DEGREE: A Data-Efficient Generation-Based Event Extraction Model

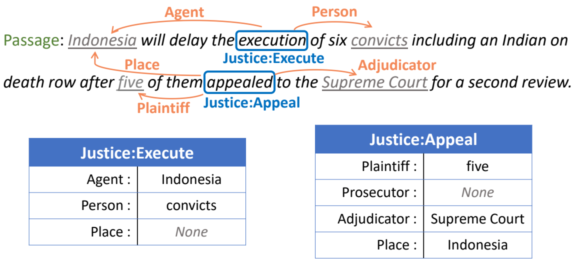
I-Hung Hsu[*†] **Kuan-Hao Huang**[*‡] **Elizabeth Boschee**[†] **Scott Miller**[†]
Premkumar Natarajan[†] **Kai-Wei Chang**[‡] **Nanyun Peng**[†‡]
[†]Information Science Institute, University of Southern California
[‡]Computer Science Department, University of California, Los Angeles
`{ihunghsu, boschee, smiller, pnataraj}@isi.edu`
`{khhuang, kwchang, violetpeng}@cs.ucla.edu`

Abstract

Event extraction requires high-quality expert human annotations, which are usually expensive. Therefore, learning a *data-efficient* event extraction model that can be trained with *only a few* labeled examples has become a crucial challenge. In this paper, we focus on *low-resource end-to-end* event extraction and propose DEGREE, a data-efficient model that formulates event extraction as a conditional generation problem. Given a passage and a manually designed prompt, DEGREE learns to summarize the events mentioned in the passage into a natural sentence that follows a predefined pattern. The final event predictions are then extracted from the generated sentence with a deterministic algorithm. DEGREE has three advantages to learn well with less training data. First, our designed prompts provide semantic guidance for DEGREE to leverage *label semantics* and thus better capture the event arguments. Moreover, DEGREE is capable of using additional *weakly-supervised* information, such as the description of events encoded in the prompts. Finally, DEGREE learns triggers and arguments jointly in an *end-to-end manner*, which encourages the model to better utilize the shared knowledge and dependencies among them. Our experimental results demonstrate the strong performance of DEGREE for low-resource event extraction.

1 Introduction

Event extraction (EE) aims to extract events, each of which consists of a trigger and several participants (arguments) with their specific roles, from a given passage. For example, in Figure 1, a *Justice:Execute* event is triggered by the word *"execution"* and this event contains three argument roles, including an *Agent* (*Indonesia*) who carries out the execution, a *Person* who is executed (*convicts*), and a *Place* where the event occurs (not mentioned in the passage). Previous work (Yang et al., 2019a; Fincke et al., 2021) usually divides EE into two subtasks: (1) **event detection**, which identifies event triggers and their types, and (2) **event argument extraction**, which extracts the arguments and their roles for given event triggers. EE has been shown to benefit a wide range of applications, e.g., building knowledge graphs (Zhang et al., 2020), question answering (Berant et al., 2014; Han et al., 2021), and other downstream studies (Han et al., 2019a; Hogenboom et al., 2016; Sun and Peng, 2021).


Agent
Person
Passage: Indonesia will delay the execution of six convicts including an Indian on
Place
Justice:Execute
Adjudicator
death row after five of them appealed to the Supreme Court for a second review.
Plaintiff
Justice:Appeal

Justice:Execute	
Agent :	Indonesia
Person :	convicts
Place :	None

Justice:Appeal	
Plaintiff :	five
Prosecutor :	None
Adjudicator :	Supreme Court
Place :	Indonesia

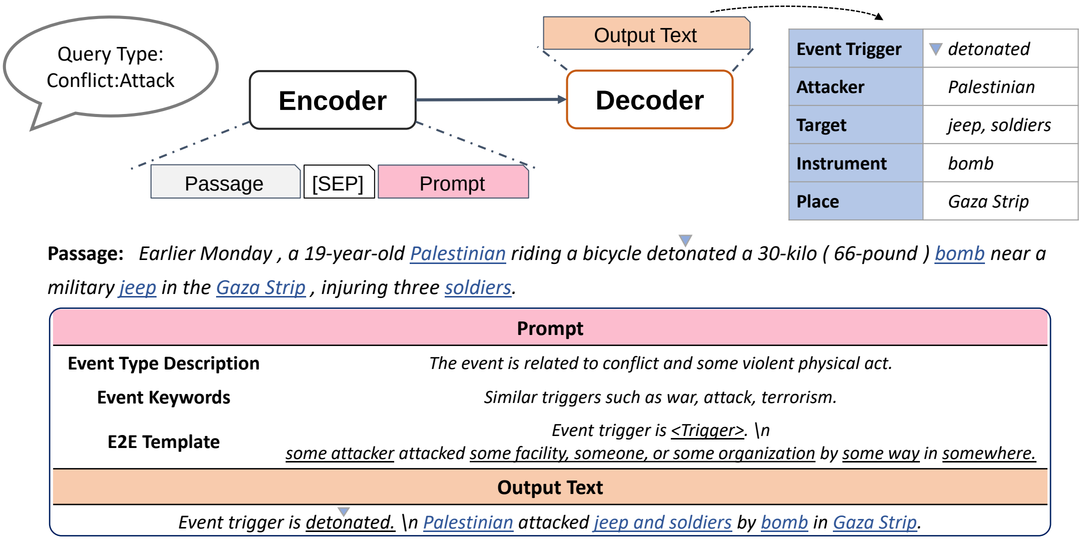
Figure 1: Two examples of events (*Justice:Execute* and *Justice:Appeal*) extracted from the given passage.

Most prior works on EE rely on a large amount of annotated data for training (Nguyen and Grishman, 2015; Nguyen et al., 2016; Han et al., 2019b; Du and Cardie, 2020; Huang et al., 2020; Huang and Peng, 2021; Paolini et al., 2021). However, high-quality event annotations are expensive to obtain. For example, the ACE 2005 corpus (Doddington et al., 2004), one of the most widely used EE datasets, requires two rounds of annotations by linguistics experts. The high annotation costs make these models hard to be extended to new domains and new event types. Therefore, how to learn a *data-efficient* EE model trained with *only a few* annotated examples is a crucial challenge.

In this paper, we focus on *low-resource* event extraction, where only a small amount of training examples are available for training. We propose DEGREE (**D**ata-**E**fficient **G**ene**R**ation-Based **E**vent **E**xtraction), a generation-based model that takes a passage and a manually designed prompt as the input, and learns to summarize the passage into a

[*] The authors contribute equally.

Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 1890 - 1908
July 10-15, 2022 ©2022 Association for Computational Linguistics



Event Trigger	detonated
Attacker	Palestinian
Target	jeep, soldiers
Instrument	bomb
Place	Gaza Strip

Passage: *Earlier Monday , a 19-year-old Palestinian riding a bicycle detonated a 30-kilo (66-pound) bomb near a military jeep in the Gaza Strip , injuring three soldiers.*

Prompt	
Event Type Description	*The event is related to conflict and some violent physical act.*
Event Keywords	*Similar triggers such as war, attack, terrorism.*
E2E Template	*Event trigger is <Trigger>. \n* *some attacker attacked some facility, someone, or some organization by some way in somewhere.*
Output Text	
Event trigger is detonated. \n Palestinian attacked jeep and soldiers by bomb in Gaza Strip.	

Figure 2: An illustration of DEGREE for predicting a *Contact:Attack* event. The input of DEGREE consists of the given passage and our design prompt that contains an event type description, event keywords, and a E2E template. DEGREE is trained to generate an output to fill in the placeholders (underlined words) in the E2E template with triggers and arguments. The final event prediction is then decoded from the generated output.

natural sentence following a predefined template, as illustrated in Figure 2. The event triggers and arguments can then be extracted from the generated sentence by using a deterministic algorithm.

DEGREE enjoys the following advantages to learn well with less training data. First, the framework provides *label semantics* via the designed template in the prompts. As the example in Figure 2 shows, the word *"somewhere"* in the prompt guides the model to predict words being similar to *location* for the role *Place*. In addition, the sentence structure of the template and the word *"attacked"* depict the semantic relation between the role *Attacker* and the role *Target*. With these kinds of guidance, DEGREE can make more accurate predictions with less training examples. Second, the prompts can incorporate additional weak-supervision signal about the task, such as the description of the event and similar keywords. These resources are usually readily available. For example, in our experiments, we take the information from the annotation guideline, which is provided along with the dataset. This information facilitates DEGREE to learn under a low-resource situation. Finally, DEGREE is designed for end-to-end event extraction and can solve event detection and event argument extraction at the same time. Leveraging the shared knowledge and dependencies between the two tasks makes our model more data-efficient.

