JSEEGraph: Joint Structured Event Extraction as Graph Parsing

Huiling You[1], **Samia Touileb**[2] and **Lilja Øvrelid**[1]
[1]University of Oslo
[2]University of Bergen
{huiliny, liljao}@ifi.uio.no
samia.touileb@uib.no

Abstract

We propose a graph-based event extraction framework JSEEGraph that approaches the task of event extraction as general graph parsing in the tradition of Meaning Representation Parsing. It explicitly encodes entities and events in a single semantic graph, and further has the flexibility to encode a wider range of additional IE relations and jointly infer individual tasks. JSEEGraph performs in an end-to-end manner via general graph parsing: (1) instead of flat sequence labelling, nested structures between entities/triggers are efficiently encoded as separate nodes in the graph, allowing for nested and overlapping entities and triggers; (2) both entities, relations, and events can be encoded in the same graph, where entities and event triggers are represented as nodes and entity relations and event arguments are constructed via edges; (3) joint inference avoids error propagation and enhances the interpolation of different IE tasks. We experiment on two benchmark datasets of varying structural complexities; ACE05 and Rich ERE, covering three languages: English, Chinese, and Spanish. Experimental results show that JSEEGraph can handle nested event structures, that it is beneficial to solve different IE tasks jointly, and that event argument extraction in particular benefits from entity extraction. Our code and models are released as open-source[1].

1 Introduction

Event extraction (EE) deals with the extraction of complex, structured representations of events from text, including overlapping and nested structures (Sheng et al., 2021; Cao et al., 2022). While there are existing datasets annotated with such rich representations (Doddington et al., 2004; Song et al., 2015), a majority of current approaches model this task using simplified versions of these datasets or sequence-labeling-based encodings which are not capable of capturing the full complexity of the events. Figure 1 shows an example from the Rich ERE dataset (Song et al., 2015) of a sentence containing both nested and overlapping events: *"buy"* serves as trigger for two overlapping events, `transfermoney` and `transferownership` with their respective argument roles, and similarly *"made"* for two `artifact` events triggered by the coordination of two GPE entities *Canada* and *USA*; at the same time, the event trigger *"made"* is nested inside the entity span *"things made in Canada or USA"*. For this example, models based on token tagging (such as the commonly used BIO-encoding) would fail completely when a token contributes to multiple information extraction elements. In this case, the version of the ACE05 dataset widely employed for EE would not fully capture the double-tagged event triggers, by simply disregarding one of the two events, and the nested entity *"things made in Canada or USA"* would be *"things"*.

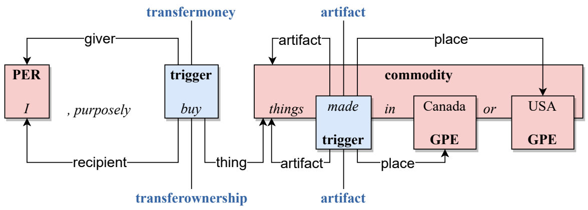

Figure 1: Example of nested and overlapping events in the sentence *"I, purposely buy things made in Canada or USA."*, *taken from Rich ERE (Song et al., 2015).*

Event extraction is a subtask of a wider set of Information Extraction (IE) tasks, jointly dealing with extracting various types of structured information from unstructured texts, from named entities, relations, to events. There have been continued efforts in creating benchmark datasets that can be used for evaluating a wide range of IE tasks. Both ACE05 (Doddington et al., 2004)[2] and Rich

[1]https://github.com/huiling-y/JSEEGraph

[2]https://catalog.ldc.upenn.edu/LDC2006T06

*Proceedings of the The 12th Joint Conference on Lexical and Computational Semantics (*SEM 2023)*, pages 115–127
July 13-14, 2023 ©2023 Association for Computational Linguistics

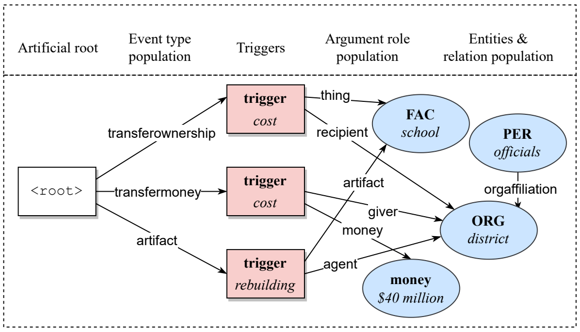

Figure 2: Example of graph representation for entities, relations, and events from the sentence *"School district officials have estimated the cost of rebuilding an intermediate school at $40 million."*, from Rich ERE (Song et al., 2015).

ERE (Song et al., 2015)[3] provide consistent annotations of entities, relations, and events. While there are clear inter-relations between these different elements, and despite the availability of rich annotations, existing works often deal with individual tasks, such as named entity recognition (NER) (Chiu and Nichols, 2016; Bekoulis et al., 2018) or event extraction (EE) (Yang and Mitchell, 2016; Du and Cardie, 2020; Li et al., 2020). Recently there have been some efforts in jointly modelling multiple IE tasks (Wadden et al., 2019; Lin et al., 2020; Nguyen et al., 2022), but these methods explicitly avoid nested instances.

We here propose to represent events, along with entities and relations, as general graphs and approach the task of event extraction as Meaning Representation Parsing (Oepen et al., 2020; Samuel and Straka, 2020). As shown in Figure 2, in such an information graph, event triggers and entities are represented as nodes; event types, argument roles, and relations are constrained edges; and nested/overlapped structures are straightforwardly represented, since a surface string can be abstracted into an unlimited number of nodes, as illustrated by the two separate nodes for the event triggers for *"cost"*. Our approach does not rely on ontology- or language-specific features or any external syntactic/semantic parsers, but directly parses raw text into an information graph. We experiment on the benchmark datasets ACE05 (Doddington et al., 2004) and Rich ERE (Song et al., 2015), zooming in on nested structures. Our results show JSEE-Graph to be versatile in solving entity, relation, and event extraction jointly, even for heavily nested instances and across three different languages. Ablation studies consistently show that event extraction especially benefits from entity extraction.

