Zero-shot Event Extraction via Transfer Learning: Challenges and Insights

Qing Lyu[1], **Hongming Zhang**[2*], **Elior Sulem**[1], **Dan Roth**[1]

[1]Department of Computer and Information Science, UPenn
[2]Department of Computer Science and Engineering, HKUST
{lyuqing, eliors, danroth}@seas.upenn.edu
hzhangal@cse.ust.hk

Abstract

Event extraction has long been a challenging task, addressed mostly with supervised methods that require expensive annotation and are not extensible to new event ontologies. In this work, we explore the possibility of zero-shot event extraction by formulating it as a set of Textual Entailment (TE) and/or Question Answering (QA) queries (e.g. "*A city was attacked*" entails "*There is an attack*"), exploiting pretrained TE/QA models for direct transfer. On ACE-2005 and ERE, our system achieves acceptable results, yet there is still a large gap from supervised approaches, showing that current QA and TE technologies fail in transferring to a different domain. To investigate the reasons behind the gap, we analyze the remaining key challenges, their respective impact, and possible improvement directions[1].

1 Introduction

Event extraction (EE) has long been an important and challenging NLP task. Figure 1 exemplifies a TRANSFER-OWNERSHIP event from the ACE-2005 dataset (Walker et al., 2006), where the *trigger* is "*purchased*" and the *arguments* include "*China*" (Buyer), "*Russia*" (Seller), etc. The subtasks of EE involve identifying and classifying event triggers and their corresponding arguments.

The predominant approaches normally require supervision (e.g. Lin et al., 2020), which is both expensive and inflexible when moving to new event ontologies. Recent works (Chen et al., 2020; Du and Cardie, 2020) have pointed out the connection between Question Answering (QA) and EE in developing supervised systems. Meanwhile, several efforts have explored unsupervised methods. Peng et al. (2016) first attempted to extract event *triggers* with minimal supervision using similarity-based heuristics. Huang et al. (2018) and Lai et al. (2020) explored both trigger and argument extraction under a slightly different setting: training on some event types and testing on unseen ones. Recently, Liu et al. (2020) proposed a QA-based zero-shot argument extraction method, which did not handle triggers. So far, no method has been proposed to extract *both* event triggers and arguments without any EE training data[2]. Moreover, the performance of existing zero-shot attempts, especially on arguments, is still far from satisfactory, yet little is known about possible underlying reasons.

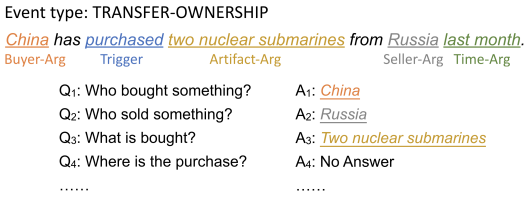

Figure 1: An example of an event from ACE-2005, and how arguments are extracted via QA.

In this work, we investigate the possibility of zero-shot EE via transfer learning from Textual Entailment (TE) and QA. Observe that given pretrained TE/QA models, extracting events can be viewed as answering questions/verifying hypotheses about a text. For example, the sentence in Figure 1, taken as the premise, would entail the hypothesis "*There is a transfer of ownership*", therefore providing the event type. Then, by asking Q_1 "*Who bought something?*", we obtain "*China*" as the Buyer. Similarly, Q_2, Q_3 will yield the Seller and Artifact, and so on.

Based on the observation above, we propose an intuitive zero-shot EE approach. It does not require any event training data, but we still make several design choices based on the development set. To demonstrate the level of generalization, we choose the optimal model with the ACE development set, and evaluate it on both ACE and ERE (LDC2015E29) test sets. The performance

[*] This work was done when the author was visiting the University of Pennsylvania.

[1]Our code and models will be available at http://cogcomp.org/page/publication_view/943.

[2]An exception is Zhang et al. (2021), done concurrently.

Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Short Papers), pages 322–332
August 1–6, 2021.

surpasses previous zero-shot approaches on every subtask when the gold trigger span is given, yet is still unsatisfying compared to supervised methods, revealing a large gap in using off-the-shelf TE/QA models for direct transfer. To shed light on why it is the case, we identify the key challenges behind the gap, and attribute each of them to the intrinsic weakness of pretrained models, our usage of them, or the task itself. We then anatomize their individual impact with an ablation study.

Our contributions are: (1) We propose the first TE/QA-based event extraction system that tackles *both* triggers and arguments without any event training data; (2) We show that existing TE/QA models do not support direct domain transfer well; and (3) We provide insights into the remaining challenges, their individual influence, and possible directions for future research.

2 Approach

Our pipeline consists of two modules, trigger extraction and argument extraction, both relying on pretrained TE/QA models for direct transfer.

