Transfer Learning from Semantic Role Labeling to Event Argument Extraction with Template-based Slot Querying

Zhisong Zhang, Emma Strubell, Eduard Hovy
Language Technologies Institute, Carnegie Mellon University
zhisongz@cs.cmu.edu, strubell@cmu.edu, hovy@cmu.edu

Abstract

In this work, we investigate transfer learning from semantic role labeling (SRL) to event argument extraction (EAE), considering their similar argument structures. We view the extraction task as a role querying problem, unifying various methods into a single framework. There are key discrepancies on role labels and distant arguments between semantic role and event argument annotations. To mitigate these discrepancies, we specify natural language-like queries to tackle the label mismatch problem and devise argument augmentation to recover distant arguments. We show that SRL annotations can serve as a valuable resource for EAE, and a template-based slot querying strategy is especially effective for facilitating the transfer. In extensive evaluations on two English EAE benchmarks, our proposed model obtains impressive zero-shot results by leveraging SRL annotations, reaching nearly 80% of the fully-supervised scores. It further provides benefits in low-resource cases, where few EAE annotations are available. Moreover, we show that our approach generalizes to cross-domain and multilingual scenarios.

1 Introduction

Event argument extraction (EAE) is a key component in the task of event extraction (Ahn, 2006) that aims to identify the arguments that serve as roles for event frames. While recent developments in neural network models have enabled impressive improvements on this task in the fully-supervised setting (Wang et al., 2019b; Pouran Ben Veyseh et al., 2020; Ma et al., 2020; Li et al., 2021b), EAE remains challenging when abundant annotations are not available. In particular, event schemes are usually *specific* to the target scenarios. For example, events in biomedical domains, like GENE-EXPRESSION in GENIA (Kim et al., 2008), can be quite different than the ones in ACE (LDC, 2005), such as ATTACK and CONTACT. It is costly and inefficient to annotate large amounts of data for every new application.

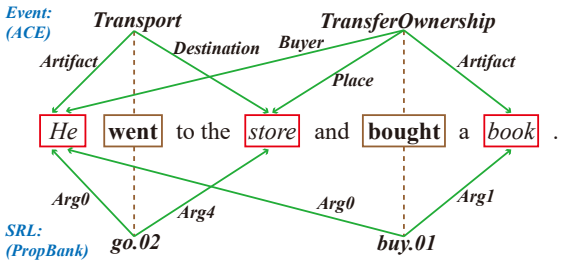

Figure 1: Example annotations with ACE events (above) and PropBank semantic frames (below). Brown and red rectangles indicate predicate and argument words, respectively. Green lines denote argument links.

Compared with the *specific* schemes in EAE, semantic role labeling (SRL; Gildea and Jurafsky, 2002; Palmer et al., 2010) extracts predicate-argument structures with more *general* and broad-coverage frame ontologies. SRL also enjoys rich and carefully annotated resources, such as PropBank (Palmer et al., 2005) and FrameNet (Baker et al., 1998), covering a wide range of semantic frame types. As shown in the example in Figure 1, SRL closely resembles EAE: they both specify semantic frames triggered by predicate words and aim at finding arguments for participating roles. Therefore, it is natural to consider applying transfer learning[1] (Pan and Yang, 2009; Ruder et al., 2019) to enhance EAE with general SRL resources.

Notwithstanding the similarities, there are two main discrepancies between SRL and EAE structures that should be managed in order to facilitate transfer between them. The first is *label mismatch*. For example, ACE adopts role names with natural language words, such as BUYER and PLACE, whereas PropBank utilizes generalized labels like ARG0 and ARGM-LOC. PropBank also provides

[1]Because of the similarities, EAE and SRL may be arguably viewed as two versions of the same task. Even in this case, we can still view this as a special form of transductive transfer learning, if not inductive transfer on different tasks.

Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, pages 2627–2647
December 7-11, 2022

more specific role descriptions, but many are inconsistent and not well-formed for direct use as role names. Although FrameNet also adopts natural language role names, it is laborious and sometimes challenging to find all the direct mappings to the target event frames. Moreover, SRL resources do not typically annotate *distant arguments*, where there are no explicit syntactic encodings expressing the argument relation.[2] For example, in the sentence depicted in Figure 1, though it can be understood that the "store" is very likely to be the place where the "buying" happens, SRL annotations do not include this semantically inferred link, whereas it is considered an argument in event annotations.

Although there have been previous works utilizing SRL for argument linking (O'Gorman, 2019), it remains unclear how to best directly transfer from SRL to EAE, especially with recent pre-trained models. In this work, we provide a comprehensive investigation on the transfer from SRL to EAE. We view the tasks as a role querying problem within a *unified framework*, which covers varioius different argument extraction methods, including classification-based methods (Ouchi et al., 2018; Ebner et al., 2020), machine reading comprehension (MRC)-based methods (Liu et al., 2020; Du and Cardie, 2020; Li et al., 2020; Feng et al., 2020; Lyu et al., 2021; Liu et al., 2021a) as well as sequence-to-sequence generation based ones (Li et al., 2021b; Hsu et al., 2022; Lu et al., 2021). We further explore a template-based slot querying strategy, by querying argument roles using contextualized representations of the corresponding role slots in the frame template. We tackle the label-mismatch problem by forming the queries in templated natural language, which allows for the same query representation to be shared across varied schemes. To mitigate the lack of distant argument annotations in SRL, we apply two argument augmentation techniques: Data augmentation by shuffling input texts, which reduces the model's reliance on local syntax, and knowledge distillation from question answering (QA) data, which incorporates distant argument signals.

With experiments on the standard ACE and ERE English event benchmarks, we show that SRL annotations are valuable resources for EAE. With the template-based querying strategy, a model trained with SRL can reach nearly 80% of the fully-supervised F1 score in the zero-shot scenario, and an intermediate-training scheme provides further benefits in the low-resource setting. The model also obtains promising results in extensions to cross-domain and multi-lingual scenarios, demonstrating its generalizability. Our work highlights the utility of SRL annotations in the context of downstream applications with limited direct annotations.

Our implementation is available at https://github.com/zzsfornlp/zmsp/.

[2]These are also known as *implicit arguments* (O'Gorman, 2019). While there are more fine-grained linguistic criteria, we take a simplified approximate approach by checking the syntactic distances between triggers and arguments.

2 Method

2.1 Querying Methods

For either semantic roles or event arguments, we can view the extraction task as a role querying problem. Specifically, we are given a sequence of words $\mathbf{s} = \{w_1, ..., w_n\}$ as input contexts as well as a predicate or event trigger word w_e and the semantic frame or event type t. Each type is associated with a list of participating roles to be filled and the task is to extract arguments from the input contexts for each role. We adopt one specific modeling simplification, that is, our model only predicts the syntactic head word of an argument. For EAE, a heuristic method is further adopted to expand from head words to spans: We simply include the head word's child that is linked with an MWE dependency relation[3] and has an uppercase first letter. We find that this heuristic works well in practice, expanding correctly to 95% of the argument spans in the ACE and ERE event datasets. We take this approach to make it easier to transfer across different schemes, which may have different annotation criteria on argument spans.

In this way, we can view both SRL and EAE as role querying problems over the input words (all the queries depend on the predicates, which we assume given and omit for brevity). Specifically, the probability of a candidate word w to be the argument filling a role r is:

$$p_r(w) = \frac{\exp(\lambda \mathbf{h}_w^T \mathbf{q}_r)}{\sum_{w' \in \mathbf{s} \cup \{\epsilon\}} \exp(\lambda \mathbf{h}_{w'}^T \mathbf{q}_r)}$$

Here, $\mathbf{h}_w$ denotes the representation vector of the word w, and $\mathbf{q}_r$ indicates the querying vector of the role r. We further include a scaling factor λ, which is fixed to $\frac{1}{\sqrt{d}}$, where d is the dimension of $\mathbf{h}$ and $\mathbf{q}$, following the attention calculation in Transformer

[3]Multi-word expressions: {"fixed", "flat", "compound"}.