The paper is structured as follows: section 2 provides the relevant background for our work, and section 3 further describes the tasks addressed and the datasets we employ, focusing in particular on their complexity, as measured by level of nesting. Section 4 presents the JSEE graph parsing framework and section 5 the experimental setup for evaluating the JSEE parser. Section 6 presents the results of our evaluations and provides a study of the performance for nested structures, as well as an ablation study assessing the effect of joint IE modeling and an error analysis. Finally we provide conclusions (Section 7) and discuss limitations of our work.

2 Related work

Event extraction is commonly approached as supervised classification, even though other approaches relying *e.g.* on generation (Paolini et al., 2021; Lu et al., 2021; Li et al., 2021; Hsu et al., 2022) or prompt tuning inspired by natural language understanding tasks (Shin et al., 2020; Gao et al., 2021; Li and Liang, 2021; Liu et al., 2022) also are gaining ground. Classification-based methods break event extraction into several subtasks (trigger detection/classification, argument detection/-classification), and either solve them separately in a pipeline-based manner (Ji and Grishman, 2008; Li et al., 2013; Liu et al., 2020; Du and Cardie, 2020; Li et al., 2020) or jointly infer them as multiple subtasks (Yang and Mitchell, 2016; Nguyen et al., 2016; Liu et al., 2018; Wadden et al., 2019; Lin et al., 2020). Classification-based joint methods typically apply sequence-labeling-based encoding and extract all event components in one pass, whereas pipeline methods break the problem into separate stages which are performed sequentially. Whereas sequence-labeling approaches cannot distinguish overlapping events/arguments by the nature of the BIO-encoding, pipeline methods may in principle detect these. However, they typically suffer from error propagation and are not equipped to model the interactions between the different event elements (triggers, arguments).

Nested events Some previous work addresses the problem of overlapping or nested arguments in EE. Xu et al. (2020) address overlapping arguments in the Chinese part of the ACE05 dataset and jointly perform predictions for event triggers and argu-

[3] https://catalog.ldc.upenn.edu/LDC2020T18

ments based on common feature representations derived from a pre-trained language model. Sheng et al. (2021) propose a joint framework with cascaded decoding to tackle overlapping events, and sequentially perform type detection, event and argument extraction in a Chinese financial event dataset. They deal with cases of both "overlapping events" and "overlapping arguments", however, their approach may suffer from error propagation due to the cascading approach. Cao et al. (2022) distinguish between overlapped and nested events and propose the OneEE tagging scheme which formulates EE as a word-to-word relation recognition, distinguishing separate span and role relations. OneEE is evaluated on the FewFC Chinese financial event dataset and the biomedical event datasets Genia11 and Genia13. While specifically focusing on nested events, these previous works are limited by focusing only on one language or on specialized (financial/biomedical) domains. In this work we aim to provide a more comprehensive evaluation over two datasets in several versions with increasing levels of structural complexity (see below) and across three different languages.

Joint IE approaches Wadden et al. (2019) propose the DyGIE++ model which approaches joint modeling of IE entities and relations via span-based prediction of entities and event triggers, and subsequent dynamic graph propagation based on relations. They evaluate on ACE05 and Genia datasets and limit their experiments to English only. Their approach is restricted to a certain span width, limiting the length of possible entities. OneIE (Lin et al., 2020) is a joint system for IE using global features to model cross-subtask or cross-instance interactions between the subtasks and predict an information graph. They propose the E+ extension of ACE05 which includes multi-token events (E^+) as we do. As in our work, they also present results on Spanish and Chinese as well and develop a multilingual model, but their experiments avoid nested structures, by using only the head of entity mentions and specifically removing overlapped entities. Nguyen et al. (2022) model joint IE in a two-stage procedure which first identifies entities and event triggers and subsequently classify relations between these starting from a fully connected dependency graph; a GCN is employed to encode the resulting dependency graphs for computation of the joint distribution. While the approach is shown to be effective, it is still a pipeline approach which can suffer from error propagation. Since it relies on sequence labeling for entity/event detection, it cannot identify overlapping entities/event triggers. Furthermore, the approach relies on syntactic information from an external parser and focuses only on English and Spanish in the Light ERE dataset (Song et al., 2015).

Meaning Representation Parsing Meaning Representation Parsing (MRP) (Oepen et al., 2014, 2015, 2020) is a framework covering several types of dependency-based semantic graph frameworks. Unlike syntactic dependency representations, these semantic representations are not trees, but rather general graphs, characterised by potentially having multiple top nodes (*roots*) and not necessarily being connected, since not every token is necessarily a node in the graph. The semantic frameworks include representations with varying levels of "anchoring" to the input string (Oepen et al., 2020), ranging from the so-called "bi-lexical" representations where every node in the graph corresponds to a token in the input string to a framework like AMR (Banarescu et al., 2013) which constitutes the most abstract and unanchored type of framework, such that the correspondence between the nodes in a graph and tokens in the string is completely flexible. This allows for straightforward representation of nesting and overlapping structures, where multiple nodes may be anchored to overlapping sub-strings. There have been considerable progress in developing variants of both transition-based and graph-based dependency parsers capable of producing such semantic graphs (Hershcovich et al., 2017; Dozat and Manning, 2018; Samuel and Straka, 2020). Previous research has further made use of AMR-based input representations to constrain the tasks of event extraction (Huang et al., 2018) and more recently joint information extraction (Zhang and Ji, 2021), where an off-the-shelf AMR parser is used to derive candidate enitity and event trigger nodes before classifying pairwise relations guided by the AMR hierarchical structure. While there are clear parallels between the MRP semantic frameworks and the tasks proposed in IE, little work has focused on the direct application of MRP parsing techniques to these tasks. You et al. (2022) is a notable exception in this respect, who presents an adaptation of the PERIN semantic parser (Samuel and Straka, 2020) to the event extraction task. While their work is promising it is limited to only one dataset (ACE05), which does