Existing works on EE usually have only one or two of above-mentioned advantages. For example, previous classification-based models (Nguyen et al., 2016; Wang et al., 2019; Yang et al., 2019b; Wadden et al., 2019; Lin et al., 2020) can hardly encode label semantics and other weak supervision signals. Recently proposed generation-based models for event extraction solved the problem in a pipeline fashion; therefore, they cannot leverage shared knowledge between subtasks (Paolini et al., 2021; Li et al., 2021). Furthermore, their generated outputs are not natural sentences, which hinders the utilization of label semantics (Paolini et al., 2021; Lu et al., 2021). As a result, our model DEGREE can achieve significantly better performance than prior approaches on low-resource event extraction, as we will demonstrate in Section 3.

Our contributions can be summarized as follows:

- We propose DEGREE, a generation-based event extraction model that learns well with less data by better incorporating label semantics and shared knowledge between subtasks (Section 2).
- Experiments on ACE 2005 (Doddington et al., 2004) and ERE-EN (Song et al., 2015) demonstrate the strong performance of DEGREE in the low-resource setting (Section 3).
- We present comprehensive ablation studies in both the low-resource and the high-resource setting to better understand the strengths and weaknesses of our model (Section 4).

Our code and models can be found at `https://github.com/PlusLabNLP/DEGREE`.

2 Data-Efficient Event Extraction

We introduce DEGREE, a generation-based model for low-resource event extraction. Unlike previous works (Yang et al., 2019a; Li et al., 2021), which separate event extraction into two pipelined tasks (event detection and event argument extraction), DEGREE is designed for the end-to-end event extraction and predict event triggers and arguments at the same time.

2.1 The DEGREE Model

We formulate event extraction as a conditional generation problem. As illustrated in Figure 2, given a passage and our designed prompt, DEGREE generates an output following a particular format. The final predictions of event triggers and argument roles can be then parsed from the generated output with a deterministic algorithm. Compared to previous classification-based models (Wang et al., 2019; Yang et al., 2019b; Wadden et al., 2019; Lin et al., 2020), the generation framework provides a flexible way to include additional information and guidance. By designing appropriate prompts, we encourage DEGREE to better capture the dependencies between entities and, therefore, to reduce the number of training examples needed.

The desired prompt not only provides information but also defines the output format. As shown in Figure 2, it contains the following components:

- **Event type definition** describes the definition for the given event type.[1] For example, *"The event is related to conflict and some violent physical act."* describes a *Conflict:Attack* event.
- **Event keywords** presents some words that are semantically related to the given event type. For example, *war*, *attack*, and *terrorism* are three event keywords for the *Conflict:Attack* event. In practice, we collect three words that appear as the triggers in the example sentences from the annotation guidelines.
- **E2E template** defines the expected output format and can be separated into two parts. The first part is called ED template, which is designed as *"Event trigger is <Trigger>"*, where *"<Trigger>"* is a special token serving as a placeholder. The second part is the EAE template, which differs based on the given event type. For example, in Figure 2, the EAE template for a *Conflict:Attack* event is *"some attacker attacked some facility, someone, or some organization by some way in somewhere"*. Each underlined string starting with *"some-"* serves as a placeholder corresponding to an argument role for a *Conflict:Attack* event. For instance, *"some way"* corresponds to the role *Instrument* and *"somewhere"* corresponds to the role *Place*. Notice that every event type has its own EAE template. We list three EAE templates in Table 1. The full list of EAE templates and the construction details can be found in Appendix A.

2.2 Training

The training objective of DEGREE is to generate an output that replaces the placeholders in E2E template with the gold labels. Take Figure 2 as an example, DEGREE is expected to replace *"<Trigger>"* with the gold trigger (*detonated*), replace *"some attacker"* with the gold argument for role *Attacker* (*Palestinian*), and replace *"some way"* with the gold argument for role *Instrument* (*bomb*). If there are multiple arguments for the same role, they are concatenated with *"and"*; if there is no predicted argument for one role, the model should keep the corresponding placeholder (i.e, *"some-"* in the E2E template). For the case that there are multiple triggers for the given event type in the input passage, DEGREE is trained to generate the output text that contains multiple E2E template such that each E2E template corresponds to one trigger and its argument roles. The hyperparameter settings are detailed in Appendix B.

2.3 Inference

We enumerate all event types and generate an output for each event type. After we obtain the generated sentences, we compare the outputs with E2E template to determine the predicted triggers and arguments in string format. Finally, we apply string matching to convert the predicted string to span offsets in the passage. If the predicted string appears in the passage multiple times, we choose all span offsets that match for trigger predictions and choose the one closest to the given trigger span for argument predictions.

2.4 Discussion

Notice that the E2E template plays an important role for DEGREE. First, it serves as the control signal and defines the expected output format. Second,

[1] The definition can be derived from the annotation guidelines, which are provided along with the datasets.

Event Type	**EAE Template**
Life:Divorce	somebody divorced in somewhere.
Transaction:Transfer-Ownership	someone got something from some seller in somewhere.
Justice:Sue	somebody was sued by some other in somewhere. The adjudication was judged by some adjudicator.

Table 1: Three examples of EAE templates for the ACE 2005 corpus.

it provides label semantics to help DEGREE make accurate predictions. Those placeholders (words starting with *"some-"*) in the E2E template give DEGREE some hints about the entity types of arguments. For instance, when seeing *"somewhere"*, DEGREE tends to generate a location rather than a person. In addition, the words other than *"some-"* describe the relationships between roles. For example, DEGREE knows the relationship between the role *Attacker* and the role *Target* (who is attacking and who is attacked) due to E2E template. This guidance helps DEGREE learn the dependencies between entities.

Unlike previous generation-based approaches (Paolini et al., 2021; Li et al., 2020; Huang et al., 2021), we intentionally write E2E templates in natural sentences. This not only uses label semantics better but also makes the model easier to leverage the knowledge from the pre-trained decoder. In Section 4, we will provide experiments to demonstrate the advantage of using natural sentences.

Cost of template constructing. DEGREE does require human effort to design the templates; however, writing those templates is much easier and more effortless than collecting complicated event annotations. As shown in Table 1, we keep the EAE templates as simple and short as possible. Therefore, it takes only about one minute for people who are not linguistic experts to compose a template. In fact, several prior works (Liu et al., 2020; Du and Cardie, 2020; Li et al., 2020) also use constructed templates as weakly-supervised signals to improve models. In Section 4, we will study how different templates affect the performance.

Efficiency Considerations. DEGREE requires to enumerate all event types during inference, which could cause efficiency considerations when extending to applications that contain many event types. This issue is minor for our experiments on the two datasets (ACE 2005 and ERE-EN), which are relatively small scales in terms of the number of event types. Due to the high cost of annotations, there is hardly any public datasets for end-to-end event extraction on a large scale,[2] and we cannot provide a more thorough studies when the experiments scale up. We leave the work on benchmarking and improving the efficiency of DEGREE in the scenario considering more diverse and comprehensive types of events as future work.

2.5 DEGREE in Pipeline Framework

DEGREE is flexible and can be easily modified to DEGREE(PIPE), which first focuses event detection (ED) and then solves event argument extraction (EAE). DEGREE(PIPE) consists of two models: (1) DEGREE(ED), which aims to exact event triggers for the given event type, and (2) DEGREE(EAE), which identifies argument roles for the given event type and the corresponding trigger. DEGREE(ED) and DEGREE(EAE) are similar to DEGREE but with different prompts and output formats. We describe the difference as follows.

DEGREE(ED). The prompt of DEGREE(ED) contains the following components:

- **Event type definition** is the same as the ones for DEGREE.
- **Event keywords** is the same as the one for DEGREE.
- **ED template** is designed as *"Event trigger is <Trigger>"*, which is actually the first part of the E2E template.

Similar to DEGREE, the objective of DEGREE(ED) is to generate an output that replaces *"<Trigger>"* in the ED template with event triggers.

DEGREE(EAE). The prompt of DEGREE(EAE) contains the following components:

- **Event type definition** is the same as the one for DEGREE.
- **Query trigger** is a string that indicates the trigger word for the given event type. For example, *"The event trigger word is detonated"* points out that *"detonated"* is the given trigger.

[2] To the best of our knowledge, MAVEN (Wang et al., 2020) is the only publicly available large-scale event dataset. However, the dataset only focuses on event detection without considering event arguments.

- **EAE template** is an event-type-specific template mentioned previously. It is actually the second part of E2E template.

