# DICE: Data-Efficient Clinical Event Extraction with Generative Models

**Mingyu Derek Ma*** **Alexander K. Taylor*** **Wei Wang** **Nanyun Peng**
Computer Science Department
University of California, Los Angeles
{ma, ataylor2, weiwang, violetpeng}@cs.ucla.edu

## Abstract

Event extraction for the clinical domain is an under-explored research area. The lack of training data along with the high volume of domain-specific terminologies with vague entity boundaries makes the task especially challenging. In this paper, we introduce DICE, a robust and data-efficient generative model for clinical event extraction. DICE frames event extraction as a conditional generation problem and introduces a contrastive learning objective to accurately decide the boundaries of biomedical mentions. DICE also trains an auxiliary mention identification task jointly with event extraction tasks to better identify entity mention boundaries, and further introduces special markers to incorporate identified entity mentions as trigger and argument candidates for their respective tasks. To benchmark clinical event extraction, we compose MACCROBAT-EE, the first clinical event extraction dataset with argument annotation, based on an existing clinical information extraction dataset, MACCROBAT (Caufield et al., 2019). Our experiments demonstrate state-of-the-art performances of DICE for clinical and news domain event extraction, especially under low data settings.

## 1 Introduction

Event extraction (EE) is an information extraction task that aims to identify event triggers and arguments from unstructured texts (Ahn, 2006). The EE task consists of two subtasks: 1) event detection, in which the model extracts trigger text and predicts the event type; and 2) event argument extraction, in which the model extracts argument text and predicts the role of each argument given an event trigger and associated event type.

Clinical EE aims to extract clinical events, which are occurrences at specific points in time during a clinical process, such as diagnostic procedures, symptoms, etc. The arguments for such events are entities that modify or describe properties of these events (Caufield et al., 2019). Figure 1 shows an example sentence with two clinical events. The overwhelming volume and details of clinical information necessitate clinical EE, which benefits many downstream tasks such as adverse medical event detection (Rochefort et al., 2015), drug discovery (Wang et al., 2009), clinical workflow optimization (Hsu et al., 2016), and automated clinical decision support (Yadav et al., 2013).

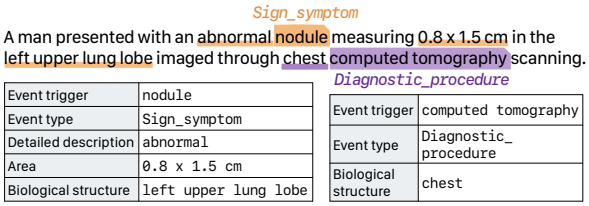



Figure 1: Illustration of a SIGN_SYMPTOM event triggered by "nodule" with multiple arguments including an AREA argument "0.8x1.5cm", and a DIAGNOSTIC_PROCEDURE event whose predicate is "computed tomography" described by argument "chest" of role BIOLOGICAL_STRUCTURE.

However, there are several non-trivial challenges of clinical EE compared to general domain EE. First, most triggers and arguments of clinical events consist of domain-specific terms that are more than 50% longer than the general domain on average, as shown in Table 1, and have vague boundaries because most clinical mentions[1] contain several descriptors. For instance, given the text span "massive heart attack", "heart attack" should be identified as the trigger (instead of "massive heart attack" or "attack") because it refers to a specific condition, and "massive" is an argument of the role type SEVERITY. However, when we consider "right common carotid artery", the entire text span describes a biological structure, and thus it functions as an argument of the role type BIOLOGICAL_STRUCTURE despite "right" and "common" being descriptors

*Equal contribution.

[1]Clinical mentions are defined as meaningful text spans of occurrences or their properties (Caufield et al., 2019).



*Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*
*Volume 1: Long Papers*, pages 15898–15917

for "carotid artery". The second challenge is the diversity and density of clinical arguments: there are on average 10 unique argument roles for each clinical event type compared to 3.7 in the general domain. Finally, it is challenging to obtain high-quality annotated data for clinical events due to both patient privacy concerns and the cost of expert annotations. Due to these challenges, there have been no clinical EE datasets with argument annotations to the best of our knowledge.

In this paper, we present DICE, a **D**ata-eff**I**cient generative model for **C**linical **E**vent extraction.[2] We build upon existing prompt-based generative event extraction models to formulate EE as a sequence-to-sequence text generation task (Hsu et al., 2022; Ma et al., 2023b). To handle the special challenges of clinical EE, DICE 1) introduces a mention identification-enhanced EE model, which specializes in clinical mention identification by performing contrastive learning to distinguish correct mentions from the ones with perturbed mention boundary, training an auxiliary mention identification module to learn implicit mention properties, and adding explicit mention markers to hint mention boundaries; 2) performs independent queries for each argument role to better handle long-tail argument roles.

To address the training data availability issue, we introduce MACCROBAT-EE, the first clinical event extraction dataset with argument information, which we derive from clinical experts' annotation on PubMed clinical case reports.

We benchmark DICE on MACCROBAT-EE against several recent event extraction models. Experiments show that DICE achieves state-of-the-art clinical event extraction results on MACCROBAT-EE, and we observe a larger performance gain under low-resource settings. Moreover, DICE also achieves better performance on the ACE05 dataset, demonstrating its generalizability to other domains.

Our contributions are threefold: 1) We develop DICE, a mention-enhanced clinical event extraction model that better identifies mention boundaries and is scalable to many argument roles; 2) We construct the first clinical event extraction dataset with argument annotations; 3) Our model achieves state-of-the-art performance on clinical and news EE and demonstrates more significant performance gains under low-resource settings.

[2]Please refer to https://derek.ma/DICE for code and data.

## 2 Related Works

### 2.1 General Domain Event Extraction

Many prior works formulate EE as token-level classification tasks and trained in an ED-EAE pipeline-style (Wadden et al., 2019; Yang et al., 2019; Ma et al., 2021b) or optimized jointly (Li et al., 2013; Yang and Mitchell, 2016; Lin et al., 2020; Nguyen et al., 2022a). Recent work formulates the EE task as text generation with transformer-based pre-trained language models that prompt the generative model to fill in synthetic (Paolini et al., 2021; Huang et al., 2021; Lu et al., 2021; Li et al., 2021) or natural language templates (Huang et al., 2022; Hsu et al., 2022; Ma et al., 2022; Ye et al., 2022). These generative EE models are not optimized to handle complicated domain-specific mentions. To our knowledge, there is no existing approach to clinical EE using a text generation formulation, which we hypothesize is due to both data unavailability and to the aforementioned domain challenges.

### 2.2 Event Extraction in Biomedical Domain

Biomedical EE is a type of biomedical IE tasks (Soysal et al., 2017; Fu et al., 2020; Xu et al., 2023). Existing approaches to biomedical EE (Huang et al., 2020; Trieu et al., 2020; Wadden et al., 2019; Ramponi et al., 2020; Wang et al., 2020) typically focus on extracting interactions or relationships between biological components such as proteins, genes, drugs, diseases and outcomes related to these interactions (Ananiadou et al., 2010). The mentions in these biological component interactions are short, distinctive biomedical terms and do not have rich event type-argument role ontologies because of the lack of interaction types present in the datasets (Ohta et al., 2011; Kim et al., 2011, 2013; Pyysalo et al., 2011, 2012). Li et al. (2020) develop a clinical event extraction model, but it only handles single-word events without considering arguments (Bethard et al., 2016). Our work addresses these concerns by introducing MACCROBAT-EE as well as providing a benchmark in a previously under-explored domain.

## 3 Clinical Domain Event Extraction

### 3.1 Task Formulation

We follow the framework of prior works that decomposes the EE task into Event Detection (ED) and Event Argument Extraction (EAE),

while introducing our novel Mention Identification module as an auxiliary task performed alongside both the ED and EAE modules. ED subtask takes a sentence (`passage`) as input to extract event triggers and predict event types. The trigger must be a sub-sequence of the `passage` and the event type must be one of the $n_{event\_type}$ pre-defined types. The EAE subtask takes a tuple of (`passage`, `event trigger`, `event type`), and extracts arguments from `passage` and predicts the argument role. Each event type holds a pool of $n_{arg\_role}^{event\_type}$ argument roles as defined in the event ontology.

### 3.2 The MACCROBAT-EE Dataset

Due to high annotation costs and privacy concerns, dataset availability is a primary bottleneck for clinical EE. We propose a repurposing of an existing expert-annotated dataset, MACCROBAT (Caufield et al., 2019),[3] to compose a clinical EE benchmark, MACCROBAT-EE.

The MACCROBAT dataset consists of 200 pairs of English clinical case reports from PubMed accompanying annotation files with partial event annotation provided by 6 annotators with prior experience in biomedical annotations. To our knowledge, this is the only openly accessible collection of clinical case reports annotated for entities and relations by human experts. Following existing sentence-level EE works (Lin et al., 2020), we construct an event extraction dataset with full event structure, MACCROBAT-EE, which contains annotated span information for *entities*, *event triggers*, *event types*, *event arguments* and *argument roles* for each sentence. Mentions are defined as meaningful text spans of occurrences and their properties (Caufield et al., 2019). We include all tagged mentions in MACCROBAT as *entities*, and further specify that mentions tagged as events and their respective types are included as *event triggers* and *event types*.