Lang	Split	#Sents	#Events	#Roles	#Entities	#Relations
			Dataset: ACE05			
en	Train	19 371	4 419	6 609	47 546	7 172
	Dev	896	468	759	3 421	729
	Test	777	461	735	3 828	822
zh	Train	6 706	2 928	5 576	29 674	8 003
	Dev	511	217	406	2 246	601
	Test	521	190	336	2 389	686
			Dataset: Rich ERE			
en	Train	12 421	8 368	15 197	34 611	7 498
	Dev	692	459	797	1 998	366
	Test	745	566	1 195	2 286	544
zh	Train	9 253	5 325	9 066	26 128	6 044
	Dev	541	366	522	1 609	379
	Test	483	439	776	2 022	502
es	Train	8 292	5 013	8 575	20 347	4 140
	Dev	383	254	447	1 068	199
	Test	598	334	609	1 438	287

Table 1: Statistics of the preprocessed datasets.

not contain a lot of nested structures and is further limited to English event extraction only. In this work we extend their approach to the task of joint information extraction, covering both entities, events and relations taken from two different datasets in several versions and for three languages, and further demonstrates the effectiveness of approaching general information extraction from text via graph-parsing and the interpolation of different IE tasks.

3 Task and Data

While the main focus of this work is on event extraction, we hypothesize that our graph-based approach lends itself to dealing with two challenging aspects of current research on this task: the processing of nested and overlapping event structures, and the joint modeling of inter-related IE structures. In the following we quantify the level of nesting in two widely used datasets which contain rich annotations for both entities, events, and relations. We further propose two versions of each dataset with varying potential for nesting, which allows us to focus on this aspect during evaluation.

Event Extraction is the task of extracting events into structured forms, namely event triggers and their arguments. An event trigger is the word(s) that most clearly describes an event, such as *"buy"*, which evokes a `transferownership` and an `transfermoney` event in Figure 1. Event arguments are the participants and attributes of an event, and can be tagged as entities at the same time, as demonstrated in Figure 2.

We use the benchmark datasets ACE05 (Doddington et al., 2004) and Rich ERE (Song et al., 2015), both containing consistent annotations for entities, relations, and events, for joint evaluation of multiple IE tasks and in multiple languages (ACE05 in English and Chinese, and ERE in English, Chinese, and Spanish). Table 1 summarizes the relevant statistics of the datasets. The inventory of event types, argument roles, entity types and relation types are listed in Table 2. Despite targeting the same IE tasks, from ACE05 to Rich ERE, the annotation guidelines have shifted towards more sophisticated representations, resulting in more complex structures in Rich ERE (Song et al., 2015). Prominent differences between ACE05 and Rich ERE are:

Dataset	#Event-types	#Argument-roles	#Entity types	#Relation type
ACE05	33	22	7	6
Rich ERE	38	20	15	6

Table 2: Inventory of event types, argument roles, entity types and relation types in ACE05 and Rich ERE.

- **Entities**, and hence event arguments, are more fine-grained in Rich ERE, with 15 entity types, as compared to 7 types in ACE05. In terms of entity spans, ACE05 explicitly marks the head of the entity versus the entire mention, providing the possibility of solving a simpler task for entity extraction and recognizing only the head token as opposed to the full span of the entity in question. This is commonly done for this task in previous work of EE. However, in Rich ERE, the entire string of text is annotated for entity mentions, and heads are only marked explicitly for nominal mentions that are not named entities or pronominal entities.
- **Event triggers** can be double-tagged in Rich ERE, namely one trigger can serve multiple event mentions, giving rise to overlapping events, as shown in Figure 1, while in ACE05, an event trigger only evokes one event. This means that Rich ERE presents a more complex task of event extraction.

We measure the nested instances in ACE05 and Rich ERE as a way to showcase different levels of complexity for extracting entities, relations, and events. More specifically, we quantify nested instances in two versions of each dataset, one using only the head of an entity mention (when it is annotated), and the other with the entire mention text. Following Lin et al. (2020) we dub the version which only marks the head of entities ACE-E$^+$ and

Dataset	Lang	Nesting			#Sents	
		Trg-Trg	Ent-Ent	Trg-Ent	Nested	All
ACE05-E^{+}	en	0	0	4	4	21044
	zh	0	4	9	12	7738
ACE05-E^{++}	en	0	13387	716	5315	21044
	zh	0	10797	252	3748	7738
Rich ERE-E^{+}	en	1066	1329	244	1529	13858
	zh	301	1383	284	1266	10277
	es	485	523	97	712	9273
Rich ERE-E^{++}	en	1063	9453	1517	4277	13858
	zh	301	7303	622	2993	10277
	es	485	5526	854	2614	9273

Table 3: Nesting instances in ACE05 and Rich ERE. Nesting between a pair of event triggers is referred to as `Trg-Trg`; between a pair of entity mentions as `Ent-Ent`, and between an event trigger and an entity as `Trg-Ent`. For both datasets, in the E^{+} version, entity mentions include only heads, while in the E^{++} version, entity mentions include the full text spans.

Rich ERE-E^{+}, and introduce two additional versions of the datasets, dubbed, ACE-E^{++} and Rich ERE-E^{++} which retain the full annotated mention text span. Nesting is measured between any pair of triggers and entities. Note that our notion of nesting subsumes both *overlapping* and *nested* target/entities (Cao et al., 2022), *i.e.* both full and partial overlap of text spans. As shown in Table 3, Rich ERE features many cases of nested triggers, while these are not found in ACE05, due to the aforementioned double-tagging in Rich ERE (see Figure 1); when only considering the head of an entity, ACE05 exhibits very little nesting, but Rich ERE exhibits a considerable amount of nesting within entities, as well as between entity-trigger. The reason for this is that in Rich ERE, only certain nominal mentions are marked with explicit heads; when the full entity mentions are considered, both datasets are heavily nested.

