monte Carlo voting improves robustness (§5).

Summary. This two-stage system (Fig. 3) helps ensure both efficiency and accuracy—Step 1 greatly reduces the number of sentences requiring LLM analysis compared to previous zero-shot EE approaches and the naive approach (efficiency analysis in §5), while Step 2 protects against spurious matches. We find that irrelevant trigger candidates from Step 1 do not change performance much (§10.5), but too many matches can significantly hurt disambiguation efficiency (see §5). Note that prompts and tense-insensitive stem matching naturally disregard modality, allowing detection of events with future, past, hypothetical, or other semantic modalities, which we could control for in §4.2.

4.2 Event Argument Extraction

We propose a multistage generative LM argument extraction method that is similar to that of Du and Cardie (2020) and Liu et al. (2020), which use extractive QA to determine arguments (e.g. *"Who injures?"*, illustrated in Fig. 4, following the running example in Fig. 1). Output are actor argument tuples $\langle a_1, a_2 \rangle$ (if existing) corresponding to each input tuple of event type and trigger ($\langle t, g \rangle$) from event detection. Our task aims to extract actor pairs only, not single arguments.

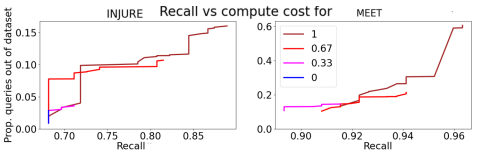
Query step. Querying extracts dyadic agent and patient actors $\langle a_1, a_2 \rangle$ for each event instance. Each query is over a sentence or a modified output of *pre-processing*. For event names that are regular verbs, the query has the form *Who [n_t]s?*, *Who is [n_t]ed?* (Fig. 4); this format has simple modifications for accommodating other forms of n_t (e.g. multi-words) and ensuring grammatical correctness (§10.5). The query includes definition d_t (addressing ambiguity) and an instruction to specify that the answer belongs to the text. MC (§5) increases robustness.

Pre-process step. Pre-processing text (if needed) before querying allows *controlling for modality:* since query phrasing assumes a current or past-tense event instance, the method asks (1) if a sentence has hypotheticals or intentions; if so, it (2) converts them to past-tense using an LLM instruction (Fig. 4). The step could easily be modified to consider only past-tense events (remove pre-processing) or consider additionally just future-tense events (remove consideration of hypotheticals), depending on the application. In addition, to extract arguments for each unique instance of the same event class in a sentence (if more than one), the method prompts to split the sentence into text spans corresponding to each instance of t (Fig. 4).

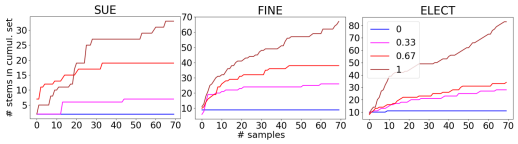
5 Monte Carlo (MC) for Synonym Set Generation and Robustness

We propose an MC approach that is useful for generating lexical resources and producing more robust outputs, positively impacting performance and efficiency of our pipeline. Generative LMs

[5] Stems are matches as prefixes to words in s.



(a) Recall vs. compute cost for INJURE (left) and MEET (right) in ACE for different temperatures.



(b) Cumulative set sizes over 70 samples for 3 events of temps. in range $\{0, 0.33, 0.67, 1\}$

Figure 5: Monte Carlo candidate trigger generation analysis given different event names.

are nondeterministic, which benefits our approach by allowing construction of diverse cumulative synonym sets from many samples of synonym sets, but decreases output robustness on QA tasks.

MC to generate synonym sets. MC allows control for broadness of a synonym set while balancing compute cost during event detection. Let a word's synonym set be A: our method populates it by repeatedly prompting to create synonym sets A_y:

$$A = \cup_{y=1}^{Y} A_y \tag{1}$$

i.e. any $a \in A$ must appear in at least one sample.[6] Our goal is to construct broad synonym sets (or over-generate — while irrelevant synonyms in a trigger stem set do not affect performance much (§10.5), they decrease efficiency, necessitating more queries during filtering (disambiguation step)).

Recall and efficiency. Fig. 5a shows that increasing the temperature hyperparameter, which controls the extent of LM output randomness, corresponds with higher recall ($\%$ of event instances for t that have triggers in K_t) but also higher compute cost ($\%$ of queries performed out of number of sentences in ACE) as a tradeoff. Further, temperature 1 may result in far lower efficiency. Yet, the efficiencies are significant improvements to TE (§6).

Additionally, higher temperatures correlate with larger cumulative synonym sets, which nearly converge for temperatures ≤ 0.67 (e.g. SUE, FINE in Fig. 5b converge for temperature 0.67.). Therefore, drawing more samples at lower temperatures does not seem likely to increase synonym set sizes much.

In summary, temperature is crucial for controlling synonym set size (Fig. 5b) which corresponds to increasing recall (Fig. 5a), but may also incur higher compute cost. Informed by these results which also hold over the rest of the event names in ACE, our MC approach generates synonym sets with temperature 0.67 over 70 samples. While an alternative of MC is to run a single LLM prompt to generate a specific number of synonyms, this is not flexible: in Fig. 5b, after 70 samples for temperature 1, ELECT has 80 unique synonyms but SUE has 30.

MC for pipeline QA tasks. Unfortunately, even for temperature 0, LM outputs (GPT-family) are non-deterministic, which hurts robustness of our pipeline's QA outputs. We experiment and observe that most answers out of 10 are unanimous, and if any, only a few (typically one) differ – the proportion of unanimous boolean outputs using our system on text-davinci-003 is 0.9986, 0.9938, and 0.9887 for temperatures 0, 0.33, and 0.67 respectively. Our method takes the majority (similar to self-consistency prompting; (Wang et al. (2023))) from 9 samples of each query in §6 but could sample much fewer (e.g. 3; see §10.6).