To infer event arguments and their roles, which are not provided in MACCROBAT, we consider non-event entities that hold a MODIFY relation with event triggers as arguments, and we use the assigned entity types as argument roles. We infer arguments via the MODIFY relation because its definition of an entity modifying an event matches well with the argument definition of further characterizing the properties of an event as shown in Appendix B.2. The entity type in MACCROBAT defines a type of fine-grained physical or procedure property, which matches the argument role definition of being a type of participant or attribute of an event. We traverse all (`event type`, `argument role`) pairs to obtain the argument roles possible for each event type to create an event ontology, as shown in Appendix B.3. The definitions of each event type and argument role written by clinical experts are provided.

[3] We use the 2020 version of MACCROBAT. We show more details about MACCROBAT in Appendix B.1.

### 3.3 Data Statistics

| Metric | ACE05 | ERE | MACCROBAT-EE |
|---|---|---|---|
| Unique event types | 33 | 38 | 13 |
| Unique argument roles | 22 | 21 | 22 |
| Unique arg. roles per event type | 4.73 | 2.87 | 10 |
| Documents # | 599 | 459 | 200 |
| Sentences # | 20,862 | 17,114 | 4,539 |
| Entities # | 54,820 | 46,185 | 23,898 |
| Trigger mentions # | 5,348 | 7,287 | 13,128 |
| Argument mentions # | 8,102 | 10,479 | 8,599 |
| Avg entities # per sentence | 3.18 | 3.20 | 5.43 |
| Avg events # per sentence | 1.34 | 1.47 | 3.21 |
| Avg args # per sentence | 2.39 | 2.24 | 2.67 |
| Avg args per event # | 1.48 | 1.42 | 0.81 |
| Avg entity word count | 1.12 | 1.10 | 1.89 |
| Avg trigger word count | 1.05 | 1.06 | 1.61 |
| Avg argument word count | 1.14 | 1.14 | 1.72 |

Table 1: Statistics of MACCROBAT-EE.

In Table 1, we show the statistics for MACCROBAT-EE as well as the comparable values for two widely-used EE datasets, ACE05 (Doddington et al., 2004) and ERE-EN (Song et al., 2015). MACCROBAT-EE differs from general-domain EE datasets because it contains fewer sentences and the average occurrences of entities, triggers, and arguments per sentence are significantly higher. Note that the average length of the entities in MACCROBAT-EE is significantly longer. Besides single-span entities, there are also nested and discontinuous entities used as event arguments in MACCROBAT-EE. This demonstrates that MACCROBAT-EE fills a different niche than ACE05 and ERE-EN and provides a valuable benchmark for EE under a clinical setting with high mention density, and allowing for future work to adapt clinical case report domain-specific features.

### 3.4 Human Verification

We conduct a human annotation to examine the coverage of the induced arguments and the correctness of their roles. Arguments and their roles in 96% out of 100 randomly sampled events are considered comprehensive and appropriate by both of the two annotators with consensus.

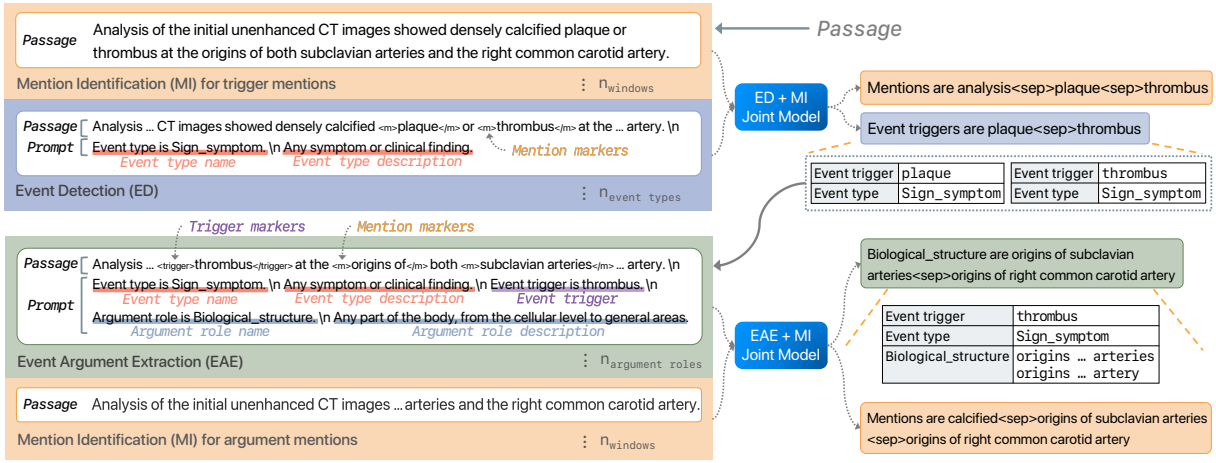



Figure 2: Model design of DICE. We use T5-large (Raffel et al., 2020) as the backbone text generation model for the two joint models. The ED module extracts event trigger and type, and the EAE module extracts argument and roles. They are trained jointly with the trigger and argument MI modules for mention-enhanced event extraction.

## 4 The DICE Event Extraction Model

We formulate EE as a conditional generation task, so that we can incorporate domain knowledge such as event type and argument role definitions via natural language in the input prompt. To tackle the challenges of clinical EE, we 1) further enhance the EE model's specialization in mention identification by techniques introduced in §4.2 to handle long clinical mentions with vague boundaries; and 2) perform an independent query for each event type/argument role for better long-tail performance in settings with many event types/argument roles as introduced in §4.1. Figure 2 shows the model design.

### 4.1 Seq2seq Components

There are three components: 1) Mention Identification (MI) which identifies the candidate pool of event triggers or event arguments, 2) Event Detection (ED) which extracts event triggers and predicts event types, and 3) Event Argument Extraction (EAE) which extracts arguments and predicts argument roles. We integrate these components to form the MI-ED-EAE pipeline (details in §4.3). We use pre-trained text generation model T5-large (Raffel et al., 2020) as the backbone LM. The input is a natural language sequence consisting of the original input *passage* and *prompt*. We design input-output formats with shared common elements across different tasks to enable synergistic joint optimization, as all three modules aim to generate a sub-sequence of the input passage.

**Mention Identification (MI).** To better align the MI task with the ED and EAE tasks, the MI module extracts all mentions that are candidate event triggers or arguments from the input passage. The input is the *passage* and the output includes all trigger or argument candidates in the input passage separated by a special token "[SEP]" following the prefix "Mentions are". If there are no mentions, a placeholder is generated (*i.e.* "Mentions are <mention>"). We extract mentions by inputting the entire passage as well as sentence segments selected by a sliding window with a size of a few words, which enables shorter outputs and higher mention coverage. We enforce the condition that the order of output mentions match the order of their appearance in the passage. This consistency helps the generative model to learn its expected behavior as well as allows for prior mention predictions to inform subsequent mention predictions. We keep the full passages in addition to the sliced sub-sequence during both training and inference to ensure the longer dependencies are captured.

**Event Detection (ED).** The ED module extracts event triggers from the passage. For a given passage, we construct $n_{event\_type}$ queries. For each query, we input the concatenation of *passage* and the following *prompt* segments: *event type name* and *event type description*. The output of the ED task is the concatenation of the event trigger texts predicted for the queried event type separated by a special token "[SEP]", following the prefix "Event triggers are". When there is no valid trigger for the queried event type (which are considered to be negative samples), a special placeholder is generated (*i.e.* "Event triggers are <trigger>"). The balance between positive and negative samples is a hyper-

parameter that may be tuned for a better precision-recall trade-off. We decode the output sequence and obtain a list of `(event type, trigger)` pairs.

**Event Argument Extraction (EAE).** The EAE module extracts event arguments from queries consisting of the input passage, a given role type, and a pair consisting of an event trigger and its event type. We perform $n_{arg\_roles}^{event\_type}$ queries to extract arguments corresponding to each potential argument role where $n_{arg\_roles}^{event\_type}$ is the number of unique argument roles for a certain event type. The input sequence contains *passage*, *event type name*, and *event type description* segments in addition to:

- *Trigger markers* which are special tokens (*i.e.* "<trigger>" and "</trigger>") to wrap trigger text to explicitly provide the trigger position
- *Trigger phrase* such as "Event trigger is plaque"
- *Argument role name* for the queried argument role, such as "Argument role is Severity"
- *Argument role description*

The expected output begins with a reiteration (Ma et al., 2023a) of the querying argument role (*e.g.* "Severity is") followed by the concatenated predicted argument texts or a placeholder ("<argument>") if there are no valid predictions.

### 4.2 Mention Identification Enhanced EE

We propose techniques to enhance the generative model's ability to accurately identify long mentions with vague boundaries: 1) contrastive learning with instances of perturbed mention boundaries, 2) explicit boundary hints with markers and 3) implicit joint mention representation learning.