As mentioned above, this work deals with three IE tasks, as exemplified by Figure 2: entities, relations, and events. Given a sentence, our JSEEGraph framework extracts its entity mentions, relations, and event mentions. In addition to event extraction, we thus target two additional IE tasks in our graph-based model:

Entity Extraction is to identify entity mentions from text and classify them into types according to a pre-defined ontology. For example, in Figure 2, *"district"* is an organization (`ORG`) entity.

Relation Extraction aims to assign a relation type to an ordered pair of entity mentions, based on a pre-defined relation ontology. For example, in Figure 2, the relation between `PER` *"officials"* and `ORG` *"district"* is `orgaffiliation`.

4 Graph parsing framework

Our JSEEGraph framework is a text-to-graph parser tailored for EE tasks, additionally with different IE components explicitly encoded in a single graph, as shown in Figure 2. Our framework builds on Samuel and Straka (2020) who developed the PERIN parser in the context of Meaning Representation Parsing (Oepen et al., 2020), as well as (You et al., 2022) who applied PERIN to the task of event extraction. We here extend this parser to the IE graphs shown in Figure 2 in a multilingual setting.

Given a sentence, as the example shown in Figure 3, JSEEGraph encodes the input tokens with the pre-trained language model XLM-R (Conneau et al., 2020) to obtain the contextualized embeddings and further maps the embeddings onto queries; nodes (triggers and entities) are predicted by classifying the queries and anchored to surface tokens via a deep biaffine classifier (Dozat and Manning, 2017); edges are constructed between nodes with two biaffine classifiers, assigning arguments to predicted events and relations to entity pairs. We describe each module in detail in what follows.

4.1 Sentence encoding

We use XLM-R (Conneau et al., 2020) to obtain the contextualized embeddings of the input sequence. To be specific, a trainable weight w_l is used to get a weighted sum of representations of different layers, so the final contextual embedding $\mathbf{e} = \sum_{l=1}^{L} \mathrm{softmax}(w_l)\mathbf{e}_l$ with $\mathbf{e}_l$ as the intermediate output from the l^{th} layer. If an input token consists of multiple subwords, the final contextual embedding will be the weighted sum over all subword embeddings with a learned subword attention.

Each contextual embedding is mapped into $\mathbf{q} = \{\mathbf{q}_1, \cdots, \mathbf{q}_n\}$ queries via a linear layer, and further transformed into hidden features $\mathbf{h} = \{\mathbf{h}_1, \cdots, \mathbf{h}_n\}$ with a stack of transformer encoder layers, which models inter-query dependency with multi-head self-attention.

4.2 Node prediction

The node prediction module consists of a node label classifier and an anchor biaffine attention classifier.

The node label classifier is a linear classifier classifying each query into a node in the graph, and the node label is predicted by a single-layer feedforward network (FNN). If a query is classified

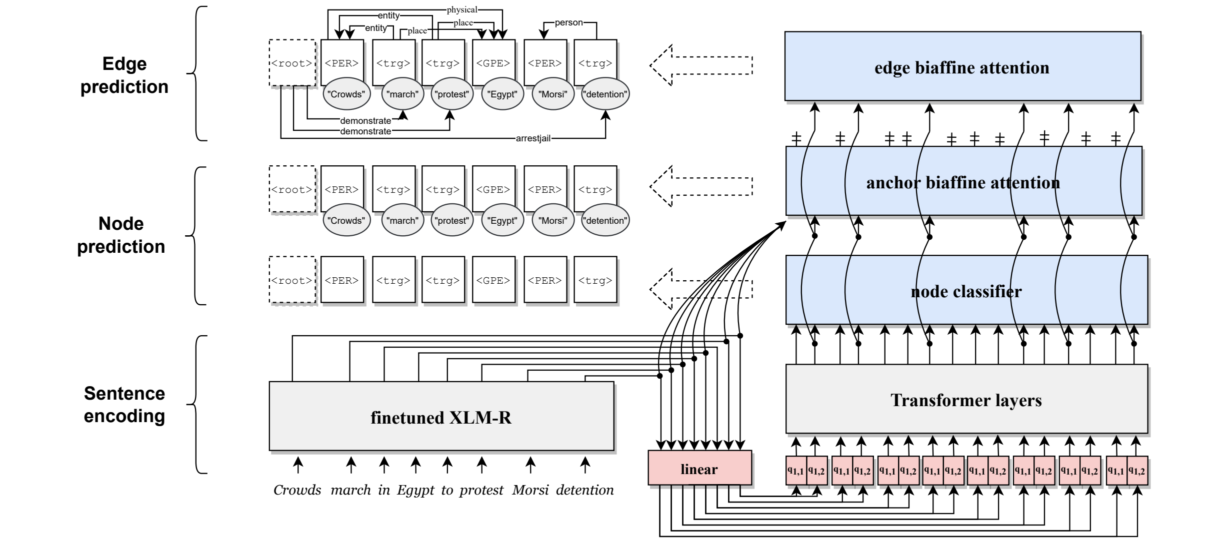

Figure 3: An illustration of our JSEEGraph parsing the sentence *"Crowds march in Egypt to protest Morsi detention."*, example from Rich ERE.

into "null", no node is created from this query.

Node anchoring, as shown in Equation (1), is performed by biaffine attention (Dozat and Manning, 2018) between the contextual embeddings **e** and hidden feature of queries **h**, to map each query (a candidate node) to surface tokens, as shown in Equation (3). For each query, every input token is binary classified into anchor or non-anchor.

$$\mathrm{Bilinear}(\mathrm{X}_1, \mathrm{X}_2) = \mathrm{X}_1^T \mathrm{U} \mathrm{X}_2 \tag{1}$$

$$\mathrm{Biaffine}(\mathrm{X}_1, \mathrm{X}_2) = \mathrm{X}_1^T \mathrm{U} \mathrm{X}_2 + \mathrm{W}(\mathrm{X}_1 \oplus \mathrm{X}_2) + b \tag{2}$$

$$\mathrm{node}^{(\mathrm{anchor})} = \mathrm{Biaffine}^{(\mathrm{anchor})}(\mathbf{h}, \mathbf{e}) \tag{3}$$

Node prediction is complete with queries that are classified into nodes and anchored to corresponding surface tokens. Predicted nodes are either event triggers or entities, labeled as "trigger" or entity type. A dummy node is randomly generated to add to predicted nodes to play the role of `<root>` node, and always holds the first position.