Hyperparameter and prompt sensitivity. Our fine-grained pipeline approach is less susceptible to hyperparameter and prompt variability: precision of event detection does not decrease with temperature because the disambiguation step tends to prevent irrelevant synonyms from counting towards false positives (§10.5). Number of samples does not need to be tuned; we advocate a large number for generating cumulative synonym sets, which nearly converge, and small for QA, where it has little effect. Finally, since tasks are fine-grained, we were able to choose the first prompts that we tried on the validation set without prompt engineering (§10.5), and MC helps to further weaken variability.

[6] Future work could explore higher thresholds of τ, giving another method of controlling A's breadth.

Setting	System	Ev Dect	Arg Ext (non-dyad)	Arg Ext (dyad)
supervised	Lin et al 20	74.7	56.8	-
zero-shot	Huang et al 18	49.1	15.8	-
	Zhang et al 21	53.5	6.3	-
	Lyu et al 21	41.7	16.8	22.4
	This work	**61.2**	n/a	**28.6**
zero-shot	Naive exhaust (§3)	44.5	n/a	n/a

Table 2: F1 micro-average performance of our and other methods on ACE, on event detection and argument extraction.

6 Evaluation

We evaluate event detection and argument extraction on the Automatic Content Extraction (ACE) dataset Ahn (2006), which contains event class, trigger, and argument annotations over 33 event classes and 598 documents of news articles, conversations, and blogs. The ACE dataset is known as most popular for evaluating EE; many evaluations use the same train, validation, and test split Ji and Grishman (2008); Liao and Grishman (2011); Hong et al. (2011); Nguyen and Grishman (2015); Nguyen et al. (2016); Liu et al. (2016); Huang et al. (2017); Sha et al. (2016); Chen et al. (2017); Huang et al. (2018); Liu et al. (2017); Zhang et al. (2019); Wadden et al. (2019); Chen et al. (2020); Du and Cardie (2020); Liu et al. (2020); Ahmad et al. (2021). Our evaluation is over the same 40 documents as many EE evaluations, following Wadden et al. (2019)'s pre-processing and modifications Cai and O'Connor (2023). We verified LM instructions on the same 30 validation documents used by others (§10.3).

In Table 2, we present results of our pipeline using text-davinci-003 (Ouyang et al. (2022)). Unfortunately, baselines performing both event detection and argument extraction tasks in zero-shot EE are sparse.

Event detection. We provide F1 micro-average performance of event detection for our pipeline and baselines in column 3 of Table 2, where a true positive corresponds to a correctly identified event class (t) in a sentence. Our approach outperforms the other recent zero-shot EE approaches (see §10.7 for populating d_t, W_t, and complications in ACE). Its performance also exceeds that of our best-performing naive exhaustive LM query baseline (§3) by 16.7 F1 percent points, and is much closer to that of supervised EE than the other zero-shot EE performances are.

Argument extraction. For reference, column 4 shows non-dyadic argument extraction performance (not applicable for our method) on ACE, where each event class corresponds to argument types beyond actors (e.g. *location*), and F1 performance is calculated based on correctness of individual arguments—the performances seem poor.

Finally, we provide dyadic argument extraction performance on ACE, counting a true positive only when the event and *both* actors (t,a_1,a_2) are correct. For each event class in ACE, we map pairs of argument types (e.g. *{Agent,Victim}*; *{Attacker, Target}*; *{Prosecutor, Defendant}*) to analogous *agent* and *patient* types defined in §2 (if they exist) to serve as a_1 and a_2. Evaluation is over the subset of event classes (20 out of 33) that have mappable dyadic arguments (§10.7). Since this zero-shot subtask is new, we aim to adapt other evaluations to this setting. However, we find incompatibility between code for the zero-shot baselines and models that they depend on (e.g. SRL), which underwent extensive updates. We thankfully got access to outputs of Lyu et al. (2021)'s approach, and reproduced our dyadic evaluation setting for it to serve as a competitive baseline. Our dyadic approach outperforms it (col. 5), which we find outperforms other methods for non-dyadic argument extraction (column 4).

Alternatives. Besides text-davinci-003 (used for the Table 2 evaluation), we could use gpt-3.5-turbo, which is much cheaper and outperforms the baselines (e.g. 58 F1 on event detection). Using open-source generative LMs would have research and application advantages, but we find Llama 2 (Touvron et al. (2023)) and Alpaca inconsistently produce inadequately formatted outputs for synonym generation (§10.3). We hope future work could explore more open-source LMs, which are rapidly improving.

Efficiency analysis. Our approach is efficient because it uses candidate triggers as a filter for performing LM queries. To analyze efficiency gains, we count boolean LM queries for disambiguation

(§4.1), which dominates pipeline runtime (by contrast, argument extraction queries are sparse and synonym set generation queries do not vary with the size of the data). After considering MC, we find our method performs between 3% and 30% of Lyu et al. (2021)'s queries, varying due to selecting between 1 to 9 MC samples per query, where a small number (e.g. 3, leading to performing 11.3% of Lyu et al. (2021)'s queries) seems sufficient (§5).

7 Affiliations and International Relations

A benefit of our dyadic EE pipeline is flexibility to add new application-specific components. To define vertexes in the resulting dyadic event graph, we extend the pipeline with country-level affiliation detection for international relations analysis.

Specifically, our extension has a sociopolitical application of extracting events between high-level entities that actor arguments are representatives of (e.g. countries; companies; any type of organization). The task input (§2) are sets of event instance tuples of type, trigger and arguments $\langle t, g, a_1, a_2 \rangle$ from argument extraction, and $\mathcal{C}$, which is a higher level entity *category*. Output are event instances with additionally affiliations of arguments $\langle h_1, h_2 \rangle$ where $h_1, h_2 \in \mathcal{C}$ (in Fig. 1, h_1 and h_2 are `France`, `Iran`). In the case study, we set $\mathcal{C}$ as *countries* and *rebel groups*. To perform affiliation detection, our approach (1) finds all mentions of countries, and (2) determines if a_1 or a_2 is affiliated with any.