**Contrastive learning with mention boundary perturbation.** Understanding the role of mention descriptors and distinguishing the subtle boundary difference are not specifically optimized during pre-training or fine-tuning with the text generation objective. We propose to create such a task and train the model specifically to recognize the mention with the correct boundaries from a pool of mentions with similar but shifted boundaries.

Following the seq2seq formulation introduced in §4.1, we construct $N$ input-output sequence pairs $\langle in_i, out_i \rangle$ where the input sequence $in_i$ consists of passage and prompt, and the gold output $out_i$ contains the ground-truth mentions, triggers or arguments for MI, ED or EAE respectively. For a certain input $in_i$, we consider the ground-truth output $out_i$ as a positive output (*e.g.* "Mentions are ... right common carotid artery"). We create the $k$ negative instances (*i.e.* $n_i^1, ..., n_i^k$) of $in_i$ by perturbing the left and right boundaries of mentions in $out_i$ to add/remove words (*e.g.* removing "right", removing "artery", or adding "the" before "right" etc.). We create the negative instances by perturbing output sequences without changing the input, and the contrastive learning objective applies to MI, ED and EAE. This results in a group of instances for $in_i$ including both positive and negative instances: $\mathbf{X}_i = \left\{ \langle out_i, in_i \rangle, \left\langle n_i^1, in_i \right\rangle, \ldots, \left\langle n_i^k, in_i \right\rangle \right\}$. Applying the process, we obtain instance groups for all input-output pairs $\mathbb{X} = \{\mathbf{X}_1, \ldots, \mathbf{X}_N\}$.

We use cross-entropy loss $\mathcal{L}_{CE}$ to learn to generate the correct output $out_i$ given input $in_i$. We introduce an InfoNCE loss (Oord et al., 2018) to learn to identify the positive output (items in the numerator) from a pool of output candidates with mention boundary perturbations (items in the denominator) (Ma et al., 2021a; Meng et al., 2021; Shen et al., 2020):

$$\mathcal{L}_N = \frac{1}{|\mathbb{X}|} \sum_{\mathbf{X}_i \in \mathbb{X}} \left[ \log \frac{f\left(out_i, in_i\right)}{\sum_{\left\langle n_i^j, in_i \right\rangle \in \mathbf{X}_i} f\left(n_i^j, in_i\right)} \right]$$

where $j \in [0, 1, 2, ..., k]$ and $n_i^0$ is the positive output $out_i$. We define the function $f\left(s, in_i\right)$ as the probability of generating a sequence $s$ given input $in_i$, which is estimated by multiplying logits for each token of the output produced by the decoder under the teacher-forcing paradigm while $in_i$ is fed to the encoder. This estimation is normalized by the output length and produces the output value of $f\left(s, in_i\right)$. We combine the two losses into the final objective $\mathcal{L}(\Theta) = \mathcal{L}_{CE} + \mathcal{L}_N$.

**Explicit mention marker.** Wrapping key spans with special token markers provides beneficial hints to the generative model that improve its understanding of how the components of the sentence are associated syntactically. We wrap trigger or argument mentions for the ED and EAE tasks, respectively, to provide a candidate pool for the identification task. To minimize the impact of error propagation of the imperfect MI module on downstream tasks, we consider two conditions: 1) the ED/EAE modules with markers must be robust enough to handle the compromised precision and incomplete coverage of the gold mentions and 2) the granularity of the candidate pool must not be too coarse or too fine. To address the first concern, we generate two versions of the data: one with mention

markers and one with no markers, and train the ED/EAE module over the augmented data. This trains the model to be robust in cases where the MI module provides imprecise predictions. The second concern stems from the too broad a candidate pool making the markers less informative and too strict a candidate pool making it difficult for the MI module to correctly identify mentions. To account for this issue, we use trigger mentions for the ED task and argument mentions for the EAE task as candidate pools as opposed to using words of a certain part-of-speech or named entities type. The unique properties of triggers (describing an entire process or behavior that can be linked to a specific time) and arguments (concrete details or descriptive content) make them more useful as candidate sets.

**MI as an implicit auxiliary task.** Existing works include a named-entity recognition task to provide additional supervision signals for EE (Zhao et al., 2019; Zhang et al., 2019; Sun et al., 2020; Wadden et al., 2019) for other formulations except for generative models. Since we design all three extraction tasks (ED, EAE and MI) as generation tasks, and ED and EAE can be considered as special MI with certain interest focus, identifying mentions is a synergistic capability contributing to performing ED and EAE. Thus, we add trigger MI and argument MI as auxiliary tasks to jointly optimize with the ED and EAE tasks, respectively.

### 4.3 Training and Inference

**Schedule sampling.** To gently bridge the discrepancy between gold and predicted upstream results (ED results passed to EAE, trigger/argument MI results passed to ED/EAE), we adopt the scheduled sampling technique to perform curriculum learning (Bengio et al., 2015). We force the downstream model to deal with imperfect upstream results gradually by decaying the upstream results from the gold ones to the predicted ones linearly. We perform the decay at the beginning of each epoch.

**Training.** We first train standalone trigger and argument MI modules to provide mention candidates. We then train ED+MI joint model and EAE+MI joint model with auxiliary trigger and argument MI modules respectively. We also add markers around trigger/argument mention candidates. For efficient training, the model uses downsampled negative instances (*i.e.* instances with mismatched trigger/argument and event type/argument role).

**Inference.** We use the trigger and argument mention markers produced by the standalone trigger and argument MI modules in the downstream ED+MI and EAE+MI joint models. The event triggers and their types predicted by the ED+MI joint model are provided as input to the EAE+MI joint model in a pipeline fashion.

## 5 Experiments in the Clinical Domain

We evaluate DICE on MACCROBAT-EE and compare it with existing event extraction models.

### 5.1 Experimental Setup

**Data splits.** We divide the 200 MACCROBAT-EE documents according to an 80%/10%/10% split for the training, validation, and testing sets, respectively. For low-resource settings, we consider 10%, 25%, 50%, and 75% of the number of *documents* used to build the training dataset while retaining the original validation and testing sets for evaluation.

**Evaluation metrics.** We follow previous EE works and report precision, recall and F1 scores for the following four tasks. 1) Trigger Identification: identified trigger span is correct. 2) Trigger Classification: identified trigger is correct and its predicted *event type* is correct. 3) Argument Identification: identified argument span is correct. 4) Argument Classification: identified event argument is correct and its predicted *argument role* is also correct.

**Variants.** We term two variants of our model. We refer to pipelined ED and EAE modules *without* the mention enhancement techniques described in §4.2, *with* long-tail argument handling and text generation cross-entropy loss only as **Vanilla DICE**, and the full model as **DICE**.[4]

**Baselines.** We benchmark the performance of the recent EE models on MACCROBAT-EE, including: **Text2Event** (Lu et al., 2021): a sequence-to-structure model that converts the input passage to a trie data structure to retrieve event arguments; **OneIE** (Li et al., 2013): a multi-task EE model trained with global features;[5] and **DEGREE** (Hsu et al., 2022): a prompt-based generative model that consists of distinct ED and EAE modules

[4]We show hyperparameters, implementation and baseline reproduction details in Appendix D.

[5]Note that additional entity annotation is used during training, while it is not used in other models.

| # | Model | Trigger | | | | | | Argument | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Identification | | | Classification | | | Identification | | | Classification | | |
| | | Prec. | Recall | F1 | Prec. | Recall | F1 | Prec. | Recall | F1 | Prec. | Recall | F1 |
| 1 | Text2Event | – | – | – | 66.64 | 60.57 | 63.46 | – | – | – | **55.29** | 47.89 | 51.33 |
| 2 | OneIE | **74.60** | 74.93 | 74.77 | **68.74** | 68.96 | 68.85 | 48.99 | 52.59 | 50.72 | 39.82 | 42.95 | 41.32 |
| 3 | DEGREE | 71.91 | 66.33 | 69.01 | 67.59 | 62.59 | 65.00 | 46.84 | 24.31 | 32.02 | 44.75 | 23.23 | 30.58 |
| 4 | Vanilla DICE | 65.03 | 74.08 | 69.26 | 60.51 | 70.28 | 65.03 | 49.10 | 53.60 | 51.25 | 45.95 | 50.76 | 48.24 |
| 5 | DICE | 73.53 | **76.98** | **75.22** | 68.12 | **72.97** | **70.46** | **55.41** | **57.87** | **56.61** | 53.02 | **55.03** | **54.01** |

Table 2: Event detection and event argument extraction performance (%). The EAE task takes the predicted event trigger and event type as input from the corresponding ED model in the pipeline style. DICE achieves the state-of-the-art event trigger and argument identification and classification performance.