4.3 Edge prediction

Edge prediction between nodes is performed with two deep biaffine classifiers, as in Equation (6), one to predict edge presence between a pair of nodes and the other to predict the corresponding edge label. To construct edges between nodes, only queries from which nodes have been constructed will be used, and the new hidden features is $\mathbf{h}'$, which are further split into two parts with a single-layer FNN, as show in Equation (4) and (5).

$$\mathbf{h}_1'^{(\mathrm{edge})} = \mathrm{FNN}_1^{(\mathrm{edge})}(\mathbf{h}') \tag{4}$$

$$\mathbf{h}_2'^{(\mathrm{edge})} = \mathrm{FNN}_2^{(\mathrm{edge})}(\mathbf{h}') \tag{5}$$

$$\mathrm{edge} = \mathrm{Biaffine}^{(\mathrm{edge})}(\mathbf{h}_1'^{(\mathrm{edge})}, \mathbf{h}_2'^{(\mathrm{edge})}) \tag{6}$$

The edge presence biaffine classifier performs binary classification, deciding whether or not an edge should be constructed between a pair of nodes. The edge label biaffine classifier performs multi-class classification, and the edge label set is the union of argument roles and relation types.

4.4 Constrained decoding

During inference, we apply a set of constraints specifically developed for the correct treatment of event arguments and entity relations based on the graph encoding we define for the information graph (Figure 2): 1) directed edges from the `<root>` node can only connect to a trigger node, and the corresponding edge label is an event type; 2) directed edges from a trigger node to an entity indicates an event argument, with the argument role placed as edge label; 3) directed edges between a pair of entities indicate an entity relation, and the corresponding relation type is assigned to the edge label.

5 Experimental setup

5.1 Data

As mentioned above, we evaluate our system on the benchmark datasets ACE05[4] (LDC2006T06) and Rich ERE[5] (LDC2020T18). As mentioned above, Table 1 summarizes the statistics of the preprocessed datasets.

Following Lin et al. (2020), we keep 33 event types, 22 argument roles, 7 entity types, and 6 relation types for both the English and Chinese parts of ACE05. We follow You et al. (2022) in employing the ACE-E^{++} version of this data, which uses the full text span of entity mentions instead of only the head, as described in section 3 above.

For Rich ERE, we keep 18 out of 38 event types defined in the Rich ERE event ontology [6], 18 out of 21 argument roles [7], 15 entity types, and 6 relation types for English, Chinese, and Spanish. Given no existing data splits, we randomly sample similar proportions of documents for train, development, and testing as the split proportions in ACE05.

5.2 Evaluation metrics

Following previous work (Lin et al., 2020; Nguyen et al., 2021), precision (P), recall (R), F1 scores are reported for the following information elements.

- **Entity** An entity mention is correctly extracted if its offsets and entity type match a reference entity.
- **Relation** A relation is correctly extracted if its relation type, and offsets of both entity mentions match those of reference entities.
- **Event trigger** An event trigger is correctly identified (Trg-I) if its offsets match a reference trigger, and correctly classified (Trg-C) if its event type also matches a reference trigger.
- **Event argument** The evaluation of an argument is conditioned on correct event type prediction; if a predicted argument plays a role in an event that does not match any reference event types, the argument is automatically considered a wrong prediction. An argument is correctly identified (Arg-I) if its offsets match a reference argument, and correctly classified (Arg-C) if its argument role also matches the reference argument.

[4] https://catalog.ldc.upenn.edu/LDC2006T06

[5] https://catalog.ldc.upenn.edu/LDC2020T18

[6] The Rich ERE event ontology defines 38 event types, but for Chinese and Spanish data, only 18 event types are annotated. For consistency, we also use the same 18 event types for the English part.

[7] 3 argument roles for the reduced event types are thus excluded.

5.3 Implementation detail

We adopt multi-lingual training for each dataset for the reported results. Results of monolingual models are listed in Appendix B. Detailed hyperparameter settings and runtimes are included in Appendix A.

5.4 System comparison

We compare our JSEEGraph to the following systems: 1) ONEIE (Lin et al., 2020); 2) GraphIE (Nguyen et al., 2022); 3) FourIE (Nguyen et al., 2021); 4) JMCEE (Xu et al., 2020); 5) EventGraph (You et al., 2022) on the ACE05 dataset. For Rich ERE there is little previous work to compare to; the only previously reported results (Li et al., 2022) for EE only solve the task of argument extraction, using gold entity and trigger information, hence their work is not included in our system comparison.

6 Results and discussion

We here present the results for our JSEEGraph model for the EE task, as well as its performance for the additional IE components: entities and relations, evaluated as described above. We further zoom in on the nested structures identified in Section 3 and assess the performance of our system on these rich structures which have largely been overlooked in previous work on event extraction. We go on to assess the influence of inter-related IE components in an ablation study. Finally we provide an error analysis of our model's predictions.

6.1 Overall performance

As shown in Table 4, on ACE-E^{+}, our overall results align with other systems. Our JSEEGraph results are especially strong for event argument extraction, with an improvement of around 10 percentage points from the best results of the previous best performing systems in our comparison.

On the newly introduced ACE-E^{++}, despite having more complex structures, with a higher degree of nested structures, the results of JSEEGraph on trigger extraction remain stable. We further note that our results on argument, entity, and relation extraction suffers some loss from highly nested entities, which is not surprising.