Similar to DEGREE, the goal for DEGREE(EAE) is to generate an outputs that replace the placeholders in EAE template with event arguments.

In Section 3, we will compare DEGREE with DEGREE(PIPE) to study the benefit of dealing with event extraction in an end-to-end manner under the low-resource setting.

3 Experiments

We conduct experiments for *low-resource* event extraction to study how DEGREE performs.

3.1 Experimental Settings

Datasets. We consider ACE 2005 (Doddington et al., 2004) and follow the pre-processing in Wadden et al. (2019) and Lin et al. (2020), resulting in two variants: **ACE05-E** and **ACE05-E$^+$**. Both contain 33 event types and 22 argument roles. In addition, we consider **ERE-EN** (Song et al., 2015) and adopt the pre-processing in Lin et al. (2020), which keeps 38 event types and 21 argument roles.

Data split for low-resource setting. We generate different proportions (1%, 2%, 3%, 5%, 10%, 20%, 30%, and 50%) of training data to study the influence of the size of the training set and use the original development set and test set for evaluation. Appendix C lists more details about the split generation process and the data statistics.

Evaluation metrics. We consider the same criteria in prior works (Wadden et al., 2019; Lin et al., 2020). (1) **Trigger F1-score**: an trigger is correctly identified (Tri-I) if its offset matches the gold one; it is correctly classified (Tri-C) if its event type also matches the gold one. (2) **Argument F1-score**: an argument is correctly identified (Arg-I) if its offset and event type match the gold ones; it is correctly classified (Arg-C) if its role matches as well.

Compared baselines. We consider the following classification-based models: (1) **OneIE** (Lin et al., 2020), the current state-of-the-art (SOTA) EE model trained with designed global features. (2) **BERT_QA** (Du and Cardie, 2020), which views EE tasks as a sequence of extractive question answering problems. Since it learns a classifier to indicate the position of the predicted span, we view it as a classification model. We also consider the following generation-based models: (3) **TANL** (Paolini et al., 2021), which treats EE tasks as translation tasks between augmented natural languages. (4) **Text2Event** (Lu et al., 2021), a sequence-to-structure model that converts the input passage to a tree-like event structure. Note that the outputs of both generation-based baselines are not natural sentences. Therefore, it is more difficult for them to utilize the label semantics. All the implementation details can be found in Appendix D. It is worth noting that we train OneIE with named entity annotations, as the original papers suggest, while the other models are trained without entity annotations.

3.2 Main Results

Table 2 shows the trigger classification F1-scores and the argument classification F1-scores in three data sets with different proportions of training data. The results are visualized in Figure 3. Since our task is *end-to-end* event extraction, the argument classification F1-score is the more important metric that we considered when comparing models.

From the figure and the table, we can observe that both DEGREE and DEGREE(PIPE) outperform all other baselines when using less than 10% of the training data. The performance gap becomes much more significant under the extremely low data situation. For example, when only 1% of the training data is available, DEGREE and DEGREE(PIPE) achieve more than 15 points of improvement in trigger classification F1 scores and more than 5 points in argument classification F1 scores. This demonstrates the effectiveness of our design. The generation-based model with carefully designed prompts is able to utilize the label semantics and the additional weakly supervised signals, thus helping learning under the low-resource regime.

Another interesting finding is that DEGREE and DEGREE(PIPE) seem to be more beneficial for predicting arguments than for predicting triggers. For example, OneIE, the strongest baseline, requires 20% of training data to achieve competitive performance on trigger prediction to DEGREE and DEGREE(PIPE); however, it requires about 50% of training data to achieve competitive performance in predicting arguments. The reason is that the ability to capture dependencies becomes more important for argument prediction than trigger prediction since arguments are usually strongly dependent on each other compared to triggers. Therefore, the improvements of our models for argument prediction

Trigger Classification F1-Score (%)																			
Model	Type	ACE05-E						ACE05-E$^+$						ERE-EN					
		1%	3%	5%	10%	20%	30%	1%	3%	5%	10%	20%	30%	1%	3%	5%	10%	20%	30%
BERT_QA	Cls	20.5	40.2	42.5	50.1	61.5	61.3	-	-	-	-	-	-	-	-	-	-	-	-
OneIE	Cls	38.5	52.4	59.3	61.5	67.6	67.4	39.0	52.5	60.6	58.1	66.5	66.4	11.0	36.9	46.7	48.8	**51.8**	**53.5**
Text2Event	Gen	14.2	35.2	46.4	47.0	55.6	60.7	15.7	38.4	43.9	46.3	56.5	62.0	6.3	25.6	33.5	42.4	46.7	50.1
TANL	Gen	34.1	48.1	53.4	54.8	61.8	61.6	30.3	50.9	53.1	55.7	60.8	61.7	5.7	30.8	43.4	45.9	49.0	49.3
DEGREE(PIPE)	Gen	55.1	**62.8**	63.8	**66.1**	64.4	64.4	**56.4**	62.5	61.1	62.3	62.5	**67.1**	**32.7**	44.5	41.6	50.6	51.1	**53.5**
DEGREE	Gen	**55.4**	62.1	**65.8**	65.8	**68.3**	**68.2**	49.5	**63.5**	**62.3**	**68.5**	**67.6**	66.9	27.9	**45.5**	**47.0**	**53.0**	51.7	**53.5**

Argument Classification F1-Score (%)																			
Model	Type	ACE05-E						ACE05-E$^+$						ERE-EN					
		1%	3%	5%	10%	20%	30%	1%	3%	5%	10%	20%	30%	1%	3%	5%	10%	20%	30%
BERT_QA	Cls	4.7	14.5	26.9	27.6	36.7	38.8	-	-	-	-	-	-	-	-	-	-	-	-
OneIE	Cls	9.4	22.0	26.8	26.8	42.7	47.8	10.4	20.6	29.7	35.5	46.7	48.0	2.6	20.3	29.7	35.1	40.7	43.0
Text2Event	Gen	3.9	12.2	19.1	24.9	32.3	39.2	5.7	16.5	21.3	26.4	35.2	42.1	2.3	15.2	23.6	28.7	35.7	38.7
TANL	Gen	8.5	17.2	24.7	29.0	34.0	39.2	8.6	22.3	30.4	29.2	34.6	39.0	1.4	20.2	29.5	30.1	35.6	36.9
DEGREE(PIPE)	Gen	13.1	26.1	27.6	**42.1**	40.7	44.0	16.0	26.4	29.9	39.5	41.3	48.5	12.2	**29.7**	31.4	39.4	41.9	42.2
DEGREE	Gen	**21.7**	**30.1**	**35.5**	41.6	**46.2**	**48.7**	**18.7**	**34.0**	**35.7**	**43.6**	**48.9**	**51.2**	**14.5**	28.9	**33.4**	**41.7**	**42.9**	**45.5**

Table 2: Trigger classification F1-scores and argument classification F1-scores for low-resource event extraction. Highest scores are in bold and the second best scores are underlined. "Cls" and "Gen" represent classification-based models and generation-based models, respectively. If the model is a pipelined model, then its argument predictions are based on its predicted triggers. DEGREE achieves a much better performance than other baselines. The performance gap becomes more significant for the extremely low-resource situation.

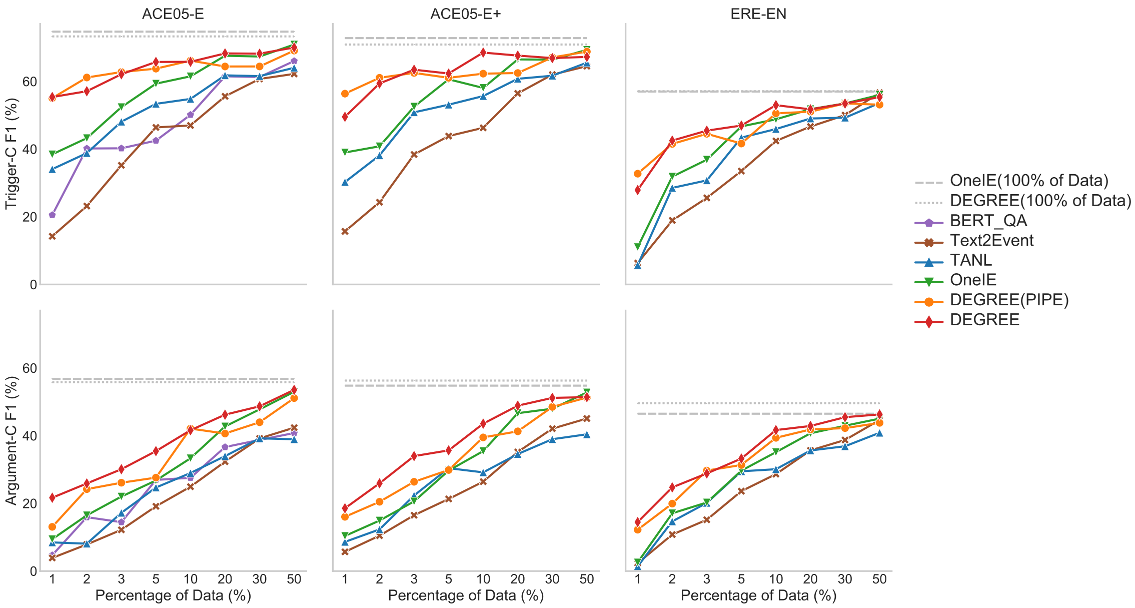


Figure 3: Trigger classification F1-scores and argument classification F1-scores for low-resource event extraction. DEGREE achieves a much better performance than other baselines. The performance gap becomes more significant for the extremely low-resource situation.

are more significant.