The pretrained models we use are all BERT-based (Devlin et al., 2019; Liu et al., 2019; Lewis et al., 2020), including a TE model trained on MNLI (Williams et al., 2018), a Yes/No QA model trained on BoolQ (Clark et al., 2019), and an extractive QA model trained on QAMR (Michael et al., 2018) and/or SQuAD2.0[3] (Rajpurkar et al., 2018)[4]. The TE model, when given a *premise* and a *hypothesis*, predicts the relation between them ("entailment", "contradiction", or "neutral"). The Yes/No QA model takes as input a *context* and a *Yes/No question*, and returns either Yes or No. Finally, the extractive QA model is also given a *context* but with a *Wh-question*, and the answer is a span in the context. With these models, we design the two modules for event extraction.

2.1 Trigger Extraction (T-Ext)

We formulate Trigger Extraction (T-Ext) as a TE or a Yes/No QA task. Only the TE case is illustrated, since the other only differs in the query format.

To obtain potential event triggers from a sentence, we first run Semantic Role Labeling (SRL) as a preprocessing step. We use a BERT-based Verb+Nominal SRL model[5]. The sentence is then chunked into "text pieces", each containing an SRL predicate and its core arguments (e.g. A_0, A_1, A_2).

Then, for each text piece, we pass it to the TE model as the premise, coupled with a *hypothesis* in the format of "*This text is about ...*" for each event type, inspired by Yin et al. (2019). For example, the hypothesis for BE-BORN is "*This text is about someone's birth.*". Then, for each hypothesis, the model returns the probability that it is entailed by the premise. If the highest entailment probability across all event types surpasses a threshold, we output the corresponding SRL predicate as an event trigger of this type.[6]

Argument	Question
Artifact	"What is bought?"
Buyer	"Who buys something?"
Seller	"Who sells something?"
Price	"How much does something cost?"
Beneficiary	"Who is something bought for?"
Time	"When is the purchase?"
Place	"Where is the purchase?"

Table 1: The predefined question for each argument type in an TRANSFER-OWNERSHIP event.

2.2 Argument Extraction (A-Ext)

We formalize the task of Argument Extraction (A-Ext) as a sequence of QA interactions with the pretrained extractive QA model.

Given an input sentence and the extracted trigger, we ask a set of questions based on the event type definition, and retrieve the QA model's answers as argument predictions.

Consider the example in Figure 1. Assume that T-Ext has identified a TRANSFER-OWNERSHIP event with the trigger "*purchased*". With this information, we consult a predefined set of questions for each argument type in the current event type. For instance, Table 1 provides a full collection of questions for all arguments in TRANSFER-OWNERSHIP. Finally, to obtain the head of the argument (e.g. "submarines" in "two nuclear submarines"), we implement a simple heuristics-based head identifier based on the AllenNLP Dependency Parser[7] as a post-processing step.

An important caveat in the above process concerns missing arguments. Specifically, many argument types in the event template do not occur in every sentence, e.g. in Figure 1, there is no Place argument. For simplicity, we call questions with a non-empty gold answer "has-answer" (HA) ques-

[3] Abbreviated as SQuAD henceforth.

[4] See Appendix A and B for model and dataset details.

[5] https://github.com/CogComp/SRL-English

[6] See Appendix C.2 for configuration details.

[7] https://demo.allennlp.org/dependency-parsing

Setting	System	TI	TI+TC	AI	AI+AC
scratch (supervised)	Lin et al. 20	78.2	74.7	59.2	56.8
scratch (zero-shot)	Huang et al. 18[8]	55.6	49.1	**27.8**	15.8
	Zhang et al. 20	**58.3**	**53.5**	16.3	6.3
	Ours	45.5	41.7	27.0	**16.8**
gold TI (zero-shot)	Huang et al. 18	-	33.5	-	14.7
	Zhang et al. 20	-	82.9	-	-
	Ours	-	**83.7**	**38.9**	**24.2**
gold TI+TC (zero-shot)	Liu et al. 20	-	-	-	25.8
	Ours	-	-	**44.3**	**27.4**

Table 2: The F1 score on ACE-2005. Subtasks include Trigger Identification (TI), Trigger Classification (TC), Argument Identification (AI), and Argument Classification (AC). See Section 3 for setting definitions. SOTA results among zero-shot methods are in boldface.

tions and the rest "no-answer" (NA) questions. The QA model is considered to output NA when it predicts an empty span or the highest non-empty span confidence is lower than a threshold.

3 Experimental Setup

We evaluate our system on the ACE-2005 dataset. Its event ontology has 7 types and 33 subtypes, and we evaluate T-Ext directly on the subtypes. The same train/development/test split from Lin et al. (2020) is used. We make several design choices[9] on the development and report results on the test, ignoring the training set.

To demonstrate how our model generalizes, we also directly evaluate the optimal model on the ERE dataset (LDC2015E29). To adapt to ERE, we define a query for each new event type.

