DEGREE^2: Efficient Extraction of Multiple Events Using Language Models

Philip Blair[1,2] and **Kfir Bar**[1]

[1]Babel Street, Reston, VA, USA[*]
[2]Blair Software, Amsterdam, The Netherlands
{pblair,kbar}@babelstreet.com

Abstract

Language models (LMs) show exceptional promise in the area of few-shot event extraction, but they suffer from certain limitations. In particular, DEGREE (Hsu et al., 2022) is an LM-based event extraction model that has recently been supplanted by other large language model-based state-of-the-art systems, but it suffers from an inability to cope with multiple events in the same region of an input document. In this work, we present a simple method for extending this system with the ability to gracefully handle different densities of events within documents, thereby rendering it competitive with the state-of-the-art once more, and additionally explore a novel evaluation metric that can be used to qualitatively compare the outputs of different event extraction systems. Finally, we show that our extension allows models to break apart documents into less small pieces during processing without sacrificing accuracy.

```
John met with Alice and then Steve.
[...template...]
Event trigger is met.
John and Alice met at some place.
```

(a) Sample ACE2005 `Contact:Meet` completed prompt from DEGREE.

```
John met with Alice and then Steve.
[...template...]
<EVENTSEP>Event trigger is met.
John and Alice met at some place.
<EVENTSEP>Event trigger is met.
John and Steve met at some place.
```

(b) Our version of the equivalent completed prompt.

Figure 1: Fine-tuning prompts used in our work compared to DEGREE (Hsu et al., 2022). Text in **blue** denotes the input text to perform the event detection on. `[...template...]` represents the input template (Section 3), with the following text being the expected generation of the Large Language Model (LLM). Text in violet denotes the trigger phrase, teal the event participants, and magenta the event location. Finally, orange text denotes special tokens added to the model vocabulary. At inference time, the LLM generates text after the input source portion.

1 Introduction

In the domain of information extraction (IE), event extraction is a task consisting of identifying specific occurrences of things which happen involving participants (LDC, 2005). This task poses a number of unique challenges for information extraction systems, as proper detection of events typically requires an in-depth understanding of the semantics of input sentences, as opposed to simple lexical information. For example, the sentence "John went to San Antonio" denotes a `Movement:Transport`-type event, whereas the sentence "The first point went to San Antonio" does not.

The bulk of the literature on event extraction descends from the original ACE2005 information extraction dataset published by the Linguistic Data Consortium (LDC, 2005). Notably, this decomposes the event extraction task into two subtasks: *event detection* (also known as *trigger extraction*) and *argument extraction*. For example, in the sentence "John met with Alice", "met" is the *trigger* (the phrase which clearly expresses the occurrence of the event), while "John" and "Alice" are the *arguments* of the event. Arguments can have a number of different event-specific types, such as meeting participants, locations, and relevant actors (e.g. the victim of a crime).

Supervised machine learning is a natural choice for modeling this problem, but the drawback of these approaches is that such training generally requires a large quantity of annotated data due to the need to understand the semantic nuances of text when performing this task. Anecdotally, this can be prohibitive in a number of real-world ap-

[*]Research conducted at Babel Street.

Proceedings of the Workshop on the Future of Event Detection (FuturED), pages 25–31
November 15, 2024 ©2024 Association for Computational Linguistics

plications of event extraction systems, due to the fact that downstream users often (a) require a diverse set of event types and (b) these event types are many times unique to their use case (preventing useful sharing of annotated datasets between different users).

With the advent of powerful language models and Large Language Models (LLMs) (Brown et al., 2020; OpenAI, 2024), a number of novel low-resource and zero-shot methods have been developed which leverage these models' abilities to be fine-tuned to new tasks with relatively little data. One such model, known as DEGREE (Hsu et al., 2022), was until recently considered the state of the art in few-shot event extraction until being supplanted by the EE-LCE (Yu et al., 2024) model. While this would suggest a superior method for fine-tuning LLMs for event extraction, we find that this performance gap can be explained away by controlling for a specific limitation of the DEGREE model: its inability to extract more than one event from the same region of text.

In summary, our contributions are as follows:

1. We present a simple extension of the DEGREE event extraction system which allows it to extract multiple events from the same piece of text.