| # | Mention-enhancing techniques | Trigger | | | | | | Argument | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Identification | | | Classification | | | Identification | | | Classification | | |
| | | Prec. | Recall | F1 | Prec. | Recall | F1 | Prec. | Recall | F1 | Prec. | Recall | F1 |
| 1 | Vanilla DICE | 65.03 | 74.08 | 69.26 | 60.51 | 70.28 | 65.03 | 70.76 | 76.48 | 73.51 | 66.47 | 72.71 | 69.45 |
| 2 | Vanilla w/ aux. task | 69.54 | 74.59 | 71.98 | 65.02 | 71.00 | 67.88 | 73.24 | 76.48 | 74.83 | 68.31 | 73.03 | 70.59 |
| 3 | Vanilla w/ marker | 72.91 | 70.71 | 71.79 | 68.58 | 67.70 | 68.14 | 74.27 | 76.91 | 75.57 | 69.66 | 72.82 | 71.20 |
| 4 | Vanilla w/ contrastive | 70.02 | 75.12 | 72.48 | 66.93 | 72.04 | 69.39 | 73.86 | 77.41 | 75.59 | 69.92 | 72.89 | 71.37 |
| 5 | Vanilla w/ all three (Full DICE) | 73.53 | 76.98 | **75.22** | 68.12 | 72.97 | **70.46** | 75.73 | 77.62 | **76.66** | 71.14 | 73.91 | **72.50** |
| 6 | Vanilla w/ perfect marker† | 97.04 | 94.11 | 95.55 | 85.23 | 88.66 | 86.91 | 91.91 | 90.72 | 91.31 | 81.71 | 86.73 | 84.14 |

Table 3: Ablation study of the technique used to incorporate mention information. The argument extraction reported here uses ground-truth event trigger and type, which removes error propagation from the upstream ED result. † indicates the settings use mention markers to wrap ground-truth mentions and they are not comparable with other lines.

that fill in event type-specific human written templates. To adapt DEGREE to the new dataset, we create the ED/EAE templates by concatenating event type/argument role phrases (*e.g.* "Biological_structure is artery").

## 5.2 Overall ED and EAE Results

We show the superiority of DICE in both high-resource and low-resource settings.

**High-resource results.** Table 2 shows the results for high-resource settings. Among the baselines, OneIE and Text2Event achieve the best F1 score on trigger extraction and argument extraction respectively. DEGREE reports low performance on the argument extraction task due to the challenges of generating long sequences containing all argument roles. DICE outperforms the baselines on *both* trigger and argument extraction tasks, with 2.7 points F1 score improvements for argument classification.

**Low-resource results.** We show the results of training in lower-resource settings in Figure 3 and Appendix C.3. We observe that DICE outperforms all baselines on all four tasks under all low-resource settings. The performance gap between DICE and the baselines increases in the lower training data percentage settings. In the argument classification task, DICE outperforms Text2Event by more than 8 (10%) and 9 (25%) points in F1 score.

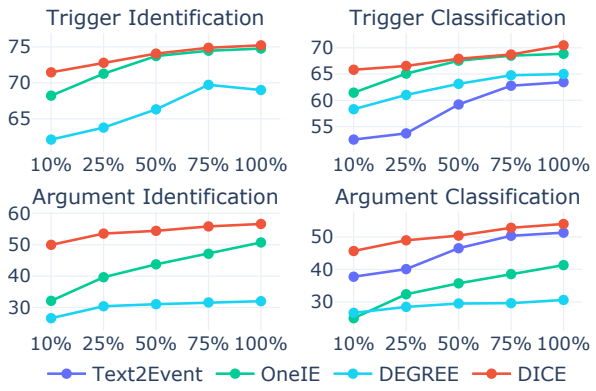



Figure 3: Performance on downsampled training data. We report F1 score (%, y-axis) for each proportion (x-axis). DICE outperforms all baselines across four tasks.

## 5.3 Ablation Studies

We show ablation studies about mention-enhancing techniques and MI module design in this section and more studies about input prompt segments and formulation in Appendix C.2.

**Mention-enhancing techniques.** We analyze the effects of the proposed mention-enhancing techniques in Table 3. We observe contrastive learning, auxiliary task, and mention markers contribute improvements of 1.92, 1.14 and 1.75 in the F1 score on argument classification, respectively. We observe that DICE improves over vanilla DICE by 5.43 and 3.05 in the F1 score for trigger and argument classification, respectively. We include an oracle setting on Line 6 that provides ground-truth mention markers during inference to illustrate the influence of the accuracy of the MI module.

| # | Model | Prec. | Recall | F1 |
|---|---|---|---|---|
| 1 | Yan et al. (2021) | 72.00 | 72.70 | 72.30 |
| 2 | OneIE entity identification module | **75.88** | 77.86 | 76.86 |
| 3 | DICE-MI without sliding window | 71.71 | 67.13 | 69.34 |
| 4 | DICE-MI without constrative learning | 71.80 | 84.14 | 77.48 |
| 5 | DICE-MI | 74.20 | **86.04** | **79.68** |

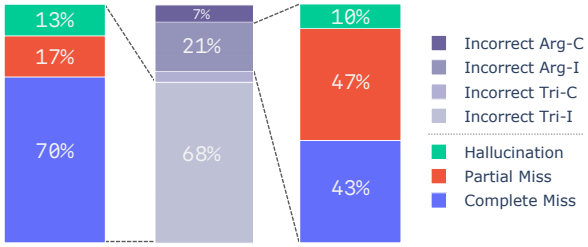
Table 4: Ablation study of MI module design.

**MI module design.** We compare our MI module with the representative of sequence tagging model OneIE, which produces BIO label for each input token, and state-of-the-art generative named-entity recognition model Yan et al. (2021), which generates token indexes, on the entity identification task. We report the performance in Table 4. The results show that the sliding window technique significantly improves recall (Line 5 vs 3) and contrastive learning improves overall performance (Line 5 vs 4). Our MI module outperforms all baselines and achieves the best F1 score.

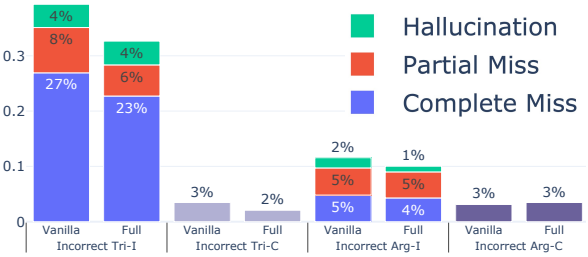
## 5.4 Error Analysis

We analyze the errors propagated through the 4 steps in the pipeline for DICE using predicted triggers on the argument classification task which shows the culmination of the errors propagated through the pipeline. The results in Figure 4a indicate that the identification sub-tasks, especially trigger identification, are the performance bottlenecks.

We further break identification errors into three types: 1) complete miss: the predicted span has no overlap with the ground-truth span; 2) partial miss: the predicted span is a subset of the ground-truth span; 3) hallucination: the predicted span partially overlaps with the ground-truth span, but also incorrectly includes additional tokens. As shown in Figure 4a, the majority of errors produced by the trigger identification step are complete misses,




(a) Proportion of error cases by steps in the pipeline.




(b) Error produced by each step of the vanilla and full version of DICE. We show *absolute* error ratios (wrong predictions among all predictions, the lower, the better).

Figure 4: Error analysis of the argument classification task, which shows the culmination of the errors propagated through the pipeline of DICE.

whereas argument identification suffers from both partial and complete misses. We also observe that the left boundaries of the trigger and argument spans are more difficult to identify as 76% of partial misses and 69% of hallucinations correctly identify the right boundary but miss the left boundary. This can be explained by that the dominant word of the entity is typically on the rightmost (*e.g.* "attack" in "heart attack"), whereas the left boundary requires separating the target entity from its descriptors (*e.g.* "massive heart attack").

We further compare the error types between the vanilla DICE and full version of DICE with mention identification enhancement techniques in Figure 4b. We observe that DICE produces fewer error cases across all error types in both trigger and argument identification steps, which supports our assertion that our mention identification enhancement techniques improve the identification of mentions with vague boundaries.

## 5.5 Qualitative Analysis

To identify challenges for future works, we summarize 4 types of common errors made by DICE and show examples in Table 5. In the first example, the MI module of DICE only identifies a subsequence of the true mention (e.g., "hearing loss" vs. "bilateral sensorineural high-frequency hearing loss"), leading to a partial miss that shows the

| | |
|---|---|
| 1 | **Task**: ED **Passage**: An {audiology evaluation} showed {severe} {bilateral} {sensorineural} {high-frequency} {hearing loss} ( {-70 dB} ).<br>**Ground-truth**: (bilateral sensorineural high-frequency hearing loss, Sign_symptom) **Pred. of DICE**: (hearing loss, Sign_symptom) |
| 2 | **Task**: EAE **Passage**: The patient underwent a {resection} of the { 15 cm segment IVb } mass [SIGN_SYMPTOM] in {June 2010} .<br>**Ground-truth**: (15 cm, Distance), (segment IVb, Biological_structure) **Pred. of DICE**: (15 cm segment IVb, Biological_structure) |
| 3 | **Task**: ED **Passage**: Core biopsies from the {breast lump} showed {ductal carcinoma} in situ (sample labelled P1.1).<br>**Ground-truth**: (biopsies, Diagnostic_procedure), (ductal carcinoma, Disease_disorder) **Pred. of DICE**: *None* |
| 4 | **Task**: EAE **Passage**: Serum total bilirubin and {tumor markers} , carcinoembryonic antigen [DIAGNOSTIC_PROCEDURE] ( {CEA} ) and carbohydrate antigen 19-9 [DIAGNOSTIC_PROCEDURE] ( {CA19-9} ), were all {within {normal ranges}} .<br>**Ground-truth**: *None* **Pred. of DICE**: (within normal ranges, Lab_value) was predicted as the argument for both events. |

Table 5: Qualitative analysis. We mark event trigger [EVENT_TYPE], ground-truth mentions and {mention prediction} made by our MI module.