Model	Trg-I	Trg-C	Arg-I	Arg-C	Entity	Relation
			Dataset: ACE05-E^{+} English			
EventGraph	——	70.0	——	65.4	——	——
GraphIE	——	**74.8**	——	59.9	91.0	**65.4**
ONEIE	75.6	72.8	57.3	54.8	89.6	58.6
FourIE	**76.7**	73.3	59.5	57.5	**91.1**	63.6
JSEEGraph	74.2	71.3	**70.7**	**68.4**	90.7	62.6
JSEEGraph w/o ent&rel	74.8	71.7	67.5	64.6	——	——
			Dataset: ACE05-E^{+} Chinese			
JMCEE	**82.3**	**74.0**	53.7	50	——	——
ONEIE	——	67.7	——	53.2	**89.9**	62.9
FourIE	——	70.3	——	56.1	89.1	**65.9**
JSEEGraph	71.9	69.6	**74.3**	**70.1**	87.4	63.3
JSEEGraph w/o ent&rel	70.5	67.8	69.2	65.5	——	——
			Dataset: ACE05-E^{++} English			
EventGraph	——	**74.0**	——	58.6	——	——
JSEEGraph	73.5	70.0	**62.3**	**59.6**	**85.6**	**56.6**
JSEEGraph w/o ent&rel	**75.0**	71.3	60.3	57.7	——	——
			Dataset: ACE05-E^{++} Chinese			
JSEEGraph	**69.9**	**67.8**	**71.1**	**66.9**	**85.2**	**58.4**
JSEEGraph w/o ent&rel	69.5	67.4	66.5	63.3	——	——

Table 4: Experimental results on ACE05 (F1-score, %). We bold the highest score of each sub-task.

Model	Lang	Trg-I	Trg-C	Arg-I	Arg-C	Entity	Relation
				Dataset: Rich ERE-E^{+}			
JSEEGraph	en	68.6	62.3	59.6	56.2	80.3	53.7
	zh	62.7	59.0	53.1	50.1	78.1	53.2
	es	59.1	51.9	59.9	54.0	74.1	51.8
JSEEGraph w/o ent&rel	en	67.7	62.9	57.9	54.7	——	——
	zh	63.7	60.0	50.7	48.2	——	——
	es	62.3	54.3	57.3	52.5	——	——
				Dataset: Rich ERE-E^{++}			
JSEEGraph	en	67.3	62.7	55.6	52.8	77.9	46.1
	zh	65.2	61.7	51.0	48.7	77.5	54.3
	es	59.7	54.1	59.1	55.4	70.2	49.4
JSEEGraph w/o ent&rel	en	66.4	61.9	52.9	50.7	——	——
	zh	63.2	58.7	49.2	47.2	——	——
	es	57.2	48.9	50.8	46.4	——	——

Table 5: Experimental results on Rich ERE (F1-score).

From Table 5, we find that the scores on Rich ERE are consistently lower compared to those of ACE05. The double-tagging of event triggers described in Section 3 clearly pose a certain level of difficulty for the model to disambiguate events with a shared trigger. Argument and entity extraction also suffers from more fined-grained entity types.

6.2 Nesting

In order to directly evaluate our model's performance on nested instances, we split each test set into nested and non-nested parts and report the corresponding scores, as shown in Table 6[8].

We observe that JSEEGraph is quite robust in tackling nested instances across different IE tasks and languages. On ACE05-E^{++}, more than half of the test data are nested for both English and Chinese, and the results on the nested parts are lower, however consistently comparable with the non-nested parts of the datasets. On Rich ERE-E^{+}, nested instances make up only a small part of the test data, but the results are still comparable to the non-nested part. On Rich ERE-E^{++}, about one third of the test data are nested, results of the nested parts are in fact consistently better for trigger, entity, and relation extraction, but inferior for argument extraction.

[8] ACE05-E^{+} is not included as it lacks sufficient nested instances.

Lang	Nested	#sents	Trg-I	Trg-C	Arg-I	Arg-C	Entity	Relation
				Dataset: ACE05-E^{++}				
en	✓	418	72.1	68.5	59.2	57.0	85.1	57.0
	✗	359	77.0	74.0	73.2	69.0	87.4	47.5
zh	✓	277	72.2	69.7	68.9	65.5	85.4	60.8
	✗	244	57.6	57.6	87.9	77.3	84.5	33.6
				Dataset: Rich ERE-E^{+}				
en	✓	93	81.3	71.6	54.8	51.4	81.3	49.8
	✗	652	61.4	56.9	64.0	60.5	79.8	56.3
zh	✓	101	72.0	66.6	47.5	45.1	79.7	56.0
	✗	382	54.2	52.2	59.5	55.9	77.1	49.9
es	✓	51	78.1	64.7	55.5	52.3	78.4	51.8
	✗	547	49.9	45.5	63.8	55.6	73.1	51.8
				Dataset: Rich ERE-E^{++}				
en	✓	251	75.4	69.2	53.0	50.5	81.0	45.7
	✗	494	46.0	45.3	75.0	70.6	71.0	49.4
zh	✓	197	70.4	67.0	49.0	46.8	80.4	57.2
	✗	286	45.9	41.8	63.9	61.1	69.7	23.3
es	✓	163	66.0	59.3	57.2	53.7	75.2	53.5
	✗	435	47.0	43.0	65.3	61.7	61.4	30.0

Table 6: Experimental results on test data with nesting as compared to without nesting (F1-score, %).

To conclude, JSEEGraph does not suffer considerable performance loss from nesting among different IE elements, and in many cases actually gains in performance from more complex structures, notably for trigger, entity, and relation extraction. It is clear that the system can make use of inter-relations between the different IE elements of the information graph in order to resolve these structures.