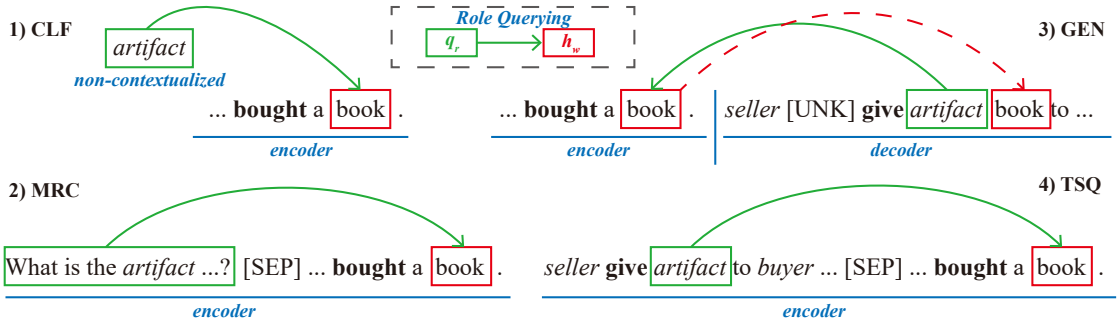

Figure 2: Illustrations of different role querying strategies (for the "artifact" role), based on 1) CLF: classification, 2) MRC: machine reading comprehension, 3) GEN: generation, and 4) TSQ: template-based slot querying.

(Vaswani et al., 2017). We specify a dummy token ϵ to handle the cases where no arguments can be found for a role. This modeling scheme is flexible and allows different argument extraction strategies to be viewed *in a unified way*. In this work, we explore four strategies, as illustrated in Figure 2. Since these strategies are not totally novel, we give brief descriptions in the main content and refer the reader to Appendix A.1 for more details.

1) CLF. We start with querying based on traditional classification, which assigns to each role a *non-contextualized* vector. To allow transfer to different role names, we initialize the role vectors with average-pooled representations obtained by passing the role names individually to a pre-trained language model. We call this strategy classification-based since the role vectors can be viewed as weights in a linear classifier. This corresponds to more traditional argument extraction methods (Ouchi et al., 2018; Ebner et al., 2020). One shortcoming of this strategy is that the query vectors are constructed without access to input contexts, limiting their representation ability.

2) MRC. Recently, the strategy of casting NLP tasks as machine reading comprehension problems (Rajpurkar et al., 2016, 2018) has been applied to EAE (Liu et al., 2020; Du and Cardie, 2020; Li et al., 2020; Feng et al., 2020; Lyu et al., 2021; Liu et al., 2021a). In this strategy, each role is queried with a *contextualized* question that is encoded together with the context. Unless otherwise specified, we form the role questions using the templates of Liu et al. (2021a), which can be automatically generated from the role names. Since each question queries only one role, this strategy requires a full pass through the encoder for each role, raising concerns regarding its computational efficiency,[4] as compared to CLF.

3) GEN. More recently, many approaches extract arguments by sequence-to-sequence generation (Paolini et al., 2021; Li et al., 2021b; Hsu et al., 2022; Lu et al., 2021; Du et al., 2021; Huang et al., 2022). Specifically, Li et al. (2021b) and Hsu et al. (2022) adopt a template-based generation strategy, which aggregates the queries of all roles for an event into one *template* sentence (or *bleached statement* (Chen et al., 2020)). This strategy is promising since the template can contain all roles and query them in one pass. Since arguments come from input contexts, we further adopt a pointer network (Vinyals et al., 2015) for argument selection rather than generating through output vocabularies, fitting our unified querying framework. Because of the auto-regressive decoding scheme, this strategy can also suffer lower efficiency compared to CLF.