There are four subtasks of event extraction: Trigger Identification (**TI**), Trigger Classification (**TC**), Argument Identification (**AI**), and Argument Classification (**AC**). We experiment under three settings: **scratch**, where the system performs all subtasks without any gold annotation; **gold TI**, where gold trigger spans are given; **gold TC**, where gold trigger spans and types are given[10].

Following Ji and Grishman (2008), Precision, Recall, and F1 are used for evaluation[11]. We evaluate argument spans on the head level, consistent with most prior work (Huang et al., 2018; Wadden et al., 2019; Lin et al., 2020; Zhang et al., 2021).

4 Results

We report results in comparison with several existing zero-shot methods (Huang et al., 2018; Liu et al., 2020; Zhang et al., 2021), as well as a supervised SOTA system (Lin et al., 2020).

Setting	System	TI	TI+TC	AI	AI+AC
scratch (supervised)	Lin et al. 20	68.4	57.0	50.1	46.5
scratch	Ours	39.8	31.8	23.0	15.0
gold TI		-	58.4	30.8	18.8
gold TI+TC (zero-shot)		-	-	47.9	27.5

Table 3: The F1 score on the ERE. The optimal model is chosen on ACE dev and directly evaluated on ERE.

As shown in Table 2, on the ACE test set, our system outperforms prior zero-shot methods in every subtask under both the "gold TI" and "gold TI+TC" settings. However, it fails in "scratch", indicating that the main bottleneck lies in identifying exact trigger spans. Compared with the supervised SOTA, our system is still notably worse on TI, AI, and AC in particular, like other zero-shot systems.

Table 3 shows the results on ERE. Compared to ACE, our argument detection module generalizes well, whereas the trigger module does not. Under the gold TI setting, the TC F1 on overlapping event types is 70.4, whereas on new event types it is only 19.0, likely because the newly added event types in ERE have a finer definition. For example, a model needs to understand "whether a contact is in-person or not" to distinguish between MEET (in-person), CORRESPONDENCE (not in-person), and CONTACT (unsure). Further research should focus on how to effectively generalize to new event types with subtle definitions.

5 Analysis

Using the results on ACE, we now present an analysis of the remaining core challenges of the task, along with an ablation study on their individual impact. To further understand the challenges, we attribute each to the fragility of the pretrained *models* (**M-Error**), our *usage* of the models (**U-Error**), or the *task* itself (**T-Error**).

5.1 Trigger Extraction

5.1.1 Error Analysis

We first analyze the distribution of error types. Specifically, we manually check 100 wrong predictions and show the counts in Figure 2(a). Only the most frequent types are discussed here, and the remaining can be found in Appendix E.1.1.

Subtle trigger (M-Error): This is the main intrinsic error from the TE model (17%). Event types like DIE & EXECUTE, ATTACK & INJURE, and MEET & PHONE-WRITE are especially confus-

[8] Trained on 10 event types; tested on unseen ones.

[9] See Appendix C.2 and C.3.

[10] We don't have a **gold AI** setting, since the proposed QA-based A-Ext module cannot do AC alone.

[11] Evaluation scripts are adapted from `http://blender.cs.illinois.edu/software/oneie`.

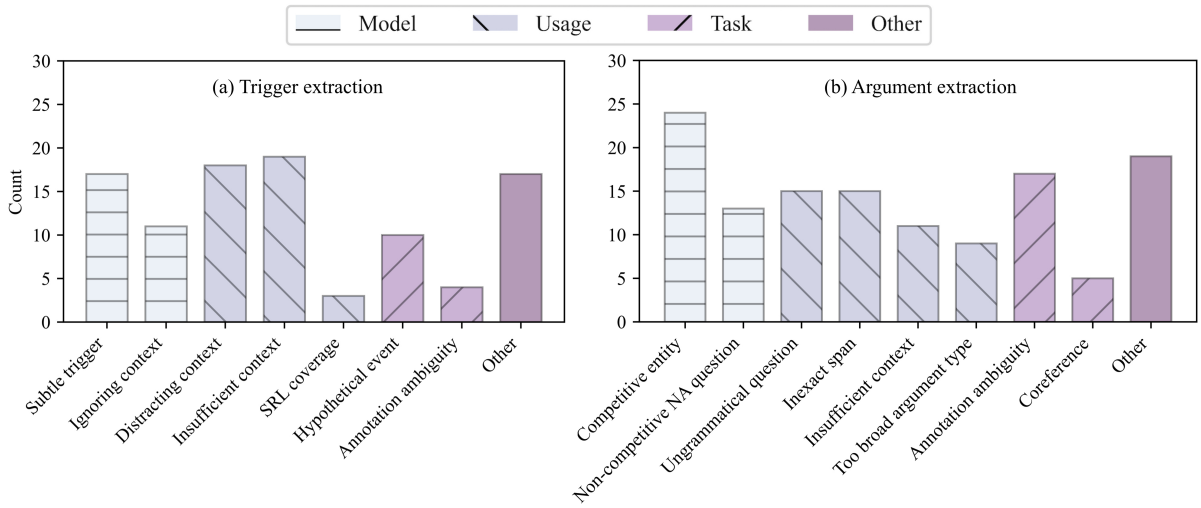

Figure 2: Error types in trigger and argument extraction in 100 wrong predictions. The count sum exceeds 100 since a prediction can contain multiple types of error. Colors/patterns indicate the origin of the error type.

ing. Though their definitions slightly differ, the model fails to capture this level of subtlety.