Step 1: Find country references. Our method finds country references through keyword matching, primarily using the `CountryInfo.txt` database (Schrodt, 2015)[7][8] which includes noun, adjective, acronym, and misspelled references to countries (§10.8). To associate cities or towns to a particular country, our method uses SpaCy 3 to identify all `GPE` and `NORP` named entities and checks if any is a location in a country by using the geocoding `Nominatim` API (having access to `OpenStreetMap` data).[9] It selects the top country output if a named entity is ambiguous (toponym disambiguation). To evaluate, we select 100 sentences from the New York Times Corpus (LDC2008T19 (Sandhaus (2008))) that mention at least two countries based on the dictionary, and compare our annotations against this step's, finding 100% accuracy.

Step 2: Extract affiliation. To determine if an actor argument extracted by dyadic EE is affiliated with a country, our method (1) identifies affiliation if any country mention is directly part of the argument span (a_1 or a_2) or (2) iterates through each country mention and asks an LLM, *"In the text, is [actor] affiliated with [country mention]?"*, applying MC to increase robustness. To evaluate, we select 100 samples that our method identified actors and countries for, and compare our annotations against the method, finding 86% accuracy.

Data and results. We demonstrate our method and extension on NYT articles in 1987-1988 over self-defined event classes, using gpt-3.5-turbo for event detection (10x cheaper than text-davinci-003 with similar performance (§6)) and text-davinci-003 for the other steps.

Proxy war. In Fig. 6, we observe interactions that are consistent with events during the 1980s, where US-backed rebels, the Contras, fought against the USSR-backed Nicaraguan government. This was a major Cold War proxy conflict. In 1987 and 1988, the USSR mostly aided Nicaragua (93%, 100% in 1987, 1988 respectively) while the US mostly aided Contras. In 1988, the US seems to aid Nicaragua slightly more because NYT began referring to Contras as the "new Nicaraguan government." We also find neither Contras nor Nicaragua aids the US or USSR, as a simple illustration of center-periphery imperial dependency structure (Galtung (1971)).

8 Event Extraction Literature

While our approach is zero-shot, annotated event instances have been critical for most EE, from older pattern matching approaches, for helping to learn and apply rules for identifying instances and arguments from surrounding context (Riloff (1993); Riloff and Jones (1999); Califf and Mooney (2003); Liao and Grishman (2011); Boschee et al. (2013)), to feature-based approaches, for helping to learn effective features for statistical models (Ji and Grishman (2008); Gupta and Ji (2009); Hong et al. (2011); Li et al. (2013); McClosky et al. (2011)), to deep learning approaches (Zhang et al.

[7] `https://github.com/openeventdata/CountryInfo`

[8] `https://github.com/brendano/OConnor_IREvents_ACL2013`

[9] `https://www.openstreetmap.org/`

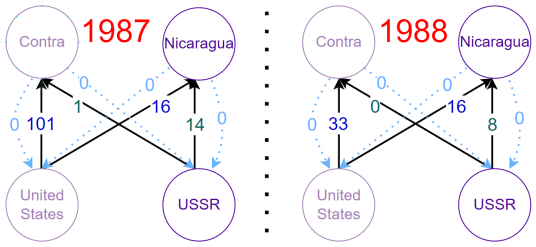


Figure 6: Dyadic event frequency network of AID in the proxy war during 1987 and 1988.

(2019); Nguyen and Nguyen (2019); Li et al. (2020a); Lin et al. (2020); Li et al. (2021); Ahmad et al. (2021)). For methods that do not use annotations (Huang et al. (2018); Mehta et al. (2022)), a common approach is to learn from text that contains seed words to identify event classes (Riloff and Jones (1999); Yangarber et al. (2000); Yu et al. (2022)). Zhang et al. (2021) present a zero-shot EE method that generates cluster embedding representations for event classes from sentences that contain synonyms of event names, and computes cosine distance from event clusters to classify instances. Zhang et al. (2022a) refine this, addressing ambiguity by using event definitions to form these event cluster representations.

The ambiguity challenge is also well-known in non-zero-shot EE, where Liao and Grishman (2010a) and Liao and Grishman (2011) design methods to "limit the effect of ambiguous patterns", Ji and Grishman (2008) try to learn "correct" word senses during training, and Liao and Grishman (2010b) use information about other event classes to resolve ambiguities in given instances.

9 Conclusion

We propose a multi-stage, fine-grained, generative LM pipeline for dyadic zero-shot EE that addresses ambiguity and efficiency challenges, controls for modality, allows for more interpretability, and outperforms other zero-shot EE methods. Our pipeline benefits from an MC approach, which future work could explore and apply on other tasks (e.g., involving lexical resource generation). As zero-shot, the pipeline is deployable in various real-world settings and we demonstrated it on a case study in international relations, an important application.

References

Agichtein, E. and Gravano, L. (2000). Snowball: Extracting relations from large plain-text collections. In *Proceedings of the Fifth ACM Conference on Digital Libraries*, DL '00, page 85–94, New York, NY, USA. Association for Computing Machinery.

Ahmad, W., Peng, N., and Chang, K.-W. (2021). Gate: Graph attention transformer encoder for cross-lingual relation and event extraction. In *The Thirty-Fifth AAAI Conference on Artificial Intelligence (AAAI-21)*.

Ahn, D. (2006). The stages of event extraction. In *Proceedings of the Workshop on Annotating and Reasoning about Time and Events*, pages 1–8, Sydney, Australia. Association for Computational Linguistics.

Beieler, J. (2016). Generating politically-relevant event data. In *Proceedings of the First Workshop on NLP and Computational Social Science*, pages 37–42, Austin, Texas. Association for Computational Linguistics.

Boschee, E., Natarajan, P., and Weischedel, R. (2013). *Automatic Extraction of Events from Open Source Text for Predictive Forecasting*, pages 51–67. SpringerLink.

Brandt, P. T., Freeman, J. R., and Schrodt, P. A. (2011). Real time, time series forecasting of inter- and intra-state political conflict. *Conflict Management and Peace Science*, 28(1):41–64.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. (2020). Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Bunescu, R. (2008). Learning with probabilistic features for improved pipeline models. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 670–679, Honolulu, Hawaii. Association for Computational Linguistics.