2. We demonstrate that this extension makes DEGREE competitive with the state-of-the-art generative event extraction model.

3. We describe a novel E2E event extraction evaluation metric which can be used to qualitatively compare model performance irrespective of whether they handle multiple events.

2 Related Work

The bulk of research into event extraction focuses on high-resource scenarios, with models based on traditional supervised machine learning techniques. Examples of this include techniques based on decision trees (Ahn, 2006), support vector machines (Hong et al., 2011), convolutional neural networks (Nguyen and Grishman, 2015), recurrent neural networks (Nguyen et al., 2016), and graph convolutional neural networks (Nguyen and Grishman, 2018). Broadly speaking, all of these approaches are based on the idea of training a machine learning algorithm from scratch to recognize event triggers and arguments using features which are either hand-crafted or, in the case of the neural network-based algorithms, automatically learned.

More recent approaches to event extraction leverage language models. The basic idea of these techniques is to leverage the natural language modeling capacity of pretrained language models in order to reduce training data requirements via posing event extraction as a text-based natural language generation task. Consequently, these techniques focus more on few-shot and zero-shot learning scenarios. The state-of-the-art in this space is EE-LCE (Yu et al., 2024), which is an extension of InstructUIE (Wang et al., 2023). These `flan-t5-xxl`-based (Chung et al., 2022) models are trained via a multi-task learning algorithm designed to cover a large number of information extraction tasks. Their results slightly beat out the previous state-of-the-art, known as DEGREE (Hsu et al., 2022), which is the inspiration for our work.

For a more detailed history of event extraction datasets and systems, see Lai (2022).

3 Methodology

Before describing our extension to the model, we first provide a brief overview of the design of DEGREE (Hsu et al., 2022). The system frames the event extraction task in terms of a natural language generation task, with the generated text being rigidly structured in order to be machine-parsable. Consider the sentence, "John met with Alice." DEGREE might query this input for `Contact:Meet` events with the following input:

```
John met with Alice.
contact event, meet sub-type
The event is related people meeting.
Similar triggers such as meet, met.
The event's trigger word is <Trigger>.
some people met at somewhere.
```

The final two lines serve as a "prototype" template that should appear in the output. In this instance, we expect the fine-tuned model to produce the following completion:

```
Event trigger is met.
John and Alice met somewhere.
```

For inputs where no event is found, the completion `Event trigger is <Trigger>` is generated.

DEGREE is trained by fine-tuning a base LLM to complete patterns such as the above. Once trained, the LLM is able to extract not only the event types which it was trained on, but also, to

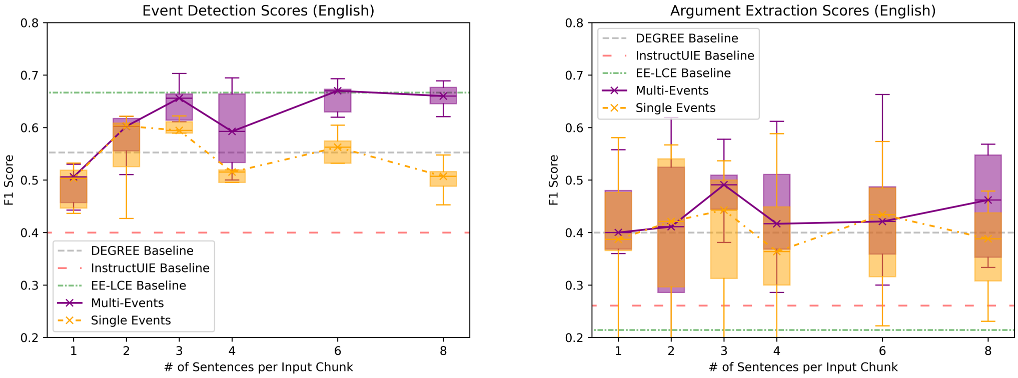

Figure 2: ACE2005 MUC-style (Chinchor and Sundheim, 1993) F1 scores for different system configurations. Horizontal lines represent the median scores of baseline systems. The box plots represent all of the scores from the events and arguments related to the five event types we analyze (Section 3), with the lines in the center of each box denoting the median score. The detailed scores can be found in Appendix B

some extent, new event types in a zero-shot fashion. These completions are easily parsable into structured formats, and the generated strings can be searched for in the original input in order to function as a text annotation algorithm.