ED module mistakenly includes incorrect descriptors. In the second example, DICE hallucinates that a DISTANCE descriptor "15 cm" is part of the BIOLOGICAL_STRUCTURE "segment IVb", which indicates that the EAE module struggles to separate mention boundaries. In the third example, the first event "biopsies" is missed by both the ED module and the MI module. However, despite the MI module correctly identifying "ductal carcinoma" as a mention, the ED module does not identify it as an event trigger. In the fourth example, DICE identifies "within normal ranges" as the LAB_VALUE for the two DIAGNOSTIC_PROCEDURE events, which are not valid LAB_VALUE for tumor marker tests.

## 6 Experiments in the General Domain

We evaluate DICE's generalizability by performing EE on the widely-used news-domain dataset ACE05 (Doddington et al., 2004), which contains 33 event types and 22 distinct argument roles. We perform both full-shot and low-resource experiments with 10% of the training data using the same data pre-processing, data splits and metrics as prior works (Wadden et al., 2019; Lin et al., 2020), and we compare with the same set of baselines introduced in §5.1. Baseline selection criteria and more results are presented in Appendix C.1.

We show the result in Table 6. We observe that DICE achieves a better performance on both low and high-resource settings for both trigger and argument classification tasks. We observe that DEGREE's performance is much closer to our model than in the clinical domain, which is due to two factors. First, the benefit of the independent query design used in DICE is diminished because ACE05 has far fewer argument roles that need to be filled in for each event type (on average 4.73) compared with in MACCROBAT-EE (on average 10). Second, DEGREE benefits from the implicit argument role dependencies established in its human-created event templates for ACE05, which were unavailable for the clinical domain. We also observe that mentions in the general domain are easier to identify as our MI module achieves 92% F1 score for entity identification on ACE05, while achieving 77% F1 score on MACCROBAT-EE. Although the mentions in the general domain are not as complex as clinical mentions, the performance of DICE supports our claim that mention-enhanced event extraction generalizes to the general domain.

| Model | 10% | | 100% | |
|---|---|---|---|---|
| | Tri-C | Arg-C | Tri-C | Arg-C |
| Text2Event | 47.0‡ | 24.9‡ | 71.9† | 53.8† |
| OneIE | 61.5‡ | 26.8‡ | <u>74.7</u>† | <u>56.8</u>† |
| DEGREE | <u>66.1</u>† | <u>42.1</u>† | 72.2† | 55.8† |
| DICE | **68.9** | **44.7** | **75.5** | **57.6** |

Table 6: ED and EAE performance (%) on the general domain dataset ACE05. We report the numbers from the original paper (†) or (Hsu et al., 2022) (‡). **Boldface** denotes the best results while <u>underscore</u> denotes the second best. DICE achieves state-of-the-art performance across both resource settings and tasks.

## 7 Conclusion and Future Work

We present DICE, a generative event extraction model designed for the clinical domain. DICE is adapted to tackle long and complicated mentions by conducting contrastive learning on instances with mention boundary perturbation, jointly optimizing EE tasks with the auxiliary mention identification task as well as the addition of mention boundary markers. We also introduce MACCROBAT-EE, the first clinical EE dataset with argument annotation as a testbench for future clinical EE works. Lastly, our evaluation shows that DICE achieves state-of-the-art EE performance in the clinical and news domains. In the future, we aim to apply transfer learning from higher-resource domains.

## Acknowledgments

Many thanks to I-Hung Hsu, Derek Xu, Tanmay Parekh and Masoud Monajatipoor for internal reviews, to lab members at PLUS lab, ScAi and UCLA-NLP for suggestions, and to the anonymous reviewers for their feedback. This work was partially supported by NSF 2106859, 2200274, AFOSR MURI grant #FA9550-22-1-0380, Defense Advanced Research Project Agency (DARPA) grant #HR00112290103/HR0011260656, and a Cisco Sponsored Research Award.

## Limitations

This work presents a repurposing of an existing dataset, MACCROBAT, and a set of novel techniques for adapting event extraction to the clinical domain. Among these new techniques is the handling of long-tailed argument roles, in which we independently query each role type. This presents an issue with scalability to domains with yet more complexity, as training the full DICE while querying both all event types and all argument types present in MACCROBAT-EE requires considerable resources during inference.

## Ethical Statement

Our experiments and proposed model framework are intended to encourage exploration in the clinical information extraction domain while avoiding the risk of privacy leakage. The data we use in this work is publicly available and fully de-identified. Though recent research has found it to be difficult to reconstruct protected personal information from such data, there remains some small risk that future models may be able to do so. We have not altered the content of data in any that would increase the likelihood of such an occurrence and are thus not risking private information leakage.

## References

David Ahn. 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.

Sophia Ananiadou, Sampo Pyysalo, Jun'ichi Tsujii, and Douglas B Kell. 2010. Event extraction for systems biology by text mining the literature. *Trends in biotechnology*, 28(7):381–390.

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. 2015. Scheduled sampling for sequence prediction with recurrent neural networks. *Advances in neural information processing systems*, 28.

Steven Bethard, Guergana Savova, Wei-Te Chen, Leon Derczynski, James Pustejovsky, and Marc Verhagen. 2016. SemEval-2016 task 12: Clinical TempEval. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 1052–1062, San Diego, California. Association for Computational Linguistics.

J Harry Caufield, Yichao Zhou, Yunsheng Bai, David A Liem, Anders O Garlid, Kai-Wei Chang, Yizhou Sun, Peipei Ping, and Wei Wang. 2019. A comprehensive typing system for information extraction from clinical narratives. *medRxiv*, page 19009118.

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).

Xinya Du and Claire Cardie. 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.

Sunyang Fu, David Chen, Huan He, Sijia Liu, Sungrim Moon, Kevin J. Peterson, Feichen Shen, Liwei Wang, Yanshan Wang, Andrew Wen, Yiqing Zhao, Sunghwan Sohn, and Hongfang Liu. 2020. Clinical concept extraction: A methodology review. *Journal of Biomedical Informatics*, 109:103526.

Yutai Hou, Wanxiang Che, Yongkui Lai, Zhihan Zhou, Yijia Liu, Han Liu, and Ting Liu. 2020. Few-shot slot tagging with collapsed dependency transfer and label-enhanced task-adaptive projection network. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1381–1393, Online. Association for Computational Linguistics.

I-Hung Hsu, Kuan-Hao Huang, Elizabeth Boschee, Scott Miller, Prem Natarajan, Kai-Wei Chang, and Nanyun Peng. 2022. DEGREE: A data-efficient generation-based event extraction model. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1890–1908, Seattle, United States. Association for Computational Linguistics.

William Hsu, Simon X Han, Corey W Arnold, Alex AT Bui, and Dieter R Enzmann. 2016. A data-driven approach for quality assessment of radiologic interpretations. *Journal of the American Medical Informatics Association*, 23(e1):e152–e156.

Kuan-Hao Huang, I-Hung Hsu, Prem Natarajan, Kai-Wei Chang, and Nanyun Peng. 2022. Multilingual generative language models for zero-shot cross-lingual event argument extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4633–4646, Dublin, Ireland. Association for Computational Linguistics.

Kung-Hsiang Huang, Sam Tang, and Nanyun Peng. 2021. Document-level entity-based extraction as template generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5257–5269, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Kung-Hsiang Huang, Mu Yang, and Nanyun Peng. 2020. Biomedical event extraction with hierarchical knowledge graphs. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1277–1285, Online. Association for Computational Linguistics.

Jin-Dong Kim, Yue Wang, Toshihisa Takagi, and Akinori Yonezawa. 2011. Overview of Genia event task in BioNLP shared task 2011. In *Proceedings of BioNLP Shared Task 2011 Workshop*, pages 7–15, Portland, Oregon, USA. Association for Computational Linguistics.

Jin-Dong Kim, Yue Wang, and Yamamoto Yasunori. 2013. The Genia event extraction shared task, 2013 edition - overview. In *Proceedings of the BioNLP Shared Task 2013 Workshop*, pages 8–15, Sofia, Bulgaria. Association for Computational Linguistics.

Qi Li, Heng Ji, and Liang Huang. 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.

Sha Li, Heng Ji, and Jiawei Han. 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.

Zhijing Li, Chen Li, Yu Long, and Xuan Wang. 2020. A system for automatically extracting clinical events with temporal information. *BMC Medical Informatics and Decision Making*, 20(1):1–13.

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.

Xiao Liu, Heyan Huang, Ge Shi, and Bo Wang. 2022. Dynamic prefix-tuning for generative template-based event extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5216–5228, Dublin, Ireland. 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.