Cai, E. and O'Connor, B. (2023). Evaluating zero-shot event structures: Recommendations for automatic content extraction (ACE) annotations. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 1651–1665, Toronto, Canada. Association for Computational Linguistics.

Califf, M. E. and Mooney, R. J. (2003). Bottom-up relational learning of pattern matching rules for information extraction. *J. Mach. Learn. Res.*, 4(null):177–210.

Cao, Q., Trivedi, H., Balasubramanian, A., and Balasubramanian, N. (2020). DeFormer: Decomposing pre-trained transformers for faster question answering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4487–4497, Online. Association for Computational Linguistics.

Chen, Y., Chen, T., Ebner, S., White, A. S., and Van Durme, B. (2020). Reading the manual: Event extraction as definition comprehension. In *Proceedings of the Fourth Workshop on Structured Prediction for NLP*, pages 74–83, Online. Association for Computational Linguistics.

Chen, Y., Liu, S., Zhang, X., Liu, K., and Zhao, J. (2017). Automatically labeled data generation for large scale event extraction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 409–419, Vancouver, Canada. Association for Computational Linguistics.

Cui, M., Li, L., Wang, Z., and You, M. (2017). A survey on relation extraction. In *China Conference on Knowledge Graph and Semantic Computing*.

Dai, Y., Radford, B., and Halterman, A. (2022). Political event coding as text-to-text sequence generation. In *Proceedings of the 5th Workshop on Challenges and Applications of Automated Extraction of Socio-political Events from Text (CASE)*, pages 117–123, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

Doddington, G., Mitchell, A., Przybocki, M., Ramshaw, L., Strassel, S., and Weischedel, R. (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).

Dowty, D. (1991). Thematic proto-roles and argument selection. *Language*, 67(3):547–619.

Du, X. and Cardie, C. (2020). Event extraction by answering (almost) natural questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 671–683, Online. Association for Computational Linguistics.

Finkel, J. R., Manning, C. D., and Ng, A. Y. (2006). Solving the problem of cascading errors: Approximate Bayesian inference for linguistic annotation pipelines. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 618–626, Sydney, Australia. Association for Computational Linguistics.

Galtung, J. (1971). A structural theory of imperialism. *Journal of Peace Research*, 8(2):81–117.

Gao, J., Zhao, H., Yu, C., and Xu, R. (2023). Exploring the feasibility of chatgpt for event extraction. *ArXiv*, abs/2303.03836.

Gao, L., Wu, J., Qiao, Z., Zhou, C., Yang, H., and Hu, Y. (2016). Collaborative social group influence for event recommendation. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, CIKM '16, page 1941–1944, New York, NY, USA. Association for Computing Machinery.

Gerner, D., Jabr, R., and Schrodt, P. (2002). Conflict and Mediation Event Observations (CAMEO): A new event data framework for the analysis of foreign policy interactions. *International Conflict Mediation*.

Gerner, D. J., Schrodt, P. A., Francisco, R. A., and Weddle, J. L. (1994). Machine coding of event data using regional and international sources. *International Studies Quarterly*, 38(1):91–119.

Gildea, D. and Jurafsky, D. (2002). Automatic labeling of semantic roles. *Computational Linguistics*, 28(3):245–288.

Gupta, P. and Ji, H. (2009). Predicting unknown time arguments based on cross-event propagation. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 369–372, Suntec, Singapore. Association for Computational Linguistics.

Halterman, A. (2020). Extracting political events from text using syntax and semantics. Working paper.

Hong, Y., Zhang, J., Ma, B., Yao, J., Zhou, G., and Zhu, Q. (2011). Using cross-entity inference to improve event extraction. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1127–1136, Portland, Oregon, USA. Association for Computational Linguistics.

Huang, L., Ji, H., Cho, K., Dagan, I., Riedel, S., and Voss, C. (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.

Huang, L., May, J., Pan, X., Ji, H., Ren, X., Han, J., Zhao, L., and Hendler, J. (2017). Liberal entity extraction: Rapid construction of fine-grained entity typing systems. *Big Data*, 5:19–31.

Hürriyetoğlu, A., editor (2021). *Proceedings of the 4th Workshop on Challenges and Applications of Automated Extraction of Socio-political Events from Text (CASE 2021)*, Online. Association for Computational Linguistics.

Ji, H. and Grishman, R. (2008). Refining event extraction through cross-document inference. In *Proceedings of ACL-08: HLT*, pages 254–262, Columbus, Ohio. Association for Computational Linguistics.

Levy, O., Seo, M., Choi, E., and Zettlemoyer, L. (2017). Zero-shot relation extraction via reading comprehension. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 333–342, Vancouver, Canada. Association for Computational Linguistics.

Li, F., Peng, W., Chen, Y., Wang, Q., Pan, L., Lyu, Y., and Zhu, Y. (2020a). Event extraction as multi-turn question answering. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 829–838, Online. Association for Computational Linguistics.

Li, M., Zareian, A., Lin, Y., Pan, X., Whitehead, S., Chen, B., Wu, B., Ji, H., Chang, S.-F., Voss, C., Napierski, D., and Freedman, M. (2020b). GAIA: A fine-grained multimedia knowledge extraction system. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 77–86, Online. Association for Computational Linguistics.

Li, Q., Ji, H., and Huang, L. (2013). Joint event extraction via structured prediction with global features. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 73–82, Sofia, Bulgaria. Association for Computational Linguistics.

Li, Q., Peng, H., Li, J., Hei, Y., Sun, R., Sheng, J., Guo, S., Wang, L., and Yu, P. S. (2022). A survey on deep learning event extraction: approaches and applications. *IEEE Transactions on Neural Networks and Learning Systems*.

Li, S., Ji, H., and Han, J. (2021). Document-level event argument extraction by conditional generation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 894–908, Online. Association for Computational Linguistics.

Liao, S. and Grishman, R. (2010a). Filtered ranking for bootstrapping in event extraction. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 680–688, Beijing, China. Coling 2010 Organizing Committee.