Furthermore, we observe that DEGREE is slightly better than DEGREE(PIPE) under the low-resource setting. This provides empirical evidence on the benefit of jointly predicting triggers and arguments in a low-resource setting.

Finally, we perform additional experiments on few-shot and zero-shot experiments. The results can be found in Appendix E.

3.3 High-Resource Event Extraction

Although we focus on data-efficient learning for low-resource event extraction, to better understand the advantages and disadvantages of our model, we additionally study DEGREE in the high-resource setting for controlled comparisons.

Compared baselines. In addition to the EE models mentioned above: **OneIE** (Lin et al., 2020), **BERT_QA** (Du and Cardie, 2020), **TANL** (Paolini et al., 2021), and **Text2Event** (Lu et al., 2021), we

Model	Type	ACE05-E		ACE05-E$^+$		ERE-EN	
		Tri-C	Arg-C	Tri-C	Arg-C	Tri-C	Arg-C
dbRNN*	Cls	69.6	50.1	-	-	-	-
DyGIE++	Cls	70.0	50.0	-	-	-	-
Joint3EE*	Cls	69.8	52.1	-	-	-	-
BERT_QA*	Cls	72.4	53.3	-	-	-	-
MQAEE*	Cls	71.7	53.4	-	-	-	-
OneIE*	Cls	**74.7**	**56.8**	**72.8**	54.8	57.0	46.5
TANL	Gen	68.4	47.6	68.6	46.0	54.7	43.2
Text2Event*	Gen	71.9	53.8	71.8	54.4	**59.4**	48.3
BART-Gen*	Gen	71.1	53.7	-	-	-	-
Degree(Pipe)	Gen	72.2	55.8	71.7	**56.8**	57.8	**50.4**
Degree	Gen	73.3	55.8	70.9	56.3	57.1	49.6

Table 3: Results for high-resource event extraction. Highest scores are in bold and the second best scores are underlined. *We report the numbers from the original paper. Degree has a competitive performance to the SOTA model (OneIE) and outperform other baselines.

Model	Type	ACE05-E		ACE05-E$^+$		ERE-EN	
		Arg-I	Arg-C	Arg-I	Arg-C	Arg-I	Arg-C
DyGIE++	Cls	66.2	60.7	-	-	-	-
BERT_QA*	Cls	68.2	65.4	-	-	-	-
OneIE	Cls	73.2	69.3	73.3	70.6	75.3	70.0
TANL	Gen	65.9	61.0	66.3	62.3	75.6	69.6
BART-Gen*	Gen	69.9	66.7	-	-	-	-
Degree(EAE)	Gen	**76.0**	**73.5**	**75.2**	**73.0**	**80.2**	**76.3**

Table 4: Results for high-resource event argument extraction. Models predict arguments based on the given gold triggers. Best scores are in bold. *We report the numbers from the original paper. Degree(EAE) achieves a new state-of-the-art performance on event argument extraction.

also consider the following baselines focusing on the high-resource setting. **dbRNN** (Sha et al., 2018) is classification-based model that adds dependency bridges for event extraction. **DyGIE++** (Wadden et al., 2019) is a classification-based model with span graph propagation technique. **Joint3EE** (Nguyen and Nguyen, 2019) is a classification-based model jointly trained with annotations of entity, trigger, and argument. **MQAEE** (Li et al., 2020) converts EE to a series of question answering problems for argument extraction . **BART-Gen** (Li et al., 2021) is a generation-based model focusing on *only* event argument extraction.[3] Appendix D shows the implementation details for the baselines.

Results for event extraction. Table 3 shows the results of high-resource event extraction. In terms of trigger predictions (Tri-C), Degree and Degree(Pipe) outperforms all the baselines except for OneIE, the current state-of-the-art model. For argument predictions (Arg-C), our models have slightly better performance than OneIE in two out of the three datasets. When enough training examples are available, models can learn more sophisticated features from data, which do not necessarily follow the learned dependencies. Therefore, the advantage of Degree over Degree(Pipe) becomes less obvious. This result justifies our hypothesis that Degree has better performance for the *low-resource setting* because of its ability to better capture dependencies.

Model	10% Data		100% Data	
	Tri-I	Tri-C	Tri-I	Tri-C
Full Degree(ED)	**69.3**	**66.1**	**75.4**	**72.2**
- w/o Event type definition	67.9	64.4	73.5	70.1
- w/o ED template	68.8	65.8	74.0	70.5
- w/o Event keywords	68.2	64.0	73.5	69.1
- only Event type definition	66.3	63.5	72.6	68.9
- only Event keywords	69.2	63.8	70.8	66.2

Table 5: Ablation study for the components in the prompt on event detection with ACE05-E.

Results for event argument extraction. In Table 4, we additionally study the performance for event argument extraction task, where the model makes argument predictions *with the gold trigger provided*. Interestingly, Degree(EAE) achieves pretty strong performance and outperforms other baselines with a large margin. Combining the results in Table 3, we hypothesize that event argument extraction is a more challenging task than event trigger detection and it requires more training examples to learn well. Hence, our proposed model, which takes the advantage of using label semantics to better capture dependencies, achieves a new state-of-the-art for event argument extraction.

4 Ablation Studies

In this section, we present comprehensive ablation studies to justify our design. To better understand the contribution of each component in the designed prompt and their effects on the different tasks, we ablate Degree(EAE) and Degree(ED) for both low-resource and high-resource situations.

Impacts of components in prompts. Table 5 lists the performance changes when removing the components in the prompts for event detection on ACE05-E. The performance decreases whenever removing any one of event type definition, event keywords, and ED template. The results suggest that three components are all necessary.

Table 6 demonstrates how different components in prompts affect the performance of event argument extraction on ACE05-E. Removing

[3] We follow the original paper and use TapKey as their event detection model.

Model	10% Data		100% Data	
	Arg-I	Arg-C	Arg-I	Arg-C
Full DEGREE(EAE)	**63.3**	**57.3**	**76.0**	**73.5**
- w/o Event type definition	60.3	54.4	74.5	71.1
- w/o EAE template	57.0	51.9	73.8	70.4
- w/o Query trigger	55.2	49.9	71.4	69.0
- only Query trigger	51.9	48.1	71.2	69.4
- only EAE template	51.2	46.9	71.4	68.6
- only Event type definition	46.7	42.3	71.4	68.2

Table 6: Ablation study for the components in the prompt on event argument extraction with ACE05-E.

any one of event type definition, query trigger, and EAE template leads to performance drops, which validates their necessity. We observe that query trigger plays the most important role among the three and when less training data is given, the superiority of leveraging any of these weakly-supervised signal becomes more obvious.

Effects of different template designs. To verify the importance of using natural sentences as outputs, we study three variants of EAE templates:

- **Natural sentence.** Our proposed templates described in Section 2, e.g., "*somebody was born in somewhere.*", where *"somebody"* and *"somewhere"* are placeholders that can be replaced by the corresponding arguments.
- **Natural sentence with special tokens.** It is similar to the natural sentence one except for using role-specific special tokens instead of "some-" words. For example, "*<Person> was born in <Place>*." We consider this to study the *label semantics* of roles.
- **HTML-like sentence with special tokens.** To study the importance of using natural sentence, we also consider HTML-like sentence, e.g., "*<Person> </Person> <Place> </Place>*". The model aims to put argument predictions between the corresponding HTML tags.