Distracting & Insufficient context (U-Error): Two other error types from our usage of the TE model concern distracting (18%) or insufficient (19%) contexts. An example of distracting context is "*The woman's parents ... found the decomposing body*". Given the word "decomposing", the model predicts it as a DIE event trigger, due to "*body*" in the premise. In contrast, insufficient contexts provide too little information. For example, in the sentence "*Turkey sent 1,000 troops ... and said it would send more*", the TE model is asked to predict the event type of "*send*" but only sees "*it send more*" as the premise, since "troops" is not part of the SRL arguments of "*send*". As a result, the model predicts a TRANSFER-MONEY instead of TRANSPORT event.

Hypothetical event & Annotation ambiguity (T-Error): Finally, two error types stem from the task itself: "hypothetical events" (10%) and "annotation ambiguity" (4%) . Hypothetical events refer to sentences like "*They will not buy it if it is too expensive*", where the TE model predicts "*buy*" as a TRANSFER-OWNERSHIP event trigger. Though such events should be annotated as per the ACE Annotation Guideline (3.4), this is not always strictly followed. Other cases of inconsistent annotation also cause errors, e.g. among all occurrences of "*give birth to*", the trigger is "*give*" in some cases, while "*birth*" in others.

5.1.2 Ablation Study

We further explore the two U-Error types, by measuring their influence on the performance while controlling for other factors. Only one type is included in this section, and the remaining can be found in Appendix E.1.2.

Premise design: To see the impact of **insufficient & distracting context**, we select all instances of these two types, and change the premise design. The re-prediction is done under gold-TI. For insufficient contexts, the premise is now the entire sentence. For distracting contexts, we adopt a "minimal-pair premise" strategy: Premise A is the original (e.g. "*...decomposing body...*"); Premise B is formed by deleting the candidate trigger from A (e.g. "*...body...*"). Then, we take the event type with the highest entailment probability *difference* between A and B as the prediction. Intuitively, this difference signifies the semantic *contribution* of the candidate trigger toward an event type.

After re-prediction, 59% errors are corrected on insufficient contexts. Among the remaining 41%, it is either the case that the model still ignores the context, or that the longer context now brings distraction.

On distracting contexts, only 18% errors are corrected. The model still cannot overcome the distraction in most remaining errors, which suggests that a more complicated strategy is needed in addition to manipulating the premise.

5.2 Argument Extraction

5.2.1 Error Analysis

Likewise, we analyze 100 wrong argument predictions and discuss several major error types. Figure 2(b) shows their respective counts. For a full explanation, see Appendix E.2.1.

Competitive entity & Non-competitive NA ques-

tions (M-Error): The QA model is intrinsically weak on "competitive entities" (24%) and "non-competitive NA questions" (13%).

When identifying an argument for the target event, another entity of the same type, i.e. a "competitive entity", can co-occur in the context. For example, the sentence "*A unit ... meets in confidential sessions to review terrorist activities in Europe*" has a MEET event. When asked "*Where is the meeting*", our model answers "*Europe*" whereas the gold answer is empty, since "*in Europe*" is attached to "*activities*". We find that models trained on extractive QA data are easily fooled by such entities, if they are of the desired type asked by the question. Note that competitive entities can occur for both HA and NA questions.

The other type involves NA questions without any competitive entity. For example, given the sentence "*Iraqi forces responded with artillery fire*", the question "*When is the fire*" has no answer, and there is no Time-type entity to distract the model. However, the model can still give arbitrary answers (e.g. "*artillery*") with very high confidence, due to its inherent incapacity for NA questions.

Ungrammatical question (U-Error): This relatively frequent error type (15%) is attributable to our usage of the QA model. To facilitate the model to better locate the target event, we embed the trigger in the questions whenever possible, which sometimes unavoidably makes them ungrammatical. For example, our question for the Place argument in a TRANSFER-OWNERSHIP event is "*Where is the* {*trigger*}". This is only grammatical when the trigger is a noun. Thus, the QA model may be confused by such questions.

5.2.2 Ablation Study

To isolate A-Ext, we perform the ablation study under the gold TI+TC setting. We explore four error types involving both M-Error and U-Error, two of which are included in this section, the rest in Appendix E.2.2.