Liao, S. and Grishman, R. (2010b). Using document level cross-event inference to improve event extraction. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 789–797, Uppsala, Sweden. Association for Computational Linguistics.

Liao, S. and Grishman, R. (2011). Acquiring topic features to improve event extraction: in pre-selected and balanced collections. In *Proceedings of the International Conference Recent Advances in Natural Language Processing 2011*, pages 9–16, Hissar, Bulgaria. Association for Computational Linguistics.

Lin, Y., Ji, H., Huang, F., and Wu, L. (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.

Liu, J., Chen, Y., Liu, K., Bi, W., and Liu, X. (2020). Event extraction as machine reading comprehension. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1641–1651, Online. Association for Computational Linguistics.

Liu, S., Chen, Y., He, S., Liu, K., and Zhao, J. (2016). Leveraging FrameNet to improve automatic event detection. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2134–2143, Berlin, Germany. Association for Computational Linguistics.

Liu, S., Chen, Y., Liu, K., and Zhao, J. (2017). Exploiting argument information to improve event detection via supervised attention mechanisms. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1789–1798, Vancouver, Canada. Association for Computational Linguistics.

Lyu, Q., Zhang, H., Sulem, E., and Roth, D. (2021). Zero-shot event extraction via transfer learning: Challenges and insights. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 322–332, Online. Association for Computational Linguistics.

Manning, C., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S., and McClosky, D. (2014). The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60, Baltimore, Maryland. Association for Computational Linguistics.

McClosky, D., Surdeanu, M., and Manning, C. (2011). Event extraction as dependency parsing. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1626–1635, Portland, Oregon, USA. Association for Computational Linguistics.

Mehta, S., Rangwala, H., and Ramakrishnan, N. (2022). Improving zero-shot event extraction via sentence simplification. In *Proceedings of the 5th Workshop on Challenges and Applications of Automated Extraction of Socio-political Events from Text (CASE)*, pages 32–43, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

Miller, G. A. (1995). Wordnet: A lexical database for english. *Commun. ACM*, 38(11):39–41.

Nguyen, T. H., Cho, K., and Grishman, R. (2016). Joint event extraction via recurrent neural networks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 300–309, San Diego, California. Association for Computational Linguistics.

Nguyen, T. H. and Grishman, R. (2015). Event detection and domain adaptation with convolutional neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 365–371, Beijing, China. Association for Computational Linguistics.

Nguyen, T. M. and Nguyen, T. H. (2019). One for all: Neural joint modeling of entities and events. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence and Thirty-First Innovative Applications of Artificial Intelligence Conference and Ninth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI'19/IAAI'19/EAAI'19. AAAI Press.

Nik, A., Zhang, G., Chen, X., Li, M., and Fu, J. (2022). 1Cademy @ causal news corpus 2022: Leveraging self-training in causality classification of socio-political event data. In *Proceedings of the 5th Workshop on Challenges and Applications of Automated Extraction of Socio-political Events from Text (CASE)*, pages 91–99, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

Norris, C., Schrodt, P., and Beieler, J. (2017). Petrarch2: Another event coding program. *Journal of Open Source Software*, 2(9):133.

O'Brien, S. P. (2010). Crisis Early Warning and Decision Support: Contemporary Approaches and Thoughts on Future Research. *International Studies Review*, 12(1):87–104.

O'Connor, B., Stewart, B. M., and Smith, N. A. (2013). Learning to extract international relations from political context. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1094–1104, Sofia, Bulgaria. Association for Computational Linguistics.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., et al. (2022). Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744.

Peng, N., Ferraro, F., Yu, M., Andrews, N., DeYoung, J., Thomas, M., Gormley, M. R., Wolfe, T., Harman, C., Van Durme, B., and Dredze, M. (2015). A concrete Chinese NLP pipeline. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, pages 86–90, Denver, Colorado. Association for Computational Linguistics.

Pennington, J., Socher, R., and Manning, C. (2014). GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Piantadosi, S. (2023). Modern language models refute Chomsky's approach to language. *lingbuzz/007180*.

Porter, M. F. (2001). Snowball: A language for stemming algorithms. Published online. Accessed 11.03.2008, 15.00h.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Riloff, E. (1993). Automatically constructing a dictionary for information extraction tasks. In *Proceedings of the Eleventh National Conference on Artificial Intelligence*, AAAI'93, page 811–816. AAAI Press.

Riloff, E. and Jones, R. (1999). Learning dictionaries for information extraction by multi-level bootstrapping. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence and the Eleventh Innovative Applications of Artificial Intelligence Conference Innovative Applications of Artificial Intelligence*, AAAI '99/IAAI '99, page 474–479, USA. American Association for Artificial Intelligence.

Rogers, A., Kovaleva, O., and Rumshisky, A. (2020). A primer in BERTology: What we know about how BERT works. *Transactions of the Association for Computational Linguistics*, 8:842–866.

Sandhaus, E. (2008). The New York Times Annotated Corpus. *Linguistic Data Consortium*, LDC2008T19.

Schrodt, P. (2015). CountryInfo.txt: Country names, codes, places and leaders.

Schrodt, P. and Gerner, D. (2004). An event data analysis of third-party mediation. *Journal of Conflict Resolution*, 48:310–330.

Schrodt, P. A. and Gerner, D. J. (1994). Validity assessment of a machine-coded event data set for the middle east, 1982-92. *American Journal of Political Science*, 38:825.

Sha, L., Liu, J., Lin, C.-Y., Li, S., Chang, B., and Sui, Z. (2016). RBPB: Regularization-based pattern balancing method for event extraction. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1224–1234, Berlin, Germany. Association for Computational Linguistics.

Smith, S. T., Kao, E. K., Mackin, E., Shah, D. C., Simek, O., and Rubin, D. B. (2020). Automatic detection of influential actors in disinformation networks. *Proceedings of the National Academy of Sciences*, 118.