The results of all variants of EAE templates on ACE05-E are shown in Table 7. We notice that writing templates in a natural language style get better performance, especially when only a few data is available (10% of data). This shows our design's capability to leverage pre-trained knowledge in the generation process. Additionally, there are over 1 F1 score performance drops when replacing natural language placeholders with special tokens. This confirms that leveraging label semantics for different roles is beneficial.

Model	10% Data		100% Data	
	Arg-I	Arg-C	Arg-I	Arg-C
OneIE	48.3	45.4	73.2	69.3
BART-Gen	-	-	69.9	66.7
Natural sentence	**63.3**	**57.3**	**76.0**	**73.5**
Natural sentence w/ special tokens	59.8	55.5	74.7	72.3
HTML-like sentence w/ special tokens	60.8	51.9	74.6	71.4

Table 7: Performances of DEGREE(EAE) on ACE05-E with different types of templates.

Model	10% Data		100% Data	
	Arg-I	Arg-C	Arg-I	Arg-C
OneIE	48.3	45.4	73.2	69.3
BART-Gen	-	-	69.9	66.7
DEGREE(EAE)	63.3	**57.3**	**76.0**	**73.5**
DEGREE(EAE) + variant template 1	61.6	55.5	73.4	70.4
DEGREE(EAE) + variant template 2	**63.9**	56.9	75.5	72.5

Table 8: Study on the effect of different template constructing rules. Experiments is conducted on ACE05-E.

Sensitivity to template design. Finally, we study how sensitive our model is to the template. In addition to the original design of templates for event argument extraction, we compose other two sets of templates with different constructing rules (e.g., different word choices and different orders of roles). Table 8 shows the results of using different sets of templates. We observe a performance fluctuation when using different templates, which indicates that the quality of templates does affect the performance to a certain degree. Therefore, we need to be cautious when designing templates. However, even though our model could be sensitive to the template design, it still outperforms OneIE and BART-Gen, which are the best classification-based model and the best generation-based baseline, respectively.

5 Related Work

Fully supervised event extraction. Event extraction has been studied for over a decade (Ahn, 2006; Ji and Grishman, 2008) and most traditional event extraction works follow the fully supervised setting (Nguyen et al., 2016; Sha et al., 2018; Nguyen and Nguyen, 2019; Yang et al., 2019b; Lin et al., 2020; Liu et al., 2020; Li et al., 2020). Many of them use classification-based models and use pipeline-style frameworks to extract events (Nguyen et al., 2016; Yang et al., 2019b; Wadden et al., 2019). To better leverage shared knowledge in event triggers and arguments, some works propose incorporating global features to jointly decide triggers and arguments (Lin et al., 2020; Li et al., 2013; Yang and Mitchell, 2016).

Recently, few generation-based event extraction models have been proposed (Paolini et al., 2021;

Huang et al., 2021, 2022; Li et al., 2021). TANL (Paolini et al., 2021) treats event extraction as translation tasks between augmented natural languages. Their predicted target—augmented language embed labels into the input passage via using brackets and vertical bar symbols. TempGen (Huang et al., 2021) is a template-based role-filler entity extraction model, which generate outputs that fill role entities into non-natural templated sequences. The output sequence designs of TANL and TempGen hinder the models from fully leveraging label semantics, unlike DEGREE that generates natural sentences. BART-Gen (Li et al., 2021) is also a generation-based model focusing on document-level event argument extraction. They solve event extraction with a pipeline, which prevents knowledge sharing across subtasks. All these fully supervised methods can achieve substantial performance with a large amount of annotated data. However, their designs are not specific for low-resource scenarios, hence, these models can not enjoy all the benefits that DEGREE obtains for low-resource event extraction at the same time, as we mentioned in Section 1.

Low-resource event extraction. It has been a growing interest in event extraction in a scenario with less data. Liu et al. (2020) uses a machine reading comprehension formulation to conduct event extraction in a low-resource regime. Text2Event (Lu et al., 2021), a sequence-to-structure generation paradigm, first presents events in a linearized format, and then trains a generative model to generate the linearized event sequence. Text2Event's unnatural output format hinders the model from fully leveraging pre-trained knowledge. Hence, their model falls short on the cases with only extremely low data being available (as shown in Section 3).

Another thread of works are using meta-learning to deal with the less label challenge (Deng et al., 2020; Shen et al., 2021; Cong et al., 2021). However, their methods can only be applied to event detection, which differs from our main focus on studying end-to-end event extraction.

6 Conclusion & Future Work

In this paper, we present DEGREE, a data-efficient generation-based event extraction model. DEGREE requires less training data because it better utilizes label semantics as well as weakly-supervised information, and captures better dependencies by jointly predicting triggers and arguments. Our experimental results and ablation studies show the superiority of DEGREE for low-resource event extraction.

DEGREE assumes that some weakly-supervised information (the description of events, similar keywords, and human-written templates) is accessible or not expensive for the users to craft. This assumption may holds for most situations. We leave the automation of template construction for future work, which can further ease the needed efforts when deploying DEGREE in a large-scale corpus.

Acknowledgments

We thank anonymous reviewers for their helpful feedback. We thank the UCLA PLUSLab and UCLA-NLP group members for their initial review and feedback for an earlier version of the paper. This work is supported in part by the Intelligence Advanced Research Projects Activity (IARPA), via Contract No. 2019-19051600007, and research awards sponsored by CISCO and Google.

Ethics Considerations

DEGREE fine-tunes the pre-trained generative language model (Lewis et al., 2020). Therefore, the generated output is potentially affected by the corpus for pre-training. Although with a low possibility, it is possible for our model to accidentally generate some malicious, counterfactual, and biased sentences, which may cause ethics concerns. We suggest carefully examining those potential issues before deploying the model in any real-world applications.

References

David Ahn. 2006. The stages of event extraction. In *Proceedings of the Workshop on Annotating and Reasoning about Time and Events*.

Jonathan Berant, Vivek Srikumar, Pei-Chun Chen, Abby Vander Linden, Brittany Harding, Brad Huang, Peter Clark, and Christopher D. Manning. 2014. Modeling biological processes for reading comprehension. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Xin Cong, Shiyao Cui, Bowen Yu, Tingwen Liu, Yubin Wang, and Bin Wang. 2021. Few-shot event detection with prototypical amortized conditional random field. In *Findings of the Association for Computational Linguistics: ACL/IJCNLP*.

Shumin Deng, Ningyu Zhang, Jiaojian Kang, Yichi Zhang, Wei Zhang, and Huajun Chen. 2020. Meta-learning with dynamic-memory-based prototypical network for few-shot event detection. In *The Thirteenth ACM International Conference on Web Search and Data Mining (WSDM)*.

George R. Doddington, Alexis Mitchell, Mark A. Przybocki, Lance A. Ramshaw, Stephanie M. Strassel, and Ralph M. 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)*.

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)*.

Steven Fincke, Shantanu Agarwal, Scott Miller, and Elizabeth Boschee. 2021. Language model priming for cross-lingual event extraction. *arXiv preprint arXiv:2109.12383*.

Rujun Han, I-Hung Hsu, Jiao Sun, Julia Baylon, Qiang Ning, Dan Roth, and Nanyun Peng. 2021. ESTER: A machine reading comprehension dataset for reasoning about event semantic relations. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*.

Rujun Han, I-Hung Hsu, Mu Yang, Aram Galstyan, Ralph M. Weischedel, and Nanyun Peng. 2019a. Deep structured neural network for event temporal relation extraction. In *Proceedings of the 23rd Conference on Computational Natural Language Learning, CoNLL 2019, Hong Kong, China, November 3-4, 2019*.

Rujun Han, Qiang Ning, and Nanyun Peng. 2019b. Joint event and temporal relation extraction with shared representations and structured prediction. In *2019 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Frederik Hogenboom, Flavius Frasincar, Uzay Kaymak, Franciska de Jong, and Emiel Caron. 2016. A survey of event extraction methods from text for decision support systems. *Decis. Support Syst.*, 85:12–22.

Kuan-Hao Huang, I-Hung Hsu, Premkumar 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 (ACL)*.

Kung-Hsiang Huang and Nanyun Peng. 2021. Document-level event extraction with efficient end-to-end learning of cross-event dependencies. In *The 3rd Workshop on Narrative Understanding (NAACL 2021)*.

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, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*.

Kung-Hsiang Huang, Mu Yang, and Nanyun Peng. 2020. Biomedical event extraction with hierarchical knowledge graphs. In *the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)-Findings, short*.

Heng Ji and Ralph Grishman. 2008. Refining event extraction through cross-document inference. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics (ACL)*.

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

Fayuan Li, Weihua Peng, Yuguang Chen, Quan Wang, Lu Pan, Yajuan Lyu, and Yong Zhu. 2020. Event extraction as multi-turn question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings (EMNLP)*.