Mingyu Derek Ma, Muhao Chen, Te-Lin Wu, and Nanyun Peng. 2021a. HyperExpan: Taxonomy expansion with hyperbolic representation learning. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4182–4194, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Mingyu Derek Ma, Jiun-Yu Kao, Shuyang Gao, Arpit Gupta, Di Jin, Tagyoung Chung, and Nanyun Peng. 2023a. Parameter-efficient low-resource dialogue state tracking by prompt tuning. In *Proc. Interspeech 2023*.

Mingyu Derek Ma, Jiao Sun, Mu Yang, Kung-Hsiang Huang, Nuan Wen, Shikhar Singh, Rujun Han, and Nanyun Peng. 2021b. EventPlus: A temporal event understanding pipeline. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Demonstrations*, pages 56–65, Online. Association for Computational Linguistics.

Mingyu Derek Ma, Xiaoxuan Wang, Po-Nien Kung, P. Jeffrey Brantingham, Nanyun Peng, and Wei Wang. 2023b. STAR: Boosting low-resource event extraction by structure-to-text data generation with large language models. *arXiv preprint arXiv:2305.15090*.

Yubo Ma, Zehao Wang, Yixin Cao, Mukai Li, Meiqi Chen, Kun Wang, and Jing Shao. 2022. Prompt for extraction? PAIE: Prompting argument interaction for event argument extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6759–6774, Dublin, Ireland. Association for Computational Linguistics.

Yu Meng, Chenyan Xiong, Payal Bajaj, Paul Bennett, Jiawei Han, Xia Song, et al. 2021. Coco-lm: Correcting and contrasting text sequences for language model pretraining. *Advances in Neural Information Processing Systems*, 34:23102–23114.

Minh Van Nguyen, Viet Dac Lai, and Thien Huu Nguyen. 2021. Cross-task instance representation

interactions and label dependencies for joint information extraction with graph convolutional networks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 27–38, Online. Association for Computational Linguistics.

Minh Van Nguyen, Bonan Min, Franck Dernoncourt, and Thien Nguyen. 2022a. Joint extraction of entities, relations, and events via modeling inter-instance and inter-label dependencies. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4363–4374, Seattle, United States. Association for Computational Linguistics.

Minh Van Nguyen, Bonan Min, Franck Dernoncourt, and Thien Nguyen. 2022b. Learning cross-task dependencies for joint extraction of entities, events, event arguments, and relations. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9349–9360, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Tomoko Ohta, Sampo Pyysalo, and Jun'ichi Tsujii. 2011. Overview of the epigenetics and post-translational modifications (EPI) task of BioNLP shared task 2011. In *Proceedings of BioNLP Shared Task 2011 Workshop*, pages 16–25, Portland, Oregon, USA. Association for Computational Linguistics.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *ArXiv preprint*, abs/1807.03748.

Giovanni Paolini, Ben Athiwaratkun, Jason Krone, Jie Ma, Alessandro Achille, RISHITA ANUBHAI, Cicero Nogueira dos Santos, Bing Xiang, and Stefano Soatto. 2021. Structured prediction as translation between augmented natural languages. In *International Conference on Learning Representations*.

Sampo Pyysalo, Tomoko Ohta, Makoto Miwa, Han-Cheol Cho, Jun'ichi Tsujii, and Sophia Ananiadou. 2012. Event extraction across multiple levels of biological organization. *Bioinformatics*, 28(18):i575–i581.

Sampo Pyysalo, Tomoko Ohta, Rafal Rak, Dan Sullivan, Chunhong Mao, Chunxia Wang, Bruno Sobral, Jun'ichi Tsujii, and Sophia Ananiadou. 2011. Overview of the infectious diseases (ID) task of BioNLP shared task 2011. In *Proceedings of BioNLP Shared Task 2011 Workshop*, pages 26–35, Portland, Oregon, USA. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Alan Ramponi, Rob van der Goot, Rosario Lombardo, and Barbara Plank. 2020. Biomedical event extraction as sequence labeling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5357–5367, Online. Association for Computational Linguistics.

Christian M Rochefort, David L Buckeridge, and Alan J Forster. 2015. Accuracy of using automated methods for detecting adverse events from electronic health record data: a research protocol. *Implementation Science*, 10(1):1–9.

Dinghan Shen, Mingzhi Zheng, Yelong Shen, Yanru Qu, and Weizhu Chen. 2020. A simple but tough-to-beat data augmentation approach for natural language understanding and generation. *arXiv preprint arXiv:2009.13818*.

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.

Ergin Soysal, Jingqi Wang, Min Jiang, Yonghui Wu, Serguei Pakhomov, Hongfang Liu, and Hua Xu. 2017. CLAMP – a toolkit for efficiently building customized clinical natural language processing pipelines. *Journal of the American Medical Informatics Association*, 25(3):331–336.

Kai Sun, Richong Zhang, Samuel Mensah, Yongyi Mao, and Xudong Liu. 2020. Recurrent interaction network for jointly extracting entities and classifying relations. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3722–3732, Online. Association for Computational Linguistics.

Hai-Long Trieu, Thy Thy Tran, Khoa NA Duong, Anh Nguyen, Makoto Miwa, and Sophia Ananiadou. 2020. Deepeventmine: end-to-end neural nested event extraction from biomedical texts. *Bioinformatics*, 36(19):4910–4917.

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

Sijia Wang, Mo Yu, Shiyu Chang, Lichao Sun, and Lifu Huang. 2022. Query and extract: Refining event extraction as type-oriented binary decoding. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 169–182, Dublin, Ireland. Association for Computational Linguistics.

Xiaoyan Wang, George Hripcsak, Marianthi Markatou, and Carol Friedman. 2009. Active computerized pharmacovigilance using natural language processing, statistics, and electronic health records: a feasibility study. *Journal of the American Medical Informatics Association*, 16(3):328–337.

Xing David Wang, Leon Weber, and Ulf Leser. 2020. Biomedical event extraction as multi-turn question answering. In *Proceedings of the 11th International Workshop on Health Text Mining and Information Analysis*, pages 88–96, Online. 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.

Jiashu Xu, Mingyu Derek Ma, and Muhao Chen. 2023. Can NLI provide proper indirect supervision for low-resource biomedical relation extraction? In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*, Toronto, Canada. Association for Computational Linguistics.

Kabir Yadav, Efsun Sarioglu, Meaghan Smith, and Hyeong-Ah Choi. 2013. Automated outcome classification of emergency department computed tomography imaging reports. *Academic Emergency Medicine*, 20(8):848–854.

Hang Yan, Tao Gui, Junqi Dai, Qipeng Guo, Zheng Zhang, and Xipeng Qiu. 2021. A unified generative framework for various NER subtasks. 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 5808–5822, Online. Association for Computational Linguistics.

Bishan Yang and Tom M. Mitchell. 2016. Joint extraction of events and entities within a document context. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 289–299, San Diego, California. Association for Computational Linguistics.

Sen Yang, Dawei Feng, Linbo Qiao, Zhigang Kan, and Dongsheng Li. 2019. Exploring pre-trained language models for event extraction and generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5284–5294, Florence, Italy. Association for Computational Linguistics.

Hongbin Ye, Ningyu Zhang, Shumin Deng, Xiang Chen, Hui Chen, Feiyu Xiong, Xi Chen, and Huajun Chen. 2022. Ontology-enhanced prompt-tuning for few-shot learning. In *Proceedings of the ACM Web Conference 2022*, WWW '22, page 778–787, New York, NY, USA. Association for Computing Machinery.

Junchi Zhang, Yanxia Qin, Yue Zhang, Mengchi Liu, and Donghong Ji. 2019. Extracting entities and events as a single task using a transition-based neural model. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 5422–5428. International Joint Conferences on Artificial Intelligence Organization.

Zixuan Zhang and Heng Ji. 2021. Abstract Meaning Representation guided graph encoding and decoding for joint information extraction. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 39–49, Online. Association for Computational Linguistics.

Sendong Zhao, Ting Liu, Sicheng Zhao, and Fei Wang. 2019. A neural multi-task learning framework to jointly model medical named entity recognition and normalization. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):817–824.

## A Potential Questions

**What is the difference between the existing generative EE model DEGREE and DICE?** Compared with DEGREE, our model: 1) further enhances the EE model's specialization in mention identification by three techniques to learn mention-related capabilities introduced in §4.2 to handle long clinical mentions with vague boundaries; and 2) performs an independent query for each argument role for better long-tail performance in settings with many argument roles as introduced in §4.1.

**Would training and inference efficiency be an issue?** As we perform an independent query for each event type/argument role in the ED/EAE model, it is a tradeoff between performance and running cost. Though during training, we only sample a subset of negative instances to train the model for faster convergence. For example, to create seq2seq input-output pairs for a certain sentence for ED, we create 1 positive pair (*i.e.* there is an event in the sentence for the query event type) and $k$ (instead of $n_{event\_type}$, where $k$ is much smaller than $n_{event\_type}$) negative pairs (*i.e.* no event exists for the query event type).

**Why use standalone MI modules to produce mention candidates?** We use standalone trigger and argument MI modules to create markers for downstream ED+MI and EAE+MI joint models, instead of using the MI module jointly trained in the ED+MI or EAE+MI models because the standalone one yields better performance.