Pretraining data: To examine the influence of **NA questions**, we compare QA models trained on QAMR (He et al., 2020) and SQuAD2.0, only the latter of which has NA questions. Results show that the one trained on QAMR greatly outperforms the one on SQuAD (+16.9 on AI; +13.6 on AC). To unveil why it is the case, we propose three hypotheses: (1) QAMR and ACE both have one-sentence contexts, while SQuAD has paragraphs. (2) The NA questions in SQuAD "confuses" the model, i.e. SQuAD and ACE have similar types of HA questions, while different types of NA questions. (3) The *density* of answers per sentence is high in both QAMR and ACE, while low in SQuAD. We test each hypothesis using controlled experiments, but none of them turns out to provide a full explanation of the performance difference[12].

Moreover, we train a binary classifier for HA and NA questions on a balanced sample of SQuAD, resulting in over 86 in-domain accuracy. On ACE, this number drops to 57. This shows that the QA model cannot even distinguish well between HA and NA questions when it comes to a new dataset, let alone answer them.

Question grammaticality: To see the impact of **ungrammatical questions**, we manually correct the grammatical error and re-predict with the model. Among all relevant wrong predictions, 40% are now correct. The rest 60% are mostly also NA questions that prove to require more than just fixing the grammar to solve.

6 Conclusions

We propose the first complete zero-shot event extraction system via transfer learning from TE and QA. While QA/TE models perform exceptionally well on standard benchmarks (SQuAD, QAMR, MNLI), they do not generalize as expected when being used on EE datasets. We analyze the limited success and several main challenges of the current approach, and provide insights for future improvements.

Acknowledgments

This work was supported in part by Contracts FA8750-19-2-1004 and FA8750-19-2-0201 with the US Defense Advanced Research Projects Agency (DARPA) and by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via IARPA Contracts No. 2019-19051600006 and 2019-19051600004 under the BETTER Program. The views expressed are those of the authors and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

We thank Celine Lee and Hangfeng He for providing the SRL and QAMR models respectively. We also thank Ying Lin, Jian Liu, Lifu Huang, Haochen Zhang, and the anonymous reviewers for their valuable help and/or feedback.

[12] Details can be found in Appendix E.2.2.

References

Yunmo Chen, Tongfei Chen, Seth Ebner, Aaron Steven White, and Benjamin Van Durme. 2020. Reading the manual: Event extraction as definition comprehension. In *Proceedings of the Fourth Workshop on Structured Prediction for NLP*, pages 74–83.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. Boolq: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2924–2936.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Xinya Du and Claire Cardie. 2020. Event extraction by answering (almost) natural questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 671–683.

Hangfeng He, Qiang Ning, and Dan Roth. 2020. Quase: Question-answer driven sentence encoding. In *Proc. of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

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

Heng Ji and Ralph Grishman. 2008. Refining event extraction through cross-document inference. In *Proceedings of ACL-08: Hlt*, pages 254–262.

Viet Dac Lai, Thien Huu Nguyen, and Frank Dernoncourt. 2020. Extensively matching for few-shot learning event detection. In *Proceedings of the First Joint Workshop on Narrative Understanding, Storylines, and Events*, pages 38–45.

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.

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

Jian Liu, Yubo Chen, Kang Liu, Wei Bi, and Xiaojiang Liu. 2020. Event Extraction as Machine Reading Comprehension. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1641–1651, Online. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Julian Michael, Gabriel Stanovsky, Luheng He, Ido Dagan, and Luke Zettlemoyer. 2018. Crowdsourcing question-answer meaning representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 560–568, New Orleans, Louisiana. Association for Computational Linguistics.

Haoruo Peng, Yangqiu Song, and Dan Roth. 2016. Event Detection and Co-reference with Minimal Supervision. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 392–402, Austin, Texas. Association for Computational Linguistics.

Vasin Punyakanok, Dan Roth, and Wen-tau Yih. 2008. The importance of syntactic parsing and inference in semantic role labeling. *Computational Linguistics*, 34(2):257–287.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for squad. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789.

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

Christopher Walker, Stephanie Strassel, Julie Medero, and Kazuaki Maeda. 2006. Ace 2005 multilingual training corpus. *Linguistic Data Consortium, Philadelphia*, 57:45.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American*

Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Wenpeng Yin, Jamaal Hay, and Dan Roth. 2019. Benchmarking zero-shot text classification: Datasets, evaluation and entailment approach. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3914–3923, Hong Kong, China. Association for Computational Linguistics.

Hongming Zhang, Haoyu Wang, and Dan Roth. 2021. Unsupervised label-aware event trigger and argument classification. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics (ACL) Findings*.

A Dataset Statistics

The pretraining datasets we use include MNLI (Williams et al., 2018), BoolQ (Clark et al., 2019), QAMR (Michael et al., 2018), and SQuAD2.0 (Rajpurkar et al., 2018). Our evaluation dataset is ACE-2005 (LDC2006T06) and ERE (LDC2015E29). Table 4 shows the number of examples in each dataset.