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 (ACL)*.

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 (NAACL-HLT)*.

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 (ACL)*.

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)*.

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*.

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 (ACL/IJCNLP)*.

Thien Huu Nguyen, Kyunghyun Cho, and Ralph Grishman. 2016. Joint event extraction via recurrent neural networks. In *The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*.

Thien Huu Nguyen and Ralph Grishman. 2015. Event detection and domain adaptation with convolutional neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics (ACL)*.

Trung Minh Nguyen and Thien Huu Nguyen. 2019. One for all: Neural joint modeling of entities and events. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference (AAAI)*.

Giovanni Paolini, Ben Athiwaratkun, Jason Krone, Jie Ma, Alessandro Achille, Rishita Anubhai, Cícero Nogueira dos Santos, Bing Xiang, and Stefano Soatto. 2021. Structured prediction as translation between augmented natural languages. In *9th International Conference on Learning Representations (ICLR)*.

Lei Sha, Feng Qian, Baobao Chang, and Zhifang Sui. 2018. Jointly extracting event triggers and arguments by dependency-bridge RNN and tensor-based argument interaction. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence (AAAI)*.

Shirong Shen, Tongtong Wu, Guilin Qi, Yuan-Fang Li, Gholamreza Haffari, and Sheng Bi. 2021. Adaptive knowledge-enhanced bayesian meta-learning for few-shot event detection. In *Findings of the Association for Computational Linguistics: ACL/IJCNLP*.

Zhiyi Song, Ann Bies, Stephanie M. 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, (EVENTS@HLP-NAACL)*.

Jiao Sun and Nanyun Peng. 2021. Men are elected, women are married: Events gender bias on wikipedia. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 2: Short Papers), Virtual Event, August 1-6, 2021*.

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)*.

Xiaozhi Wang, Ziqi Wang, Xu Han, Wangyi Jiang, Rong Han, Zhiyuan Liu, Juanzi Li, Peng Li, Yankai Lin, and Jie Zhou. 2020. MAVEN: A massive general domain event detection dataset. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*.

Xiaozhi Wang, Ziqi Wang, Xu Han, Zhiyuan Liu, Juanzi Li, Peng Li, Maosong Sun, Jie Zhou, and Xiang Ren. 2019. HMEAE: hierarchical modular event argument extraction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*.

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. Huggingface's transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*.

Bishan Yang and Tom M. Mitchell. 2016. Joint extraction of events and entities within a document context. In *The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*.

Sen Yang, Dawei Feng, Linbo Qiao, Zhigang Kan, and Dongsheng Li. 2019a. Exploring pre-trained language models for event extraction and generation. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*.

Sen Yang, Dawei Feng, Linbo Qiao, Zhigang Kan, and Dongsheng Li. 2019b. Exploring pre-trained language models for event extraction and generation. In *Proceedings of the 57th Conference of the Association for Computational Linguistics (ACL)*.

Hongming Zhang, Xin Liu, Haojie Pan, Yangqiu Song, and Cane Wing-Ki Leung. 2020. ASER: A large-scale eventuality knowledge graph. In *The Web Conference 2020 (WWW)*.

A EAE Template Constructing

Our strategy to create an EAE template is first identifying all valid argument roles for the event type,[4] such as *Attacker*, *Target*, *Instrument*, and *Place* roles. Then, for each argument role, according to the semantics of the role type, we select natural and fluent words to form its placeholder (e.g., *some way* for *Instrument*). This design aims to provide a simple way to help the model learn both the roles' label semantics and the *event structure*. Finally, we create a natural language sentence that connects all these placeholders. Notice that we try to keep the template as simple and short as possible. Table 9 lists all designed EAE templates for ACE05-E and ACE05-E$^+$. The EAE templates of ERE-EN can be found in Table 10.

B Training Details of Proposed Model

Given a passage, its annotated event types are consider as positive event types. During training, we additionally sample m event types that are not related to the passage as the negative examples, where m is a hyper-parameter. In our experiments, m is usually set to 13 or 15.

For all of DEGREE, DEGREE(ED), and DEGREE(EAE), we fine-tune the pre-trained BART-large (Lewis et al., 2020) with Huggingface package (Wolf et al., 2020). The number of parameters is around 406 millions. We train DEGREE with our machine that equips 128 AMD EPYC 7452 32-Core Processor, 4 NVIDIA A100 GPUs, and 792G RAM. We consider AdamW optimizer (Loshchilov and Hutter, 2019) with learning rate set to 10^{-5} and the weight decay set to 10^{-5}. We set the batch size to 6 for DEGREE(EAE) and 32 for DEGREE(ED) and DEGREE. The number of training epochs is 45. It takes around 2 hours, 18 hours, 22 hours to train DEGREE(EAE), DEGREE(ED), and DEGREE, respectively.

We do hyper-parameter search on m, the number of negative examples, from $\{3, 5, 7, 10, 13, 15, 18, 21\}$, and our preliminary trials shows that m less than 10 are usually less useful. For the learning rate and the weight decay, we tune it based on our preliminary experiment for event argument extraction from $\{10^{-5}, 10^{-4}\}$, while they are both fixed to 10^{-5} for all the experiments.

C Datasets

We consider ACE 2005[5] (Doddington et al., 2004) and ERE[6] (Song et al., 2015). Both consider *LDC User Agreement for Non-Members*[7] as the licenses. Both datasets are created for entity, relation, and event extraction while our focus is only event extraction in this paper. In the original ACE 2005 dataset, it contains data for English, Chinese, and Arabic and we only take the English data for our experiment. In the original ERE dataset, it contains data for English, and Chinese and we only take the English data for our experiment as well.

Because both datasets contain event like *Justice:Execute* and *Life:Die*, it is possible that some offensive words (e.g., killed) would appear in the passage. Also, some real names may appear in the passage as well (e.g., Palestinian president, Mahmoud Abbas). How to accurately identify these kinds of information is part of the goal of the task. Therefore, we do not take any changes on the datasets for protecting or anonymizing.

We split the training data based on documents, which is a more realistic setup compared to splitting data by instance. Table 11 lists the statistics of ACE05-E, ACE05-E$^+$, and ERE-EN. Specifically, we try to make each proportion of data contain as many event types as possible.

D Implementation Details

This section describes the implementation details for all baselines we use. We run the experiments with three different random seeds and report the best value.

- **DyGIE++**: we use their released pre-trained model[8] for evaluation.
- **OneIE**: we use their provided code[9] to train the model with default parameters.
- **BERT_QA**: we use their provided code[10] to train the model with default parameters.
- **TANL**: we use their provided code[11] to train the

[4] The valid roles for each event type are predefined in the event ontology for each dataset, or can be decided by the user of interest.

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

[6] https://catalog.ldc.upenn.edu/LDC2020T19

[7] https://catalog.ldc.upenn.edu/license/ldc-non-members-agreement.pdf

[8] https://github.com/dwadden/dygiepp

[9] http://blender.cs.illinois.edu/software/oneie/

[10] https://github.com/xinyadu/eeqa

[11] https://github.com/amazon-research/tanl

model. We conduct the experiments with two variations: (1) using their default parameters, and (2) using their default parameters but with more training epochs. We observe that the second variant works better. As a result, we report the number obtained from the second setting.