As shown above, DEGREE is able to extract zero or one events from a given piece of text. How can entire documents then be handled? DEGREE addresses this by chunking input documents into pieces consisting of three sentences[1]. All of a document's chunks are processed separately (once per event type) in order to perform event extraction across the full input.

One remaining limitation is the handling of multiple events of the same type in the same chunk. DEGREE does not address this situation, so we propose an update to the fine-tuning template structure which allows this type of scenario to be handled. Our proposed template is shown in Figure 1b. The key modification is the introduction of the <EVENTSEP> special token, which separates each event in the output. While a rather minor change, we show below that this is enough to close the gap between DEGREE and the state of the art.

4 Experimental Results

We evaluate our system on a variety of configurations using the ACE2005 dataset (LDC, 2005)'s English data. To determine sentence boundaries, we use the Babel Street Analytics text analysis framework.

Our models are based on `t5-large` (Raffel et al., 2020), as we empirically found this to be a better choice than DEGREE's base model of BART (Lewis et al., 2020). For different numbers of sentences used to chunk apart input documents, we train two versions of each model: one with multi-events turned off (i.e. the same algorithm as DEGREE, with our base model and template, limited to a single event per chunk), and one with multiple events per input chunk. Additional training details can be found in Appendix A.

Additionally, we compare against three baselines: DEGREE, InstructUIE (Wang et al., 2023), and EE-LCE (Yu et al., 2024). For DEGREE and InstructUIE, we use the models published by the authors. For EE-LCE, we use the provided training code to create a model.

To focus on the most pertinent subset of the dataset, we limit our analysis to the five event types with the highest support in the test data: Conflict:Attack, Contact:Meet, Movement:Transport, Personnel:End-Position, and Transaction:Transfer-Ownership. Finally, since we feel that it is more representative of performance on argument extraction, we opt to use a MUC-style (Chinchor and Sundheim, 1993) formula for calculating F1. This is identical to the traditional formula, except partial matches are counted as 50% correct (rather than completely incorrect).

When interpreting the data in Figure 2, we find that extending DEGREE to support multiple events causes two changes in the behavior of the model. First, the event detection performance becomes very similar to the state-of-the-art EE-LCE system, despite being based on a model with 750MM pa-

[1] This choice of three was not explained in DEGREE's paper, but our results in Section 4 agree with this choice.

rameters (in contrast to EE-LCE's 11B). Second, model is able to process more sentences at once without sacrificing accuracy. Because DEGREE requires `num_chunks`×`num_event_types` invocations in order to process a document broken apart into `num_chunks` pieces, this means that we can effectively halve (or more) the number of model invocations required to process a document.

5 Relaxed F1: Co-Arity-Invariant Comparison of Event Extraction Algorithms

Our exploration into the impact of this single-event limitation of DEGREE on its comparative performance led us to consider whether there was a way to compare the *qualitative* performance of these algorithms in a mathematical way. For example, suppose that there is a dataset where each sentence contains one `Conflict:Attack` event, and we run algorithms A and B on chunks of two sentences. Algorithm A is limited to zero or one outputs per sentence, but it detects an attack event in each pair of sentences. In contrast, algorithm B can output an arbitrary number of events, and it detects both attack events in each pair.

Which is better? In an absolute sense, algorithm B outperforms, since we calculate precision and recall metrics with respect to the number of events contained in the document. For certain applications, however, we may be more interested in knowing which of the two qualitatively performs better. In a certain sense, these algorithms are equivalent, since *within the scope of its limitations*, A and B both extract attack events as much as is possible.