6.3 Ablation study

In order to gauge the effect of the joint modeling of entities, events, and relations, we perform an ablation study where we remove the entity and relation information from our information graph, hence only performing the task of event extraction directly from text. In the reduced information graph, node labels for entity types are removed, and relation edges between entities are also removed. We find that event extraction clearly benefits from entity and relation extraction, especially for event argument extraction. As shown in Table 4 and Table 5, when we train our model only for event extraction, the performance on argument extraction drops consistently across different datasets and languages, but the performance on trigger extraction remains quite stable.

6.4 Error analysis

The experimental results show that JSEEGraph has an advantage when it comes to the task of argument extraction. In a manual error analysis we therefore focus on the errors of event trigger extraction. After a manual inspection of our model's predictions on the test data, we find that the errors fall into the following main categories.

Over-predict non-event sentences. Our system tends to be more greedy in extracting event mentions, and wrongly classifies some tokens as event triggers even though the sentence does not contain event annotation. For instance, the sentence *"Anne-Marie will get the couple's 19-room home in New York state"* (from ACE05) does not have annotated events, but our system extracts *"get"* as trigger for a `Transfer-Ownership` event; in this case, however, one could argue that the `Transfer-Ownership` should be annotated.

Under-predict multi-event sentences When a sentence contains multiple event mentions, JSEEGraph sometimes fails to extract all of the event triggers. For example, this sentence *"Kelly , the US assistant secretary for East Asia and Pacific Affairs , arrived in Seoul from Beijing Friday to brief Yoon, the foreign minister"* from ACE05 contains a `Transport` event triggered by *"arrived"* and a `Meet` event triggered by *"brief"*, but our system fails to extract the trigger for the `Meet` event; in this example, it requires a certain level of knowledge to be able to identify *"brief"* as an event trigger, which is beyond the capacity of our model.

Wrong event types In some cases, even though our model successfully identifies an event trigger, it assigns a wrong event type. Some event types can easily be confused with each other. In this sentence from Rich ERE, *"The University of Arkansas campus was buzzing Friday after a student hurt himself when a gun went off in his backpack in the KUAF building"*, an `Injure` event is evoked by *"hurt"*, but our model assigns an event type of `Attack`. Clearly, `Injure` and `Attack` events are one typical case of event types that can be easily confused.

Context beyond sentence This error applies specifically to Rich ERE: even though the annotation of events is on a sentence level, annotators were instructed to take into account the context of the whole article. Our model fails completely when a trigger requires context beyond the sentence. For instance, this sentence *"If Mickey can do it , so can we!"* is taken from an article describing an on-going demonstration in Disney Land, and *"it"* is the trigger for a `demonstrate` event; without the context, our model fails to identify the trigger. These are cases which would require information about event coreference.

7 Conclusion

In this paper, we have proposed JSEEG, a graph-based approach for joint structured event extraction, alongside entity, and relation extraction. We experiment on two benchmark datasets ACE05 and Rich ERE, covering the three languages English, Chinese, and Spanish. We find that our proposed JSEEGraph is robust in solving nested event structures, and is especially strong in event argument extraction. We further demonstrate that it is beneficial to jointly perform EE with other IE tasks, and event argument extraction especially gains from entity extraction.

Limitations

Our work has two main limitations. Firstly, we do not compare our system to previous works on the Rich ERE dataset. This is mainly due to the fact that most work use the light ERE (Song et al., 2015) dataset. We were unfortunately not able to got access to this version of the data[9], which is why no experiments were carried out on it.

Secondly, we only experiment with one language model, the multilingual model XLM-R. As our model is language agnostic, and we aimed to test its performance on datasets in different languages, the choice of a multilingual model was obvious. XLM-R has been chosen based on its good performance in other tasks, and to make our work comparable to previous work (You et al., 2022). However, another approach would be to test our model with a selection of language-specific language models.

Acknowledgements

This work was supported by industry partners and the Research Council of Norway with funding to *MediaFutures: Research Centre for Responsible*

[9]Here we refer to the datasets with LDC codes: *LDC2015E29*, *LDC2015E68*, and *LDC2015E78* for English ERE, and *LDC2015E107* for the Spanish ERE.

Media Technology and Innovation, through the centers for Research-based Innovation scheme, project number 309339.

References

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract meaning representation for sembanking. In *Proceedings of the 7th linguistic annotation workshop and interoperability with discourse*, pages 178–186.

Giannis Bekoulis, Johannes Deleu, Thomas Demeester, and Chris Develder. 2018. Adversarial training for multi-context joint entity and relation extraction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2830–2836, Brussels, Belgium. Association for Computational Linguistics.

Hu Cao, Jingye Li, Fangfang Su, Fei Li, Hao Fei, Shengqiong Wu, Bobo Li, Liang Zhao, and Donghong Ji. 2022. OneEE: A one-stage framework for fast overlapping and nested event extraction. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 1953–1964, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Jason P.C. Chiu and Eric Nichols. 2016. Named entity recognition with bidirectional LSTM-CNNs. *Transactions of the Association for Computational Linguistics*, 4:357–370.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

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

Timothy Dozat and Christopher D. Manning. 2017. Deep biaffine attention for neural dependency parsing. In *International Conference on Learning Representations*.

Timothy Dozat and Christopher D. Manning. 2018. Simpler but more accurate semantic dependency parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 484–490, Melbourne, Australia. Association for Computational Linguistics.

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.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. 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 3816–3830, Online. Association for Computational Linguistics.

Daniel Hershcovich, Omri Abend, and Ari Rappoport. 2017. A transition-based directed acyclic graph parser for UCCA. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1127–1138, Vancouver, Canada. 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 generative event extraction model. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

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

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

Fayuan Li, Weihua Peng, Yuguang Chen, Quan Wang, Lu Pan, Yajuan Lyu, and Yong Zhu. 2020. 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.

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.

Rui Li, Wenlin Zhao, Cheng Yang, and Sen Su. 2022. A dual-expert framework for event argument extraction. In *Proceedings of the 45th International ACM*

SIGIR Conference on Research and Development in Information Retrieval, pages 1110–1121.

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.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. 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 4582–4597, Online. Association for Computational Linguistics.