Stoehr, N., Hennigen, L. T., Valvoda, J., West, R., Cotterell, R., and Schein, A. (2023). An ordinal latent variable model of conflict intensity. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*. Forthcoming.

Stoehr, N., Torroba Hennigen, L., Ahbab, S., West, R., and Cotterell, R. (2021). Classifying dyads for militarized conflict analysis. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7775–7784, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Toutanova, K., Haghighi, A., and Manning, C. (2005). Joint learning improves semantic role labeling. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 589–596, Ann Arbor, Michigan. Association for Computational Linguistics.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., Rodriguez, A., Joulin, A., Grave, E., and Lample, G. (2023). Llama: Open and efficient foundation language models.

Wadden, D., Wennberg, U., Luan, Y., and Hajishirzi, H. (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.

Wang, X., Wei, J., Schuurmans, D., Le, Q., Chi, E., Narang, S., Chowdhery, A., and Zhou, D. (2023). Self-consistency improves chain of thought reasoning in language models.

Wasserman, S. and Faust, K. (1994). *Social Network Analysis: Methods and Applications*. Cambridge University Press, 1 edition.

Watanabe, Y., Iwatate, M., Asahara, M., and Matsumoto, Y. (2008). A pipeline approach for syntactic and semantic dependency parsing. In *CoNLL 2008: Proceedings of the Twelfth Conference on Computational Natural Language Learning*, pages 228–232, Manchester, England. Coling 2008 Organizing Committee.

Williams, A. (2015). *Agent and Patient*, page 141–162. Key Topics in Syntax. Cambridge University Press.

Yangarber, R., Grishman, R., Tapanainen, P., and Huttunen, S. (2000). Automatic acquisition of domain knowledge for information extraction. In *Proceedings of the 18th Conference on Computational Linguistics - Volume 2*, COLING '00, page 940–946, USA. Association for Computational Linguistics.

Yates, A., Cafarella, M., Banko, M., Etzioni, O., Broadhead, M., and Soderland, S. (2007). Textrunner: Open information extraction on the web. In *Proceedings of Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, NAACL-Demonstrations '07, page 25–26, USA. Association for Computational Linguistics.

Yin, W., Hay, J., and Roth, D. (2019). Benchmarking zero-shot text classification: Datasets, evaluation and entailment approach. 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 3914–3923, Hong Kong, China. Association for Computational Linguistics.

You, H., Samuel, D., Touileb, S., and Øvrelid, L. (2022). EventGraph: Event extraction as semantic graph parsing. In *Proceedings of the 5th Workshop on Challenges and Applications of Automated Extraction of Socio-political Events from Text (CASE)*, pages 7–15, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

Yu, P., Zhang, Z., Voss, C., May, J., and Ji, H. (2022). Building an event extractor with only a few examples. In *Proceedings of the Third Workshop on Deep Learning for Low-Resource Natural Language Processing*, pages 102–109, Hybrid. Association for Computational Linguistics.

Yuan, C., Xie, Q., and Ananiadou, S. (2023). Zero-shot temporal relation extraction with chatgpt.

Zhang, H., Wang, H., and Roth, D. (2021). 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.

Zhang, H., Yao, W., and Yu, D. (2022a). Efficient zero-shot event extraction with context-definition alignment. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 7169–7179, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Zhang, S., Ji, T., Ji, W., and Wang, X. (2022b). Zero-shot event detection based on ordered contrastive learning and prompt-based prediction. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 2572–2580, Seattle, United States. Association for Computational Linguistics.

Zhang, T., Ji, H., and Sil, A. (2019). Joint Entity and Event Extraction with Generative Adversarial Imitation Learning. *Data Intelligence*, 1(2):99–120.

10 Supplementary Material

10.1 Risks

Large generative models may produce harmful word sequences, but our fine-grained pipeline avoids tasks that are more likely to produce open-ended answers, focusing on tasks that produce text with constrained output (e.g. boolean QA, extractive QA, and synonym set generation). The one task that could face more bias from an LLM involves extracting a country mention from text in the affiliation detection extension, where a generative LM could assert what counts and does not count as a country, which could be controversial (e.g. Palestine). However, we do not use a generative model for this step because we observed more errors in addition to this potential bias.

One important application of our approach, which is extracting sociopolitical events, may be of great interest to social scientists (e.g. CASE workshop; (Nik et al. (2022); Dai et al. (2022); You et al. (2022))) as well as having government and military intelligence utility (see ethical discussion of (Li et al. (2020b)) on dual use issues for their multimodal tracking and surveillance system).

10.2 Models

For the results in Table 2, we use text-davinci-003, but gpt-3.5-turboperforms similarly and also outperforms the other zero-shot EE methods. For the demonstration of our method with the affiliation detection extension, we use gpt-3.5-turbo to save cost.

We also try implementing our approach with open-source models which are useful for keeping data local and for having a reliable tool to implement the pipeline with. Further, some open-source models are more replicable; for example, Llama and Alpaca will consistently output the same response when the temperature is 0. We try Llama (1 and 2) 7b and 13b, and Alpaca 7b (instruction-tuned version of Llama 1). Although Llama 2 is able to follow instructions (where Llama 1 struggles) and answer tricker questions such as: *...it hurt their chances.\n In the sentence, does 'hurt' indicate 'injure'?*

(where Alpaca struggles), all inconsistently struggle with producing outputs of a specified format and with generating diverse synonym sets that are consistent with a word's definition. For example, Llama 2 outputs *help, assist, aid*, and *support* as synonyms of *injure*. Yet, Alpaca and Llama 2 serve as positive steps toward improving open-source models.

For the future, we foresee that not only will generative GPT-family LMs improve their performance on the fine-grained tasks in our pipeline, which will contribute to further performance gains, but also that higher performance open-source models will be available for use, which may help to increase replicability of results and eliminate cost issues for the using GPT-family models. We encourage future work to explore more open-source LMs for the pipeline.

10.3 Limitations

Our pipeline introduces a perspective of performing EE in fine-grained tasks and for implementation, we make choices about certain models, hyperparameters, and prompts.