- **Text2Event**: we use their official code[12] to train the model with the provided parameter setting.
- **dbRNN:** we directly report the experimental results from their paper.
- **Joint3EE:** we directly report the experimental results from their paper.
- **MQAEE:** we directly report the experimental results from their paper.
- **BART-Gen**: we report the experimental results from their released appendix.[13]

[12] https://github.com/luyaojie/Text2Event

[13] https://github.com/raspberryice/gen-arg/blob/main/NAACL_2021_Appendix.pdf

Event Type	**EAE Template**
Life:Be-Born	somebody was born in somewhere.
Life:Marry	somebody got married in somewhere.
Life:Divorce	somebody divorced in somewhere.
Life:Injure	somebody or some organization led to some victim injured by some way in somewhere.
Life:Die	somebody or some organization led to some victim died by some way in somewhere.
Movement:Transport	something was sent to somewhere from some place by some vehicle. somebody or some organization was responsible for the transport.
Transaction:Transfer-Ownership	someone got something from some seller in somewhere.
Transaction:Transfer-Money	someone paid some other in somewhere.
Business:Start-Org	somebody or some organization launched some organzation in somewhere.
Business:Merge-Org	some organzation was merged.
Business:Declare-Bankruptcy	some organzation declared bankruptcy.
Business:End-Org	some organzation dissolved.
Conflict:Attack	some attacker attacked some facility, someone, or some organization by some way in somewhere.
Conflict:Demonstrate	some people or some organization protest at somewhere.
Contact:Meet	some people or some organization met at somewhere.
Contact:Phone-Write	some people or some organization called or texted messages at somewhere.
Personnel:Start-Position	somebody got new job and was hired by some people or some organization in somewhere.
Personnel:End-Position	somebody stopped working for some people or some organization at somewhere.
Personnel:Nominate	somebody was nominated by somebody or some organization to do a job.
Personnel:Elect	somebody was elected a position, and the election was voted by some people or some organization in somewhere.
Justice:Arrest-Jail	somebody was sent to jailed or arrested by somebody or some organization in somewhere.
Justice:Release-Parole	somebody was released by some people or some organization from somewhere.
Justice:Trial-Hearing	somebody, prosecuted by some other, faced a trial in somewhere. The hearing was judged by some adjudicator.
Justice:Charge-Indict	somebody was charged by some other in somewhere. The adjudication was judged by some adjudicator.
Justice:Sue	somebody was sued by some other in somewhere. The adjudication was judged by some adjudicator.
Justice:Convict	somebody was convicted of a crime in somewhere. The adjudication was judged by some adjudicator.
Justice:Sentence	somebody was sentenced to punishment in somewhere. The adjudication was judged by some adjudicator.
Justice:Fine	some people or some organization in somewhere was ordered by some adjudicator to pay a fine.
Justice:Execute	somebody was executed by somebody or some organization at somewhere.
Justice:Extradite	somebody was extradicted to somewhere from some place. somebody or some organization was responsible for the extradition.
Justice:Acquit	somebody was acquitted of the charges by some adjudicator.
Justice:Pardon	somebody received a pardon from some adjudicator.
Justice:Appeal	some other in somewhere appealed the adjudication from some adjudicator.

Table 9: All EAE templates for ACE05-E and ACE05-E$^+$.

Event Type	**EAE Template**
Life:Be-Born	somebody was born in somewhere.
Life:Marry	somebody got married in somewhere.
Life:Divorce	somebody divorced in somewhere.
Life:Injure	somebody or some organization led to some victim injured by some way in somewhere.
Life:Die	somebody or some organization led to some victim died by some way in somewhere.
Movement:Transport-Person	somebody was moved to somewhere from some place by some way. somebody or some organization was responsible for the movement.
Movement:Transport-Artifact	something was sent to somewhere from some place. somebody or some organization was responsible for the transport.
Business:Start-Org	somebody or some organization launched some organzation in somewhere.
Business:Merge-Org	some organzation was merged.
Business:Declare-Bankruptcy	some organzation declared bankruptcy.
Business:End-Org	some organzation dissolved.
Conflict:Attack	some attacker attacked some facility, someone, or some organization by some way in somewhere.
Conflict:Demonstrate	some people or some organization protest at somewhere.
Contact:Meet	some people or some organization met at somewhere.
Contact:Correspondence	some people or some organization contacted each other at somewhere.
Contact:Broadcast	some people or some organization made announcement to some publicity at somewhere.
Contact:Contact	some people or some organization talked to each other at somewhere.
Manufacture:Artifact	something was built by somebody or some organization in somewhere.
Personnel:Start-Position	somebody got new job and was hired by some people or some organization in somewhere.
Personnel:End-Position	somebody stopped working for some people or some organization at somewhere.
Personnel:Nominate	somebody was nominated by somebody or some organization to do a job.
Personnel:Elect	somebody was elected a position, and the election was voted by somebody or some organization in somewhere.
Transaction:Transfer-Ownership	The ownership of something from someone was transferred to some other at somewhere.
Transaction:Transfer-Money	someone paid some other in somewhere.
Transaction:Transaction	someone give some things to some other in somewhere.
Justice:Arrest-Jail	somebody was sent to jailed or arrested by somebody or some organization in somewhere.
Justice:Release-Parole	somebody was released by somebody or some organization from somewhere.
Justice:Trial-Hearing	somebody, prosecuted by some other, faced a trial in somewhere. The hearing was judged by some adjudicator.
Justice:Charge-Indict	somebody was charged by some other in somewhere. The adjudication was judged by some adjudicator.
Justice:Sue	somebody was sued by some other in somewhere. The adjudication was judged by some adjudicator.
Justice:Convict	somebody was convicted of a crime in somewhere. The adjudication was judged by some adjudicator.
Justice:Sentence	somebody was sentenced to punishment in somewhere. The adjudication was judged by some adjudicator.
Justice:Fine	some people or some organization in somewhere was ordered by some adjudicator to pay a fine.
Justice:Execute	somebody was executed by somebody or some organization at somewhere.
Justice:Extradite	somebody was extradicted to somewhere from some place. somebody or some organization was responsible for the extradition.
Justice:Acquit	somebody was acquitted of the charges by some adjudicator.
Justice:Pardon	somebody received a pardon from some adjudicator.
Justice:Appeal	somebody in somewhere appealed the adjudication from some adjudicator.

Table 10: All EAE templates for ERE-EN.

Dataset	Split	#Docs	#Sents	#Events	#Event Types	#Args	#Arg Types
ACE05-E	Train (full)	529	17172	4202	33	4859	22
	Train (1%)	5	103	47	14	65	16
	Train (2%)	10	250	77	17	104	16
	Train (3%)	15	451	119	23	153	17
	Train (5%)	25	649	212	27	228	21
	Train (10%)	50	1688	412	28	461	21
	Train (20%)	110	3467	823	33	936	22
	Train (30%)	160	5429	1368	33	1621	22
	Train (50%)	260	8985	2114	33	2426	22
	Dev	28	923	450	21	605	22
	Test	40	832	403	31	576	20
ACE05-E^{+}	Train (full)	529	19216	4419	33	6607	22
	Train (1%)	5	92	49	15	75	16
	Train (2%)	10	243	82	19	129	16
	Train (3%)	15	434	124	24	203	19
	Train (5%)	25	628	219	27	297	21
	Train (10%)	50	1915	428	29	629	21
	Train (20%)	110	3834	878	33	1284	22
	Train (30%)	160	6159	1445	33	2212	22
	Train (50%)	260	10104	2231	33	3293	22
	Dev	28	901	468	22	759	22
	Test	40	676	424	31	689	21
ERE-EN	Train (full)	396	14736	6208	38	8924	21
	Train (1%)	4	109	61	14	78	16
	Train (2%)	8	228	128	21	183	19
	Train (3%)	12	419	179	26	272	19
	Train (5%)	20	701	437	31	640	21
	Train (10%)	40	1536	618	37	908	21
	Train (20%)	80	2848	1231	38	1656	21
	Train (30%)	120	4382	1843	38	2632	21
	Train (50%)	200	7690	3138	38	4441	21
	Dev	31	1209	525	34	730	21
	Test	31	1163	551	33	822	21

Table 11: Dataset statistics. Our experiments are conducted in sentences, which were split from documents. In the table, "#Docs" means the number of documents; "#Sents" means the number of sentences, "#Events" means the number of events; "#Event Types" means the number of event types in total; "#Args" means the number of argument in total; "#Arg Types" means the number of argument role types in total.

E Few-Shot and Zero-Shot Event Extraction

In order to further test our models' generaliability, we additionally conduct zero-shot and few-shot experiments on the ACE05-E dataset with DEGREE(ED) and DEGREE(EAE).

Settings. We first select the top n common event types as "seen" types and use the rest as "unseen/rare" types, where the top common types are listed in Table 12. To simulate a zero-shot scenario, we remove all events with "unseen/rare" types from the training data. To simulate a few-shot scenario, we keep only k event examples for each "unseen/rare" type (denoted as k-shot). During the evaluation, we calculate micro F1-scores only for these "unseen/rare" types.