Dataset	Train	Dev	Test
MNLI	392,702	20,000	20,000
BoolQ	9,427	3,270	3,245
QAMR	73,561	27,535	26,994
SQuAD2.0	130,319	11,873	8,862
ACE-2005	17,172	923	832
ERE	-	-	2,069

Table 4: Number of examples in all datasets used.

B Details on Pretrained Models

We use three different pretrained representations, BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), and BART (Lewis et al., 2020). All models are implemented with HuggingFace Transformers[13].

The pretrained model checkpoints we use include: `bert-base-uncased` (110M parameters), `bert-large-uncased` (336M parameters), `roberta-base` (125M parameters), `roberta-large` (335M parameters), `facebook/bart-base` (373M parameters), `facebook/bart-large` (406M parameters)[14].

For TE and Yes/No QA, we finetune the pretrained models using the standard `SequenceClassification` pipeline. For extractive QA, we finetune the models using the `QuestionAnswering` pipeline[15]. The finetuning scripts are adapted from the `text-classification` and `question-answering` examples in the HuggingFace Transformers repository[16]. The hyperparameter values and pretrained models will be made available via the HuggingFace model sharing service.

We run our experiments on an NVIDIA GeForce RTX 2080 Ti GPU, with half-precision floating point format (FP16) with O1 optimization. The finetuning take 3 hours to 20 hours depending on the task.

C Details on Event Extraction System

We include here a full list of hyperparameter configurations explored in building our event extraction system. To select the optimal configuration, we perform grid-search on the development set based on the F1 score.

C.1 Preprocessing

We adapt the preprocessing script from Lin et al. (2020)[17]. In addition, we use several general-purpose NLP tools to further process the text, including a Part-of-Speech Tagger, a Dependency Parser, a Constituency Parser[18].

C.2 Trigger Extraction Module

Pretrained representation As said in Appendix B, we experiment with three representations (BERT, RoBERTa, and BART) with their base and large versions.

[13] https://github.com/huggingface/transformers

[14] All models above are available at https://huggingface.co/transformers/pretrained_models.html

[15] Both pipelines are available from https://huggingface.co/transformers/model_doc/

[16] https://github.com/huggingface/transformers/tree/master/examples/legacy

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

[18] The POS tagger is from http://www.nltk.org/; the rest are from https://demo.allennlp.org/.

Pretraining task We have two pretraining task choices, TE (using MNLI as training data) and Yes/No QA (using BoolQ as training data).
SRL constituents in the premise For each predicate, we only include itself and a few core arguments to form the premise. The combinations we try include: Predicate only; Predicate, Arg0, Arg1, Arg2; Predicate and all arguments.
Confidence threshold For an SRL predicate to be identified as an event trigger, we require that the confidence score of the TE model on the "Entailment" label (resp. the Yes/No QA model on the "Yes" label) exceeds a threshold. We search the threshold value within the range of $[0.80, 0.85, 0.90, 0.95, 0.99]$.
Hypothesis format We experiment with two strategies to phrase the hypothesis:

- *Topical*: The hypothesis is in the format of "*This text is about* {*topic*}", where the "{*topic*}" is predefined for each event type. For example, for ATTACK, the hypothesis is "This text is about an attack".

- *Natural*: The hypothesis is in a natural language format. For example, for ATTACK, it is "*Someone is attacked*"[19].

The optimal configuration for trigger extraction is:
- Pretrained representation: RoBERTa-large;
- Pretraining task: TE;
- SRL arguments in the premise: Predicate, Arg0, Arg1, Arg2;
- Confidence threshold: 0.99;
- Hypothesis format: Topical.

C.3 Argument Extraction Module

Pretrained representation As said in Appendix B, we experiment with three representations (BERT, RoBERTa, and BART) with their base and large versions.
Pretraining data We have two extractive QA datasets for pretraining, SQuAD2.0 and QAMR (and also their combination).
Question format We experiment with two question formats:

- *Static*: The questions are fixed for each event type. For example, the question for the Place argument in an ATTACK event is always "*Where is the attack?*".

- *Contextualized*: The questions are instantiated with the trigger of event instances when possible. For example, the question for the Place argument in an ATTACK event is "*Where is the* {*trigger*}*?*", where "{*trigger*}" is the specific trigger token(s) of the current event instance[20].

Confidence threshold For the extractive QA model to predict a non-empty answer, we require that its confidence score should be higher than a threshold. We search within the range of $[0.0, 0.1, 0.3, 0.5, 0.7, 0.9, 0.99]$.

The optimal configuration for argument extraction is:
- Pretrained representation: RoBERTa-large;
- Pretraining data: QAMR;
- Question format: Contextualized;
- Confidence threshold: 0.0 (the threshold value makes almost no difference, since most model prediction confidence scores are over 0.99).

D Full Results

Complementary to Section 4, Table 5 and Table 6 shows the full results including Precision, Recall, and F1 score on ACE and ERE respectively.