Models. The Table 2 pipeline results use text-davinci-003, and we find that the pipeline using gpt-3.5-turbo also outperforms the zero-shot baselines. Further, these models outperform open-source generative models on specific tasks (§6 and §10.2), despite limitations of being proprietary and non-deterministic (our pipeline addresses model non-determinism in §5). While we tried open-source models including Alpaca[10] and Llama (1 and 2) 7B and 13B, Alpaca produced unreliably formatted outputs, and Llama 2 struggles to construct diverse synonym sets that are relevant to a word (e.g. *help, aid* as synonyms of *injure*). However, such models may potentially be valuable for the future — improving zero-shot performance of open-source models is an active research area.

Hyperparameters. The temperature hyperparameter is broadly an issue in many generative approaches, and we consider its effect on various factors in §5.

Prompts. We are very concerned about prompt sensitivity which we discuss in §5; at a high level, our approach has well-defined subtasks where small prompt changes do not impact results much (in contrast to a black-box neural net approach).

MC helps to weaken variability in our approach. For example, our final synonym selection prompt was: *List synonyms of [text] in bullet points.* Yet, alternative phrasings exist, such as: *Output synonyms of [text] in bullet points.*, and *What are the synonyms of [text]? Answer in bullet points.*, and *List synonyms of [text] as a numbered list.* While these alternatives may construct a slightly different synonym set after a single sample, the variance among the synonym sets after 70 draws on any of these prompts at a given temperature is very similar to that of the synonym sets after 70 MC draws on our final prompt. We chose the final prompt out of convenience; it has fewer tokens and bullet point outputs are a simple format to parse.

We selected the first prompt that we tried for the rest of the tasks using the validation set discussed in §6. While future study could explore different phrasings, our decision process suggests that with explicit and specific linguistic task definitions for word sense disambiguation, event argument extraction, and other tasks, prompt engineering may not be necessary for producing a higher accuracy output. We also find that minor changes to a prompt such as using "sentence" versus "text" and reordering the prompt does not affect the output.

The one task that we did not use LLMs for involves identifying country references in a sentence, which we found is vulnerable to bias (§10.1). We also found many errors in the output when trying LLMs and therefore opted for using a dictionary.

10.4 Naive Event Detection Details

For the naive generative LM event detection experiments in §3, we present more details about the experiments, including examples for prompts and further approaches that we tried.

Example of hypotheses. An example of the four hypotheses described in Table 1, where definitions come from or have small modifications of descriptions in the ACE annotation guidelines, is: We also try other types of hypotheses, such as using root words of event class names instead of a more natural form of the event class name with an affix of -ing.

[10] `https://github.com/replicate/cog_stanford_alpaca`

Hypotheses
1 This text is about pardoning.
2 This text discusses pardoning.
3 This text is about pardoning, where 'pardon' is to lift a sentence imposed by the judiciary.
4 This text discusses pardoning, where 'pardon' is to lift a sentence imposed by the judiciary.

Table 3: Example of hypotheses used for Table 1.

Hypotheses
This text is about 'pardon'.
This text discusses 'pardon'.
This text is about 'pardon', where 'pardon' is to lift a sentence imposed by the judiciary.
This text discusses 'pardon', where 'pardon' is to lift a sentence imposed by the judiciary.

Table 4: Example of alternative hypotheses.

Converting hypotheses to boolean queries. To convert hypotheses into boolean queries, for the hypotheses that include 'about', the queries begins with:

'*Is the text about...*'

For hypotheses that include 'discuss', the query begins with:

'*Does this text discuss...*'

Model details. We experimented with the boolean queries when using text-davinci-003, gpt-3.5-turbo, Alpaca, and the roberta model fine-tuned on the BoolQA dataset. However, we found very poor performance using the roberta model fine-tuned on BoolQA, supporting Lyu et al. (2021)'s finding.

For the text entailment experiments, we selected results using the probability threshold of 0.45 for deberta and of 0.55 for roberta since those thresholds tended to yield higher F1 performance scores over different variations of prompts.

Further alternative prompt strategies and models. We also tried experiments on other hypothesis and prompt variations and tried other models. The experiments on deberta-xlarge and bart-large models yielded similar performances as those in the table. The prompt "*Someone was pardoned*" referred to in Lyu et al. (2021) also did not produce any significantly different performance result. For definitions, we tried placing them before and after the "*This text is about...*" hypothesis or its query counterpart in generative LMs. Further, we experimented with longer and shorter definitions, finding that the performance of these variations was similar to the performance of the variations in Table 1.

10.5 Event Detection and Argument Extraction Details

Our event detection step allows for any single-word or multi-word event name n_t. However, while event detection works for any single- or multi-word event name, argument extraction requires the event class to have the potential to contain dyadic actor arguments – for example, INJURE could have dyadic actor arguments because an agent actor a_1 could cause the INJURE action and a patient actor a_2 could be the receiver of the INJURE action. On the other hand, the verb event class STAND may not be able to have dyadic actor arguments because an agent actor a_1 could instigate the instance, but no patient actor a_2 exists. Our pipeline assumes that each event class in the input could possibly have dyadic actor arguments.

The queries in argument extraction use the verb form of event name n_t. If $n_{verb,t}$ is a regular verb, the queries for extracting dyadic actors a_1, a_2 are straightforward as *Who [$n_{verb,t}$]s?* and *Who is [$n_{verb,t}$]ed?* (e.g. *Who injures?* and *Who is injured?*). When $n_{verb,t}$ is more complicated, containing a preposition as in PROTEST AGAINST, argument extraction asks questions in the form of *Who [$n_{verb,t}$]s?* and *Who is [$n_{verb,t}$]ed [preposition]?*. For example, the query for PROTEST AGAINST is *Who protests?* and *Who is protested against?*. From an error analysis of our experiments, we find that

Temp	0	1	2	3	4	5
0	.9986	.0005	.0004	.0002	.0001	.0001
0.33	.9938	.0021	.0015	.0013	.0010	.0003
0.67	.9887	.0043	.0031	.0017	.0011	.0010

Table 5: The proportion of boolean answers that are different from the rest in 10 samples over different temperatures.