To address this shortcoming, we present a new metric for event detection, which we call *relaxed F1 scores*. The formula for this score is derived from the partial-match-aware MUC formulae (Chinchor and Sundheim, 1993) and defined by the following formula for "relaxed" recall:

$$R^{\text{relaxed}} = \frac{\text{correct} + (0.5 \times \text{partial}) - \text{extra}}{\text{possible} - \text{impossible}}$$

In this equation, "correct" and "partial" denote the number of correctly-extracted, partially-extracted (extractions of the correct type but only a partial overlap with the correct location) events or arguments, "possible" the number of events or arguments in the gold annotation. "impossible" denotes the number of events greater than one in each chunk (i.e. for a chunk with five events, "impossible" would be four). Finally, the "extra" term is needed for algorithms which *can* extract multiple events, in order to make the result comparable with ones which cannot. For these algorithms, "extra" denotes the number of correct (or weighted partial) predictions which were made that would have been impossible if multiple events could not be extracted. In sum, this means that, effectively, for each chunk of text produced during processing, the calculation of relaxed recall becomes binary.

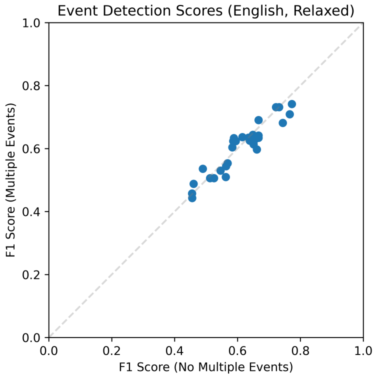

Figure 3: ACE2005 relaxed F1 scores across all system configurations. For further details, see Appendix B.

From this relaxed recall value, relaxed F1 is computed by using the standard formula alongside the standard precision P:

$$F_1^{\text{relaxed}} = \frac{2 \times P \times R^{\text{relaxed}}}{P + R^{\text{relaxed}}}$$

We use this metric to determine whether our multiple event extraction extension qualitatively decreases the event detection performance of DEGREE, with the results shown in Figure 3. This graph shoes a roughly linear correlation between the two values, meaning that our extension does not meaningfully degrade DEGREE's qualitative performance.

6 Discussion

We demonstrate that a simple extension to DEGREE is sufficient to close the gap between it and state-of-the-art systems. This suggests that different generative approaches to event extraction are potentially much more competitive with one another than previously thought.

Furthermore, we present an F1-style event detection metric which can give some insight into the qualitative performance of these algorithms.

We hope that this motivates further research into ways of analyzing these systems' performance in more fine-grained detail. Future work could include a metric that allows for assessing argument extraction performance without depending on event detection accuracy.

7 Limitations

The systems described in this paper are trained on annotated event datasets. While they have some capacity to generalize to new event types in a zero-shot fashion, users should be cautious when using them with event types not found in the training data, as they may produce unexpected predictions.

The analysis presented here focuses on the English-language ACE2005 data. Some of the conclusions presented here may not hold for certain other languages, and the systems described here may not function correctly on non-English input text.

References

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

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Nancy Chinchor and Beth Sundheim. 1993. MUC-5 evaluation metrics. In *Fifth Message Understanding Conference (MUC-5): Proceedings of a Conference Held in Baltimore, Maryland, August 25-27, 1993*.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. 2022. Scaling instruction-finetuned language models. *Preprint*, arXiv:2210.11416.

Yu Hong, Jianfeng Zhang, Bin Ma, Jianmin Yao, Guodong Zhou, and Qiaoming Zhu. 2011. Using cross-entity inference to improve event extraction. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1127–1136, Portland, Oregon, USA. Association for Computational Linguistics.

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

Viet Dac Lai. 2022. Event extraction: A survey. *Preprint*, arXiv:2210.03419.

LDC. 2005. ACE (automatic content extraction) English annotation guidelines for events. `https://www.ldc.upenn.edu/sites/www.ldc.upenn.edu/files/english-events-guidelines-v5.4.3.pdf`.

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*, pages 7871–7880, Online. Association for Computational Linguistics.

Thien Nguyen and Ralph Grishman. 2018. Graph convolutional networks with argument-aware pooling for event detection. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1).

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.

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 and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 365–371, Beijing, China. Association for Computational Linguistics.

OpenAI. 2024. GPT-4 technical report. *Preprint*, arXiv:2303.08774.

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

Noam Shazeer and Mitchell Stern. 2018. Adafactor: Adaptive learning rates with sublinear memory cost. *Preprint*, arXiv:1804.04235.

Xiao Wang, Weikang Zhou, Can Zu, Han Xia, Tianze Chen, Yuansen Zhang, Rui Zheng, Junjie Ye, Qi Zhang, Tao Gui, et al. 2023. Instructuie: Multi-task instruction tuning for unified information extraction. *arXiv preprint arXiv:2304.08085*.