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

Jian Liu, Yubo Chen, Kang Liu, Wei Bi, and Xiaojiang Liu. 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.

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.

Xiao Liu, Zhunchen Luo, and Heyan Huang. 2018. Jointly multiple events extraction via attention-based graph information aggregation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1247–1256, Brussels, Belgium. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

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.

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. 2022. 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.

Thien Huu Nguyen, Kyunghyun Cho, and Ralph Grishman. 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.

Stephan Oepen, Omri Abend, Lasha Abzianidze, Johan Bos, Jan Hajic, Daniel Hershcovich, Bin Li, Tim O'Gorman, Nianwen Xue, and Daniel Zeman. 2020. MRP 2020: The second shared task on cross-framework and cross-lingual meaning representation parsing. In *Proceedings of the CoNLL 2020 Shared Task: Cross-Framework Meaning Representation Parsing*, pages 1–22, Online. Association for Computational Linguistics.

Stephan Oepen, Marco Kuhlmann, Yusuke Miyao, Daniel Zeman, Silvie Cinkova, Dan Flickinger, Jan Hajic, and Zdenka Uresova. 2015. SemEval 2015 Task 18: Broad-Coverage Semantic Dependency Parsing. *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 915–926.

Stephan Oepen, Marco Kuhlmann, Yusuke Miyao, Daniel Zeman, Dan Flickinger, Jan Hajic, Angelina Ivanova, and Yi Zhang. 2014. SemEval 2014 Task 8: Broad-Coverage Semantic Dependency Parsing. *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 63–72.

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*.

David Samuel and Milan Straka. 2020. ÚFAL at MRP 2020: Permutation-invariant semantic parsing in PERIN. In *Proceedings of the CoNLL 2020 Shared Task: Cross-Framework Meaning Representation Parsing*, pages 53–64, Online. Association for Computational Linguistics.

Jiawei Sheng, Shu Guo, Bowen Yu, Qian Li, Yiming Hei, Lihong Wang, Tingwen Liu, and Hongbo Xu. 2021. CasEE: A joint learning framework with cascade decoding for overlapping event extraction. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 164–174, Online. Association for Computational Linguistics.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235, Online. Association for Computational Linguistics.

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

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.

Nuo Xu, Haihua Xie, and Dongyan Zhao. 2020. A novel joint framework for multiple Chinese events extraction. In *Proceedings of the 19th Chinese National Conference on Computational Linguistics*, pages 950–961, Haikou, China. Chinese Information Processing Society of China.

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.

Huiling You, David Samuel, Samia Touileb, and Lilja Øvrelid. 2022. Eventgraph: Event extraction as semantic graph parsing. In *Proceedings of CASE: The 5th Workshop on Challenges and Applications of Automated Extraction of Socio-political Events from Text*.

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.

A Training detail

We use the large version of XLM-R available on HuggingFace `transformers`[10] for obtaining contextual embeddings of the input sequence. We use the same hyperparameter configuration for all our models, as shown in Table 7, and weights are optimized with AdamW (Loshchilov and Hutter, 2019) following a warmed-up cosine learning rate schedule.

Hyperparameter	JSEEGraph
batch_size	16
beta_2	0.98
decoder_learning_rate	1.0e-4
decoder_weight_decay	1.2e-6
dropout_transformer	0.25
dropout_transformer_attention	0.1
encoder	*"xlm-roberta-large"*
encoder_learning_rate	4.0e-6
encoder_weight_decay	0.1
epochs	110
hidden_size_anchor	256
hidden_size_edge_label	256
hidden_size_edge_presence	256
n_transformer_layers	3
query_length	2
warmup_steps	1 000

Table 7: Hyperparameter setting for our system, and we use the same configuration for all models.

The training was done on a single node of Nvidia RTX3090 GPU. The runtimes and sizes (including the pretrained XLM-R) of the multilingual models for each dataset are listed in Table 8,

Dataset	Runtime	Model size
ACE05-E^+	27:52 h	343.8 M
ACE05-E^{++}	27:25 h	343.8 M
Rich ERE-E^+	33:13 h	344.6 M
Rich ERE-E^{++}	32:16 h	344.6 M

Table 8: The training times and model sizes (number of trainable weights) of all our experiments.

Lang	Trg-I	Trg-C	Arg-I	Arg-C	Entity	Relation
Dataset: ACE05-E^+						
en	73.1	70.0	68.5	65.4	90.4	61.4
	73.2	69.8	66.7	64.2	—	—
zh	69.2	67.0	71.4	67.8	85.6	60.2
	64.8	62.6	62.5	59.3	—	—
Dataset: ACE05-E^{++}						
en	73.8	70.3	63.7	60.6	85.3	55.4
	72.7	69.9	58.9	56.3	—	—
zh	66.7	64.5	66.0	63.1	82.1	53.7
	66.0	64.3	62.2	58.4	—	—
Dataset: Rich ERE-E^+						
en	65.3	60.5	59.8	56.1	80.6	53.6
	68.7	62.4	56.0	52.8	—	—
zh	62.3	57.7	53.9	50.2	78.3	54.5
	62.4	59.0	48.2	46.3	—	—
es	54.2	47.9	52.5	46.7	72.9	44.7
	56.7	49.7	51.3	47.3	—	—
Dataset: Rich ERE-E^{++}						
en	66.9	60.4	54.6	52.1	76.3	42.1
	66.2	59.2	49.5	46.8	—	—
zh	63.6	60.2	47.1	44.8	76.2	51.5
	60.5	57.2	41.5	38.8	—	—
es	54.4	48.9	47.2	43.2	68.2	43.0
	54.5	48.4	35.7	32.1	—	—

Table 9: Experimental results of monolingual models (F1-score, %)

B Monolingual training results

Apart from multilingual training, we also train two monolingual models for each language, one for joint event extraction with entity and relation and the for event extraction only. Results of monolingual models are summerized in Table 9.

[10] https://huggingface.co/docs/transformers/index