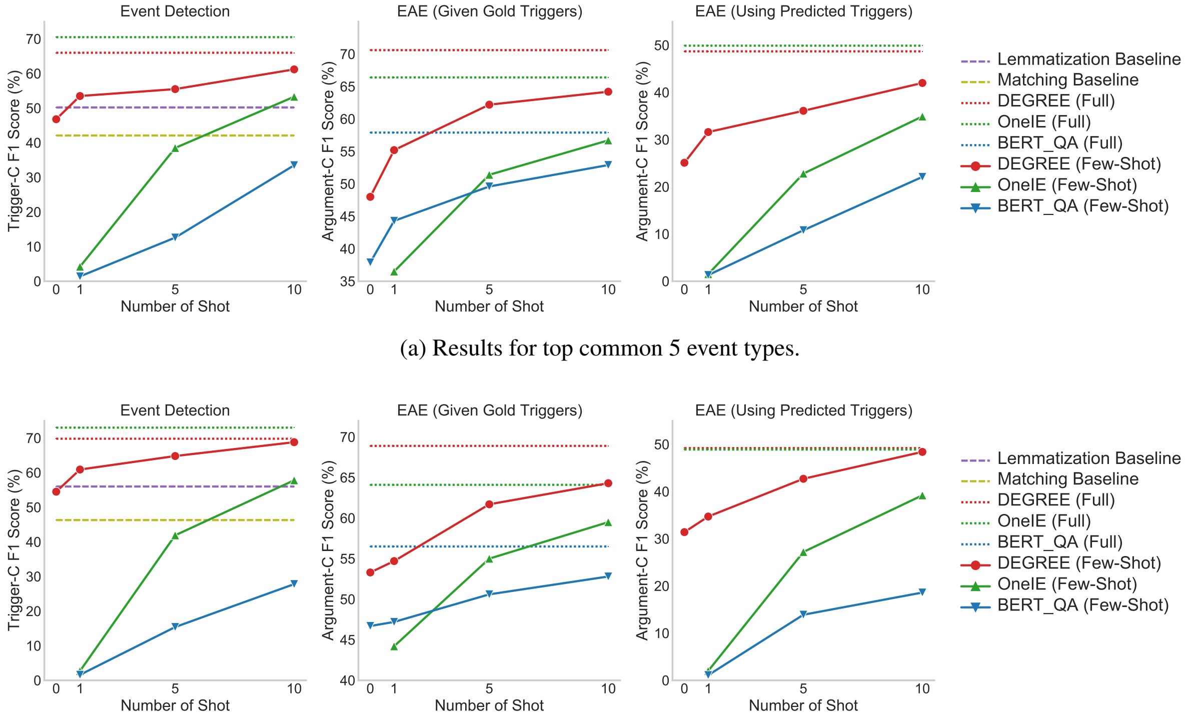
n	Seen Event Types for Training/Development
5	Conflict:Attack, Movement:Transport, Life:Die, Contact:Meet, Personnel:Elect
10	Conflict:Attack, Movement:Transport, Life:Die, Contact:Meet, Personnel:Elect, Life:Injure, Personnel:End-Position, Justice:Trial-Hearing, Contact:Phone-Write, Transaction:Transfer-Money

Table 12: Common event types in ACE05-E.

Compared baselines. We consider the following baselines: (1) **BERT_QA** (Du and Cardie, 2020) (2) **OneIE** (Lin et al., 2020) (3) **Matching baseline**, a proposed baseline that makes trigger predictions by performing string matching between the input passage and the event keywords. (4) **Lemmatization baseline**, another proposed baseline that performs string matching on lemmatized input passage and the event keywords. (Note: (3) and (4) are baselines only for event detection tasks.)

Experimental results. Figure 4, Table 13, and Table 14 show the results of $n = 5$ and $n = 10$. From the two subfigures in the left column, we see that DEGREE(ED) achieves promising results in the zero-shot setting. In fact, it performs better than BERT_QA trained in the 10-shot setting and OneIE trained in the 5-shot setting. This demonstrates the great potential of DEGREE(ED) to discover new event types. Interestingly, we observe that our two proposed baselines perform surprisingly well, suggesting that the trigger annotations in ACE05-E are actually not diverse. Despite their impressive performance, DEGREE(ED) still outperforms the matching baseline by over 4.7% absolute trigger classification F1 in both $n = 5$ and $n = 10$ cases in zero-shot scenario. Additionally, with only one training instance for each unseen type, DEGREE(ED) can outperform both proposed baselines.

Next, we compare the results for the event argument extraction task. From the two middle subfigures, we observe that when given gold triggers, our model performs much better than all baselines with a large margin. Lastly, we train models for both trigger and argument extraction and report the final argument classification scores in the two right subfigures. We justify that our model has strong generalizability to unseen event types and it can outperform BERT_QA and OneIE even when they are both trained in 5-shot settings.



(a) Results for top common 5 event types.

(b) Results for top common 10 event types.

Figure 4: The zero/few-shot experimental results. ***Left***: The result for the models on event detection task with the scores reported in trigger classification F1. ***Middle***: The models are tested under the scenario of given gold trigger and evaluated with argument classification criterion. ***Right***: The results for the models to perform event extraction task, which aims to predict triggers and their corresponding arguments (we report the argument classification F1).

Event Extraction									
Trigger	Argument	Common 5				Common 10			
		Tri-I	Tri-C	Arg-I	Arg-C	Tri-I	Tri-C	Arg-I	Arg-C
Matching Baseline		42.7	42.1	-	-	46.3	46.3	-	-
Lemmatization Baseline		51.5	50.2	-	-	56.6	56.0	-	-
BERT_QA 1-shot		10.0	1.4	1.3	1.3	8.2	1.6	1.1	1.1
BERT_QA 5-shot		14.0	12.6	11.1	10.8	20.8	15.4	14.6	13.9
BERT_QA 10-shot		37.8	33.5	22.9	22.1	32.0	27.8	19.5	18.6
OneIE 1-shot		4.2	4.2	1.5	1.5	4.1	2.7	2.0	2.0
OneIE 5-shot		39.3	38.5	24.8	22.8	41.9	41.9	29.7	27.2
OneIE 10-shot		54.8	53.3	36.0	34.9	61.5	57.8	41.4	39.2
DEGREE(ED) 0-shot	DEGREE(EAE) 0-shot	53.3	46.8	29.6	25.1	60.9	54.5	42.0	31.4
DEGREE(ED) 1-shot	DEGREE(EAE) 1-shot	60.1	53.3	38.8	31.6	61.2	60.9	41.1	34.7
DEGREE(ED) 5-shot	DEGREE(EAE) 5-shot	57.8	55.5	40.6	36.1	65.8	64.8	45.3	42.7
DEGREE(ED) 10-shot	DEGREE(EAE) 10-shot	63.8	61.2	46.0	42.0	72.1	68.8	52.5	48.4
OneIE (Full)		72.7	70.5	52.3	49.9	74.5	73.0	51.2	48.9
DEGREE(ED) (Full)	DEGREE(EAE) (Full)	68.4	66.0	51.9	48.7	72.0	69.8	52.5	49.2

Table 13: Full results of zero/few-shot event extraction on ACE05-E.

Event Argument Extraction									
Trigger	Argument	Common 5				Common 10			
		Tri-I	Tri-C	Arg-I	Arg-C	Tri-I	Tri-C	Arg-I	Arg-C
Gold Triggers	BERT_QA 0-shot	100.0	100.0	55.8	37.9	100.0	100.0	57.2	46.7
Gold Triggers	BERT_QA 1-shot	100.0	100.0	55.8	44.3	100.0	100.0	57.8	47.2
Gold Triggers	BERT_QA 5-shot	100.0	100.0	56.6	49.6	100.0	100.0	59.1	50.6
Gold Triggers	BERT_QA 10-shot	100.0	100.0	58.8	52.9	100.0	100.0	60.5	52.8
Gold Triggers	OneIE 1-shot	100.0	100.0	40.9	36.5	100.0	100.0	48.3	44.2
Gold Triggers	OneIE 5-shot	100.0	100.0	55.6	51.4	100.0	100.0	58.6	55.0
Gold Triggers	OneIE 10-shot	100.0	100.0	59.4	56.7	100.0	100.0	62.0	59.5
Gold Triggers	DEGREE(EAE) 0-shot	100.0	100.0	56.1	48.0	100.0	100.0	66.5	53.3
Gold Triggers	DEGREE(EAE) 1-shot	100.0	100.0	65.2	55.2	100.0	100.0	65.4	54.7
Gold Triggers	DEGREE(EAE) 5-shot	100.0	100.0	70.9	62.2	100.0	100.0	68.0	61.7
Gold Triggers	DEGREE(EAE) 10-shot	100.0	100.0	71.1	64.2	100.0	100.0	71.6	64.3
Gold Triggers	BERT_QA (Full)	100.0	100.0	63.1	57.9	100.0	100.0	62.1	56.5
Gold Triggers	OneIE (Full)	100.0	100.0	70.8	66.4	100.0	100.0	67.9	64.1
Gold Triggers	DEGREE(EAE) (Full)	100.0	100.0	74.5	70.6	100.0	100.0	73.6	68.9

Table 14: Full results of zero/few-shot event argument extraction on ACE05-E.