Yanhua Yu, Yuanlong Wang, Yunshan Ma, Jie Li, Kangkang Lu, Zhiyong Huang, and Tat Seng Chua. 2024. Ee-lce: An event extraction framework based on llm-generated cot explanation. In *Knowledge Science, Engineering and Management*, pages 28–40, Singapore. Springer Nature Singapore.

A Training Details

Our models were trained on Google Cloud Vertex AI `a2-highgpu-1g` machines, equipped with NVIDIA A100 GPUs. We train using the Adafactor (Shazeer and Stern, 2018) optimizer, configured with a learning rate of 10^{-4} and a weight decay of 10^{-5}. Each model is trained for 10 epochs (with the best model selected using the ACE2005 dev set), and a batch size of 8 is used.

The total compute cost for running all of the training experiments was USD$559.81 for the experiments.

B Detailed Results

The scores shown in Figures 2 and 3 can be found in Tables 1 and 2, respectively. In the latter table, we highlight the best scoring run for each number of sentences used to chunk the document, as relaxed scores from runs with different sentence-breaking rules cannot be directly compared.

Model	# Sentences	Multi-Events Enabled	Event Detection	Argument Extraction
InstructUIE	1	N/A	0.441 ± 0.076	0.266 ± 0.158
EE-LCE	1	N/A	0.602 ± 0.111	0.234 ± 0.112
DEGREE (BART)	3	N/A	0.617 ± 0.089	0.377 ± 0.115
DEGREE (T5)	**1**	**Yes**	0.488 ± 0.033	0.402 ± 0.112
DEGREE (T5)	1	No	0.487 ± 0.039	0.396 ± 0.109
DEGREE (T5)	**2**	**Yes**	0.581 ± 0.042	0.398 ± 0.170
DEGREE (T5)	2	No	0.557 ± 0.073	0.391 ± 0.158
DEGREE (T5)	**3**	**Yes**	0.650 ± 0.034	**0.463 ± 0.135**
DEGREE (T5)	3	No	0.585 ± 0.040	0.403 ± 0.122
DEGREE (T5)	**4**	**Yes**	0.597 ± 0.074	0.415 ± 0.153
DEGREE (T5)	4	No	0.506 ± 0.072	0.361 ± 0.143
DEGREE (T5)	**6**	**Yes**	0.657 ± 0.028	0.421 ± 0.159
DEGREE (T5)	6	No	0.548 ± 0.047	0.389 ± 0.149
DEGREE (T5)	**8**	**Yes**	**0.658 ± 0.024**	0.420 ± 0.149
DEGREE (T5)	8	No	0.502 ± 0.031	0.348 ± 0.127

Table 1: Mean and sample standard deviations of the MUC-Style F1 scores for the five event types we analyze. Our configurations and the best scores are in bold.

Model	# Sentences	Multi-Events Enabled	Event Detection
InstructUIE	1	N/A	0.457 ± 0.074
EE-LCE	1	N/A	**0.615 ± 0.110**
DEGREE (BART)	3	N/A	0.650 ± 0.068
DEGREE (T5)	**1**	**Yes**	0.488 ± 0.033
DEGREE (T5)	1	No	0.499 ± 0.037
DEGREE (T5)	**2**	**Yes**	0.574 ± 0.052
DEGREE (T5)	2	No	**0.598 ± 0.078**
DEGREE (T5)	**3**	**Yes**	0.655 ± 0.038
DEGREE (T5)	3	No	**0.658 ± 0.047**
DEGREE (T5)	**4**	**Yes**	0.593 ± 0.083
DEGREE (T5)	4	No	**0.596 ± 0.094**
DEGREE (T5)	**6**	**Yes**	0.667 ± 0.046
DEGREE (T5)	6	No	**0.672 ± 0.057**
DEGREE (T5)	**8**	**Yes**	0.648 ± 0.032
DEGREE (T5)	8	No	**0.653 ± 0.060**

Table 2: Mean and sample standard deviations of the relaxed F1 scores for the five event types we analyze. Our configurations and the best scores (for each value of "# Sentences") is in bold.