E Analysis (Continued)

This section elaborates on the remaining error types and ablation study experiments not covered by Section 5.

E.1 Trigger Extraction

E.1.1 Error Analysis

Ignoring context (M-Error): This is another prevalent error type (11%), which can also be attributed to the TE model. The model focuses too much on the candidate trigger itself while disregarding the context. Consider the sentence "*He was instrumental in creating such shows as 'married with children'...*". The word "*married*" is wrongly predicted as a MARRY event trigger. The TE model identifies it as an actual event rather than the name of a show.
SRL coverage (U-Error): Among all errors, 3% originate from the fact that the target trigger is not covered by SRL in the first place. This is a matter

[19] See the Supplemental Material for a list of all hypotheses.

[20] See the Supplemental Material for a list of all questions.

[21] Trained on 10 event types; tested on unseen ones.

Setting	System	TI			TI+TC			AI			AI+AC		
		P	R	F	P	R	F	P	R	F	P	R	F
scratch (supervised)	(Lin et al. 2020)	-	-	78.2	-	-	74.7	-	-	59.2	-	-	56.8
scratch (zero-shot)	(Huang et al. 2018)[21]	85.7	41.2	55.6	75.5	36.3	49.1	28.2	27.3	**27.8**	16.1	15.6	15.8
	(Zhang et al. 2020)	58.9	57.8	**58.3**	54.6	53.5	**54.0**	19.8	38.9	26.3	9.4	18.5	12.5
	Ours	34.7	66.3	45.5	31.7	60.6	41.7	20.2	40.4	27.0	12.6	25.2	**16.8**
gold TI (zero-shot)	(Huang et al. 2018)	-	-	-	-	-	33.5	-	-	-	-	-	14.7
	(Zhang et al. 2020)	-	-	-	-	-	82.9	-	-	-	-	-	-
	Ours	-	-	-	-	-	**83.7**	35.1	43.7	**38.9**	21.8	27.2	**24.2**
gold TI+TC (zero-shot)	(Liu et al. 2020)	-	-	-	-	-	-	-	-	-	25.5	26.0	25.8
	Ours	-	-	-	-	-	-	39.4	50.7	**44.3**	24.4	31.4	**27.4**

Table 5: The full performance on ACE-2005.

Setting	System	TI			TI+TC			AI			AI+AC		
		P	R	F	P	R	F	P	R	F	P	R	F
scratch (supervised)	(Lin et al. 2020)	-	-	68.4	-	-	57.0	-	-	50.1	-	-	46.5
scratch	Ours	34.5	68.2	45.8	30.2	59.7	40.1	18.2	37.9	25.1	12.1	24.3	16.1
gold TI		-	-	-	-	-	80.0	33.6	41.1	37.0	21.0	25.7	23.1
gold TI+TC (zero-shot)		-	-	-	-	-	-	39.4	50.6	44.3	24.4	31.3	27.4

Table 6: The full performance on ERE.

of our usage of the TE model. Specifically, current SRL systems cannot handle nominal triggers perfectly, and cannot detect multi-word triggers like "*step aside*" or adjectival triggers like "*dead*" at all.

Others: Other less-frequent error types besides those mentioned in the main text are related to coreference (e.g. when pronouns like "*this*" are triggers,), proper names (e.g. historical events like "*intifada*"), confidence scores being too low (thus not identifying a gold trigger), ambiguity of the hypothesis (e.g. a "*nuclear test*" is predicted as a TRIAL-HEARING event because of the word "*test*" and the hypothesis "*There is a trial or hearing*".)

E.1.2 Ablation Study

SRL models: To examine the influence of **SRL coverage**, we experiment with two more SRL models: Illinois SRL (Punyakanok et al., 2008)[22], and one that identifies almost every verb and nominal[23]. None of the three can identify adjectival/multi-word predicates. In comparison, every model can cover over 90% verb triggers, while the nominal trigger coverage varies from 60% to 95%. On T-Ext, the highest-coverage model performs the best (+4.0 F1 on TI, +6.8 on TC over the lowest-coverage model), proving that the gain from greedy identification does compensate for the cost in precision.

Pretraining task: Our results show that the TE-based TC far outperforms its Yes/No QA counterpart (by 52.6%). One hypothesis is that the pretraining data for the TE model (MNLI; about 400K examples) is much larger than that for the QA model (BoolQ; about 9K). To verify that, we retrain a TE model on a portion of MNLI of the same size as BoolQ. As a result, the gap shrinks to 31.4%, though still quite large. This proves the importance of the training data size. It also implies that in order to further improve the current TE-based method, using larger-scale training data might be promising.

Hypothesis design: It is observed that the hypothesis format also plays a nontrivial role. As said in Appendix C.2, we experiment with two hypothesis designs, *topical* and *natural*. Experiments show that "topical" is better than "natural" by 1.9% on TC, suggesting the sensitivity of current TE systems to the phrasing of texts.