	0	0.33	0.67	1
Pure Output	.9538	.8109	.6555	.5336
Aggregate substrings	.9748	.9034	.7941	.7311

Table 6: The proportion of extractive answers that are unanimous over 10 samples over different temperatures. We additionally clustered answers that are substrings of each other.

our system achieves the highest performance on event classes that are single regular verbs, but could still correctly detect and extract arguments for multi-word event classes.

Extra Candidate Triggers. From performing experiments to vary the size of each candidate trigger set and observing how many queries our method performs given a particular K_t (as in the recall vs compute cost plot (Fig 7) of Sec 6), we found that including extra and irrelevant candidate triggers in the candidate trigger set leads to higher compute cost, but performance does not change much.

Not susceptible to grammar changes or errors. While our pipeline aims to construct questions that are grammatically correct, even if it makes a mistake in the affix of a word or in using the correct article, we find that generative models are able to answer questions correctly. This is different from text entailment models where performance probabilities may be more sensitive to small grammatical errors, depending on the model.

Fewer alternatives for prompt wordings. Given the specific and fine-grained tasks in our pipeline, we find that fewer alternatives for prompt wordings exist. Adding to the discussion on synonym set generation and the disambiguation step in §10.3, for extractive QA, a "who" question with an event name follows previous literature, and there are not many simple alternative questions for extracting the desired arguments.

10.6 Monte Carlo Details

Tables 5 and 6 contain the results discussed in Section 6.

gpt-3.5-turbo results are similar, but show slightly more variability.

We also tried this procedure on Alpaca and Llama 2. Although Alpaca is able to output the same response at temperature 0 over many prompt executions unlike GPT-family models, when the temperature increases, in addition to outputting different synonym sets, it outputs the synonym sets in a different format for each execution, where the format is not easily parsable into a mathematical set representation that fits our needs. We observe similar behavior with Llama 2, where after increasing temperature, the output becomes very random and may not include English words (random letters are sometimes capitalized). Therefore, we use GPT-family models for the task but hope for future work to explore using open-source generative models, which are rapidly improving.

10.7 ACE Evaluation Details

In §6, we discussed the ACE dataset that we used to evaluate event detection and argument extraction of our method. In addition to using it to evaluate EE for its popularity, we use it because it has a variety of advantages over other datasets including whole-document annotations, realistically non-lexical-specific event classes, event modality, English data, specification of event arguments, and discourse-level entities (Cai and O'Connor (2023)).

Details about input for the ACE evaluation. Our event detection evaluation was over all 33 event subclasses in ACE, similar to most other evaluations. For n_t, we used each of the 33 subclass names. However, if a subclass in ACE is actually described as two subclasses (e.g. `arrest-jail`) with separate definitions for each, we consider the events separately (e.g. `arrest` and `jail`), aggregating their counts during evaluation. For definitions d_t, we used short one-sentence descriptions or

paraphrases in the ACE documentation; for $k_t \in K_t$, we only add keywords if the documentation emphasizes certain words as being associated with an event class, such as 'gunfire' for 'attack'.

For dyadic argument extraction, argument role pairs in ACE that map to *agent* and *patient* actors in §2 include *{Agent, Victim}* (e.g. INJURE,DIE), *{Attacker,Target}* (e.g. ATTACK), *{Agent,Person}* (e.g. NOMINATE, ELECT, ARREST-JAIL, EXECUTE, EXTRADITE), *{Prosecutor,Defendant}* (e.g. TRIAL-HEARING, CHARGE-INDICT, APPEAL), *{Plaintiff,Defendant}* (e.g. SUE), and *{Adjudicator,Defendant}* (e.g. CONVICT, SENTENCE, ACQUIT, PARDON). We map paired argument roles of 20 event classes in ACE to our *agent* and *patient* construction.

Terminology. We also note a slight difference in terminology for reporting performance on ACE from some other EE evaluations: while some evaluations report trigger identification and trigger classification as separate tasks with unique F1 performance scores, our method performs the two tasks simultaneously and we refer to their combination as simply *event detection*, reporting a single F1 performance score for them. Similarly, some evaluations report argument identification and argument classification separately with unique F1 scores, but our method performs the two tasks concurrently and we refer to their combination as *argument extraction* in Table 2 of §6.

10.8 Affiliation Detection Details

Rebel groups. For the affiliation detection case study (§7), we identify country references and rebel groups in each sentence. To identify rebel groups, we search the actor span to find "rebel..." or "insurgen..."; next, if our approach identifies that such an actor is affiliated with a country, we instead refer to that actor as being affiliated with the *rebel group* for that country instead of the country itself.

Identifying country references. The step of the affiliation detection extension that identifies country references is the only step in our pipeline that does not use generative language models. We make this choice because much disagreement exists about what counts as a country, we find that different prompts could produce slightly different outputs, and we find many errors of false positives and negatives of conventionally agreed upon countries. Instead of using generative language models, we use a dictionary from TABARI. Since NYT typically also refers to the US using presidents, government positions, and government bodies, we additionally include post-Reagan US presidents, government positions including "Attorney General" and "Secretary General", and government bodies including "Congress" and "Administration" as potential country references for the United States only. If such references are incorrect, not referring to the United States, the subsequent affiliation detection step that queries if an argument is affiliated with a high-level entity tends to filter them out.

Mapping country references to country names. Since the dictionary from TABARI maps country references to 3-letter ISO3166 country codes (contra Correlates Of War [COW] codes[11]), we use another dictionary to map[12] the country codes to country names.

MC Details. Since the NYT dataset was larger, we chose to perform MC over 6 boolean queries instead of over 9 as we did for the ACE evaluation to save time and cost. In §5, we provided evidence that boolean outputs for temperature 0 were mostly unanimous, and 9 samples are probably not needed; even 3 would be reasonable.

[11] `https://correlatesofwar.org/cow-country-codes/`

[12] `https://github.com/janmarques/IsoCountries`