## B Dataset MACCROBAT-EE Details

### B.1 MACCROBAT Annotation

MACCROBAT is annotated according to the Annotation for Case Reports using Open Biomedical Annotation Terms (ACROBAT) defined in (Caufield et al., 2019). ACROBAT describes events and entities as meaningful text spans, but differentiates events as occurrences that may be ordered chronologically and entities as objects that may modify or describe events. According to the annotation guideline, entity text spans are limited to the shortest viable length. Each event and entity is given a type such that certain events are associated with certain argument roles. According to ACROBAT, Entity text spans are limited to the shortest viable length. For example, the text span "mild asthma attack" would be annotated by labeling "asthma attack" as an event as that is the shortest span that conveys the occurrence of the event. "Mild" would be labeled an entity and the annotation would add a relation indicating that "mild" modifies "asthma attack". MACCROBAT contains 12 relation types, but for our purposes we only consider the MODIFY relation that occurs when an entity describes or characterizes an event.

### B.2 Details of Inferring Event Arguments

According to ACE2005 English Events Guidelines (AEEG),[6] the arguments of events are defined as entities and values within the scope of an event and only the closest entities and values will be selected, where a value is defined to be "a string that further characterizes the properties of some Entity or Event". The MODIFY relation in the MACCROBAT dataset connects 2 arguments, and it is defined as the "generic relationship in which one entity or event modifies another entity or event, including instances where an entity is identified following an event" (Caufield et al., 2019). The MODIFY relation satisfies the argument definition described by the AEEG by incorporating within-sentence relationships between an entity that modifies or describes an event. Thus, given a certain event trigger, we consider non-event entities that hold a MODIFY relation with the trigger as arguments of this event. We take the assigned type of the selected entity according to MACCROBAT as the role of the argument. To create an event ontology, which includes all possible event types and possible argument roles or each event type, we traverse all (event type, argument role) pairs to obtain the unique argument roles possible for each event type.

### B.3 Event Ontology

We show the full event ontology, including all event types and their possible argument roles, in Table 12.

## C Additional Experimental Results

### C.1 Additional Baselines for General Domain Event Extraction

**Baseline selection criteria.** We select published EE models reporting performance on the ACE05 dataset using ED and EAE training data *only* without using external resources (*e.g.* knowledge graph)

[6] https://www.ldc.upenn.edu/collaborations/past-projects/ace/annotation-tasks-and-specifications

or additional tasks (*e.g.* relation extraction, entity recognition) as our baselines for general domain EE experiments. We use the same data pre-processing, data splits and metrics as prior works (Wadden et al., 2019; Lin et al., 2020).

**Additional baselines.** In addition to the baselines we introduced in §5.1, we compare with **DyGIE++** (Wadden et al., 2019), a span graph-enhanced classification model for EE; **BERT_QA** (Du and Cardie, 2020), which formulates EE as an extractive question answering task with a sequence tagging classifier; **TANL** (Paolini et al., 2021), which frames EE as a translation task between augmented natural languages; **BART-Gen** (Li et al., 2021), which uses a sequence tagging model (Hou et al., 2020) with additional keywords as input for ED and performs EAE by filling in event template with a conditional generation model; and **GTEE-DynPref** (Liu et al., 2022), which tunes dynamic prefix for generative EE models.

We do not compare with Nguyen et al. (2022b, 2021); Zhang and Ji (2021) because they jointly learn additional tasks besides ED and EAE (*i.e.* entity recognition and relation extraction) and there is no codebase provided by the time of this work. We do not compare with Wang et al. (2022) since its performance is worse than DEGREE (Hsu et al., 2022) and GTEE-DynPref (Liu et al., 2022) according to Nguyen et al. (2022b).

| # | Model | 10% Tri-C | 10% Arg-C | 100% Tri-C | 100% Arg-C |
|---|---|---|---|---|---|
| 1 | DyGIE++ | – | 15.7 $^{\S}$ | 70.0$^{\ddagger}$ | 50.0$^{\ddagger}$ |
| 2 | BERT_QA | 50.1$^{\ddagger}$ | 27.6$^{\ddagger}$ | 72.4$^{\dagger}$ | 53.3$^{\dagger}$ |
| 3 | TANL | 54.8$^{\ddagger}$ | 29.0$^{\ddagger}$ | 68.4$^{\ddagger}$ | 47.6$^{\ddagger}$ |
| 4 | BART-Gen | – | – | 71.1$^{\dagger}$ | 53.7$^{\dagger}$ |
| 5 | GTEE-DynPref | – | – | 72.6$^{\dagger}$ | 55.8$^{\dagger}$ |
| 6 | Text2Event | 47.0$^{\ddagger}$ | 24.9$^{\ddagger}$ | 71.9$^{\dagger}$ | 53.8$^{\dagger}$ |
| 7 | OneIE | 61.5$^{\ddagger}$ | 26.8$^{\ddagger}$ | <u>74.7</u>$^{\dagger}$ | <u>56.8</u>$^{\dagger}$ |
| 8 | DEGREE | <u>66.1</u>$^{\dagger}$ | <u>42.1</u>$^{\dagger}$ | 72.2$^{\dagger}$ | 55.8$^{\dagger}$ |
| 9 | DICE | **68.9** | **44.7** | **75.5** | **57.6** |

Table 7: ED and EAE performance (F1 score, %) on the general domain dataset ACE05. We report the numbers from the original paper (indicated by †), (Hsu et al., 2022) (indicated by ‡) and (Ye et al., 2022) (indicated by §). **Boldface** denotes the best results while <u>underscore</u> denotes the second best. DICE achieves the best performance across both resource settings and tasks.

**Experimental results.** Table 7 shows the comparison with more baselines.

### C.2 Additional Ablation Studies

**Input prompt segments.** We analyze the importance of prompt segments in Table 8. For ED, we find that event type name is more important. For EAE, removing either the event type description (Line 5) or the argument role description (Line 9) leads to the most significant performance decreases. These results emphasize the benefits of incorporating the rich semantic information contained in the names and definitions for both event type and argument roles.

| # | Prompt segments | Identification P | Identification R | Identification F1 | Classification P | Classification R | Classification F1 |
|---|---|---|---|---|---|---|---|
| | Event Detection | | | | | | |
| 1 | w/o type name | 71.19 | 63.32 | 67.02 | 67.41 | 60.73 | 63.90 |
| 2 | w/o type description | 66.38 | 71.00 | 68.61 | 62.28 | 67.19 | 64.64 |
| 3 | Vanilla DICE | 65.03 | 74.08 | 69.26 | 60.51 | 70.28 | 65.03 |
| | Event Argument Extraction | | | | | | |
| 4 | w/o type name | 69.34 | 77.35 | 73.13 | 64.17 | 73.03 | 68.31 |
| 5 | w/o type description | 67.80 | 77.45 | 72.31 | 62.94 | 73.46 | 67.79 |
| 6 | w/o trigger phrase | 71.20 | 77.89 | 74.39 | 66.21 | 73.79 | 69.80 |
| 7 | w/o trigger marker | 68.55 | 78.53 | 73.20 | 64.70 | 75.51 | 69.69 |
| 8 | w/o arg. role name | 70.13 | 77.99 | 73.85 | 65.20 | 73.79 | 69.23 |
| 9 | w/o role description | 75.22 | 70.54 | 72.81 | 67.91 | 65.39 | 66.63 |
| 10 | Vanilla DICE | 70.76 | 76.48 | 73.51 | 66.47 | 72.71 | 69.45 |

Table 8: Ablation study of prompt segments.

**Extraction vs typing formulation.** We formulate ED and EAE as conditional text generation tasks and consider two designs for our input and target format. The first is the DICE design in which we expect the model to **extract** content given queries with event type/argument role information. The second design formulates a **typing** task that provides a query to the generative model for each mention so that the expected output is the predicted event type or argument role for the querying mention. This approach is motivated by the notion that the output space of the typing formulation is much smaller than that of the extraction task.

We formulate the ED and EAE tasks as typing tasks by querying each possible mention. For the ED task, we first use the standalone mention identification module introduced in §4.1 to extract all possible triggers detected by the MI module, and then we query the generative model with the following example input and output format:

> Input: ... calcified <query>plaque</query> ... artery.
> Output: Event type is Sign_symptom.

The output is constrained to belong to the candidate pool of event types or the placeholder event type "<Type>" following the prefix "Event type is ". For the EAE task, we first extract all possible argument candidates and then query each candidate with the input sentence containing event trigger, event trigger marker, event type name and event type description:

> Input: ... densely <query>calcified</query> <trigger>plaque</trigger> ... artery. \n Event type is Sign_symptom. \n Any symptom or clinical finding. \n Event trigger is plaque.
> Output: Argument role is Detailed_description.