E.2 Argument Extraction

E.2.1 Error Analysis

Too broad argument type (M-Error/U-Error): For this error type (9%), both the model and our usage are to blame. Though ACE has a strict definition of arguments, the QA model sometimes interprets them too broadly. For instance, with the context "*A blindfolded woman was shot in the head by a hooded militant*", given the question "*Where is the shot*", the model answers "*in the head*". This is not technically wrong, but certainly not the desired Place argument either. We cannot hold the QA model entirely accountable, since the questions

[22] https://cogcomp.seas.upenn.edu/page/software_view/SRL

[23] Also from https://github.com/CogComp/SRL-English.

are indeed too generic as well.

Inexact span (U-Error): 15% errors are because of the inexact match of gold and predicted argument spans. For instance, the gold is "*Saturday morning*" while the predicted is "*morning*". Though in our evaluation, we compare only heads of the phrases whenever possible, not all ACE arguments (i.e. those of the "value" type instead of the "entity" type) have head annotations. Under this circumstance, the current evaluation framework does not give credit to a partial match, which can be an imperfection for potential improvement.

Insufficient context (U-Error): Like in trigger extraction, the model is sometimes given insufficient context when predicting arguments (11%). The target argument can be entirely outside the SRL constituents of the predicate, thus making it impossible to extract.

Coreference & Annotation ambiguity (T-Error): Error types ascribed to the task include "coreference" (5%) and "annotation ambiguity" (17%). The former refers to the case when the model predicts a coreferent of the gold argument. However, the current evaluation framework still takes it as an error. The latter happens when the model makes a sensible prediction, yet it is inconsistent with the annotation. For example, in the sentence "*Iraqi forces responded with artillery fire*", the model recognizes "*artillery*" as the Instrument for the ATTACK event triggered by "fire". However, no Instrument is annotated. Future evaluation framework should consider allowing multiple correct answers in such cases of human disagreement.

Others: Other errors are related to multiple arguments (i.e. the model only predicts one of them), lacking document-level knowledge (i.e. the sentence itself is not informative enough), and also arbitrary predictions with no obvious reason.

E.2.2 Ablation Study

Pretraining data: Continuing from the "Pretraining data" paragraph in Section 5.2.2, we test three hypotheses for the gap between training on QAMR and SQuAD.

Hypothesis(1): QAMR and ACE both have one-sentence contexts, while SQuAD has paragraphs.

We try to verify it by retraining a QA model on a new version of QAMR with longer contexts, subject to the same length distribution of SQuAD. This is done by either a) adding random sentences, or b) repeating the original sentence. It is observed that a) almost doesn't hurt AI at all but AC a little (3%), and b) lowers AI by 4% and AC by 3%. Therefore, though longer contexts do weaken the performance slightly, it is not the main reason behind the gap between QAMR and SQUAD.

Hypothesis(2): The NA questions in SQuAD "confuses" the model, i.e. SQuAD and ACE have similar types of HA questions, while different types of NA questions.

To test this hypothesis, we retain all HA questions in SQuAD to make a new dataset. We also construct a control set of the same size, but with both NA and HA questions randomly sampled from the original SQuAD. We retrain a QA model on each dataset, and find that the HA-only set brings about an increment by 7% on AI but a drop by 2% on AC, compared to the control set. This suggests that the addition of NA questions in SQuAD does have mixed effects on event extraction. Future research should focus on how to better transfer a model's ability to identify NA questions to a different domain.

Hypothesis (3): The *density* of answers per sentence is high in both QAMR and ACE, while low in SQuAD.

To see if this is the cause, we construct a new version of QAMR by retaining only one QA pair for each sentence. A control set of the same size, but with multiple QA pairs per sentence, is also constructed by randomly deleting sentences (along with all their QA pairs) from the original QAMR. Results show that the low-density set is only worse than the control set on AI by 0.5% and on AC by 0.2%, indicating that the density of answers is not a critical aspect.

Type constraints in question: Since generic questions may have been a cause for **too broad argument types**, we experiment with a new set of question templates that contain specific entity-type requirements whenever possible. For example, instead of "*Where is the shot*", we ask "*What is the location of the shot*", which may prevent the model from answering "*in the head*". However, only 11% errors are fixed after re-prediction, indicating that encoding type constraints is non-superficial.

Question design: Like the hypothesis format in trigger extraction, the design of questions also makes a difference for arguments. As mentioned in Appendix C.3, we explore two formats, *static* and *Contextualized*. Experiments show that switching from "static" to "contextualized" boosts AI by 7%

while impairs AC by 3%, suggesting that contextualized questions overall helps the model better locate the event.

Context design: To measure the influence of *insufficient context*, we now use the entire sentence as the context on these instances, similar to trigger extraction. Results show that 27% of them are now correct, and another 27% are partially correct (inexact span).