Similarly, the output is constrained to the candidate pool of argument roles possible for the given event type following the prefix "Argument role is ".

| # | Formulation | Identification P | Identification R | Identification F1 | Classification P | Classification R | Classification F1 |
|---|---|---|---|---|---|---|---|
| | Event Detection | | | | | | |
| 1 | Typing | 74.64 | 67.19 | 70.72 | 69.24 | 63.82 | 66.42 |
| 2 | Extraction | 65.03 | 74.08 | 69.26 | 60.51 | 70.28 | 65.03 |
| | Event Argument Extraction | | | | | | |
| 3 | Typing | 58.59 | 44.95 | 50.87 | 53.63 | 41.14 | 46.56 |
| 4 | Extraction | 70.76 | 76.48 | 73.51 | 66.47 | 72.71 | 69.45 |

Table 9: Ablation study of generative task formulation.

The results in Table 9 show that the typing formulation improves ED performance over extraction (though still worse than mention-enhanced DICE), but leads to a much worse EAE performance. This is likely due to the typing task becoming more difficult as the number of candidate class increases and complicated typing spaces varied by event types.

### C.3 Full Low-Resource Results

We show the full low-resource experimental results illustrated in Figure 3 in Table 10.

## D Details of Implementation and Experiments

### D.1 Implementation Details

**Mention Identification.** The sliding window scans the passage from beginning to end with a pre-defined window size and step size, which significantly boosts the coverage of the predicted mentions. During both training and inference, we retain the original full-length input passage in addition to the sliding window segments.

| Model | 10% | 25% | 50% | 75% | 10% | 25% | 50% | 75% |
|---|---|---|---|---|---|---|---|---|
| | Trigger Identification | | | | Trigger Classification | | | |
| Text2Event | – | – | – | – | 52.54 | 53.72 | 59.21 | 62.78 |
| OneIE | 68.22 | 71.28 | 73.73 | 74.47 | 61.46 | 65.08 | 67.54 | 68.50 |
| DEGREE | 62.12 | 63.78 | 66.32 | 69.73 | 58.31 | 61.03 | 63.14 | 64.77 |
| DICE | **71.47** | **72.79** | **74.07** | **74.88** | **65.82** | **66.54** | **67.91** | **68.72** |
| | Argument Identification | | | | Argument Classification | | | |
| Text2Event | – | – | – | – | 37.74 | 40.09 | 46.53 | 50.37 |
| OneIE | 32.13 | 39.65 | 43.75 | 47.12 | 24.95 | 32.36 | 35.70 | 38.55 |
| DEGREE | 26.60 | 30.41 | 31.06 | 31.63 | 26.60 | 28.43 | 29.48 | 29.59 |
| DICE | **49.97** | **53.55** | **54.42** | **55.83** | **45.67** | **48.97** | **50.42** | **52.83** |

Table 10: Performance on the downsampled training sets. We report the F1 score for each task using different downsampled training data. We create three random splits for each proportion and report the average performance.

**Training and evaluation.** We select the best epoch based on the highest F1 score of the most downstream MI/ED/EAE task on the validation set. When evaluating correctness, we only accept an exact match between the generated trigger/argument and the ground-truth trigger/argument as a correct prediction. We use beam search with 2 beams to generate the output sequences for all three generative tasks. The generation stops either when the "end_of_sentence" token is generated or the output length reaches 30.

**Frameworks.** Our entire codebase is implemented in PyTorch.[7] The implementations of the transformer-based models are extended from the Huggingface[8] codebase (Wolf et al., 2020).

### D.2 Experiments Details

We report the median result for five runs with different random seeds by default. For the low-resources result shown in Figure 3, we sample different selections of training data of corresponding proportion for each run. All the models in this work are trained on NVIDIA A6000 GPUs on a Ubuntu 20.04.2 operating system.

### D.3 Baseline Reproduction

**Mention Identification.** For results in Table 4, we use BART-large for Yan et al. because Yan et al. (2021) only supports a generative model with absolute position embedding. OneIE uses BERT-large as its default and we use T5-large for our proposed DICE-MI module.

[7] https://pytorch.org/
[8] https://github.com/huggingface/transformers

**ED and EAE.** We use authors' codebases to produce baseline results. OneIE jointly learns ED, EAE, and MI tasks and we provide entity information to its MI module with event types and role types stripped to equate its training information with the training information provided to our model DICE. For DEGREE, human-written templates that organize the argument roles of an event type in a sentence are required by the model. We construct these templates using phrases such as "<Argument role> is <argument text>" for all potential argument roles of an event type as the template.

### D.4 Hyperparameters

For the ED module, we define positive instances as (PASSAGE, EVENT TYPE) pairs where the passage contains one or more event triggers of this event type. Negative instances are pairs in which the passage contains no event triggers of the event type. We create 10 negative instances for each positive instance. For the EAE module, we define positive instance as the (PASSAGE, EVENT TRIGGER, EVENT TYPE, ARGUMENT ROLE) tuple that there exists an argument text contained in the passage that meets the query criteria. We create 10 negative instances for each positive instance. For the MI module, we use a window size of 10 words, with a sliding step of 4 words. We retain the original full sequence in both training and evaluation. We use an AdamW optimizer with a 1e-5 learning rate without gradient accumulation. We show the hyperparameter search ranges and the final choices in Table 11.

| Hyperparameter | Search Range | Best |
|---|---|---|
| Negative instance # for ED | 1, 2, 3, 4, 5, 8, 10, all | 10 |
| Negative instance # for EAE | 1, 2, 3, 4, 5, 8, 10, all | 10 |
| MI module sliding window size | 4, 6, 8, 10, 12 | 10 |
| MI module sliding window step | 2, 4, 6, 8, 10 | 4 |
| MI module sliding window retains original long sequence during training | True, False | True |
| MI module sliding window retains original long sequence during inference | True, False | False |
| Batch size | 1, 2, 3, 4 | 4 |
| Learning rate | 1e-4, 5e-5, 1e-5, 5e-6, 1e-6 | 1e-5 |
| Decoding method | beam search, greedy | beam search |
| Max epochs | | 70 |

Table 11: Hyperparameter search ranges and the best settings.

| Event Type | Role |
|---|---|
| Sign_symptom | Biological_structure, Detailed_description, Severity, Lab_value, Distance, Shape, Area, Color, Texture, Frequency, Volume, Quantitative_concept, Qualitative_concept, Biological_attribute, Subject, Other_entity, History, Mass |
| Diagnostic_procedure | Lab_value, Biological_structure, Detailed_description, Qualitative_concept, Nonbiological_location, Frequency,Distance, Subject, Shape, Quantitative_concept, Texture, Severity, Age, Color, Area, Volume, Administration, Mass |
| Therapeutic_procedure | Detailed_description, Biological_structure, Lab_value, Dosage, Nonbiological_location, Frequency, Distance,Qualitative_concept, Subject, Quantitative_concept, Area, Administration, Other_entity |
| Disease_disorder | Detailed_description, Biological_structure, Severity, Lab_value, Quantitative_concept, Distance, Nonbiological_location, Shape, Volume, Qualitative_concept, Area, Subject, Biological_attribute |
| Medication | Dosage, Administration, Detailed_description, Frequency, Lab_value, Nonbiological_location, Quantitative_concept, Biological_structure, Volume |
| Clinical_event | Nonbiological_location, Detailed_description, Frequency, Biological_structure, Subject, Lab_value, Quantitative_concept, Volume |
| Lab_value | Biological_structure, Detailed_description, Color, Severity, Frequency |
| Activity | Detailed_description, Nonbiological_location, Biological_structure, Other_entity, Frequency, Lab_value, Quantitative_concept |
| Other_event | Biological_structure, Quantitative_concept, Nonbiological_location, Severity, Detailed_description |
| Outcome | Nonbiological_location, Subject, Detailed_description, Age |
| Date | - |
| Time | - |
| Duration | - |

Table 12: Event types and corresponding argument roles in MACCROBAT-EE, the argument roles are ordered by their appearance frequency. The most appeared roles are listed first.

## ACL 2023 Responsible NLP Checklist

### A For every submission:

☑ A1. Did you describe the limitations of your work?
*Limitation section after Conclusion*

☑ A2. Did you discuss any potential risks of your work?
*Ethical statement section*

☑ A3. Do the abstract and introduction summarize the paper's main claims?
*Abstract section. Section 1: Introduction.*

☒ A4. Have you used AI writing assistants when working on this paper?
*Left blank.*

### B ☑ Did you use or create scientific artifacts?

*Section 3 for data, Section 4 for model*

☑ B1. Did you cite the creators of artifacts you used?
*Section 3 for data, Section 4 for model*

☐ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
*Not applicable. Left blank.*

☐ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
*Not applicable. Left blank.*

☒ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
*We use a previously published dataset, the anonymization work has been done in previous work*

☒ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
*We use a previously published dataset, the anonymization work has been done in previous work*

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
*Section 3.2 The MACCROBAT-EE Dataset*

### C ☑ Did you run computational experiments?

*Section 5 and 6*

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
*Appendix D*

*The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.*

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
*Section 5 and 6, Appendix D*

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
*Appendix C3*

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
*Appendix D*

**D ☑ Did you use human annotators (e.g., crowdworkers) or research with human participants?**
*Section 3.4*

☒ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
*The annotation task is simple, we provide a textual description of the task*

☒ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
*Authors are served as annotators directly without additional recruitment*

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
*Not applicable. Left blank.*

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
*Not applicable. Left blank.*

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
*Not applicable. Left blank.*