4) TSQ. We further explore a strategy that fully exploits the representational power and querying efficiency of templates. We do not fill the templates with actual words in the context but simply keep the role names as placeholders. We concatenate this template with the context, then pass the full sequence to the encoder for contextualization. Finally, the contextualized representations of the role slots in the template are adopted as role query vectors. We refer to this strategy as Template-based Slot Querying (TSQ). This approach is similar to the contemporaneous work of Ma et al. (2022). Our approach to template querying differs primarily in that: 1) We concatenate both the template and the context and feed them to the encoder, allowing for bidirectional modeling, and; 2) Our models predict argument head words rather than spans to facilitate the transfer since unlike Ma et al. (2022) our focus is transfer learning.

[4]Please refer to Appendix B.8 for speed comparisons.

Scheme	Frame	Template
PropBank	forbid.01	authority **forbid** protagonist action in place
	rent.01	at lessor lessee **rent** goods from lessor for money in place
	swim.01	from area self mover **swim** against area to goal in place
FrameNet	Abandonment	agent **abandon** theme in place
	Employing	employer **employ** employee field position task in place
	Mention	communicator **mention** specified content message in medium in place

Table 1: Examples of the semi-automatically generated templates. The predicate is **emboldened** and roles are underlined. Some examples are specially picked to show typical problems of this noisy process.

2.2 SRL Templates

We take PropBank[5] (Palmer et al., 2005), NomBank[6] (Meyers et al., 2004) and FrameNet[7] (Baker et al., 1998) as our main SRL resources. Since many NomBank frames are derived from PropBank frames, we simply map them to the PropBank counterparts (by checking the "source" attribute in a NomBank frame) and ignore the ones that do not have such mappings. We filter event-related SRL frames by excluding the ones that do not have any verb realizations, which are judged by the POS sets provided in the frame files. Moreover, we only consider a subset of non-core or modifier roles that are related to the target EAE task, including ARGM-LOC in PropBank and {PLACE, INSTRUMENT, WEAPON, VEHICLE} in FrameNet.

To allow transfer across different schemes, we need to specify extra information required by the role querying strategies. In particular, templates are not included in SRL frame definitions and it is infeasible to manually specify them for hundreds to thousands of SRL frames. We adopt a semi-automatic method to construct the templates, with extra information collected from data statistics:

- **Role names.** We directly take the role label names of FrameNet since they are already in natural language forms. We further train[8] a role label classifier[9] with the FrameNet data and apply it to the PropBank data. Then for each frame-specific role, the most frequently predicted label will be its role name. For example, for the ARG0 role of the "buy.01" frame, its arguments in the dataset are mostly predicted to the BUYER label, which is thus assigned as its role name.
- **Role orders.** We construct a template for an SRL frame by concatenating its predicate word and role names. The main thing to specify is their ordering. We again take a statistical approach and collect each role's relative distance to the predicate. For example, in the "buy.01" frame instance of "He bought a book in a store.", ARG0 (He) gets a distance of -1, ARG1 (book) gets a +1 and ARGM-LOC (store) gets a +2. Finally, the role orders in the templates are decided by the roles' average relative distances. We aim to obtain a canonical verb-styled ordering in active voice, and thus we only consider frame instances that are realized by non-passive verbal predicates.
- **Preposition words.** When realized in natural language sentences, many roles are accompanied by prepositions. We count the frequency that a role is filled by an argument that utilizes a preposition[10] and keep the prepositions that appear more frequent than 25%. When there are such prepositions, we add the preposition before the role name and put them together into the slot. When there are multiple feasible prepositions, we randomly sample one in training and utilize the most frequent one in testing.

With these three types of extra information, we construct the templates by concatenating all the corresponding ordered pieces. For example, the "buy.01" PropBank frame gets a template of "buyer **buy** goods for recipient from seller for money in place". Most of the above heuristics are decided by manually checking the generated outputs for the PropBank and FrameNet frames.

[5] https://github.com/propbank/propbank-frames/releases/tag/v3.1

[6] https://nlp.cs.nyu.edu/meyers/nombank/nombank.1.0/

[7] https://framenet.icsi.berkeley.edu/fndrupal/frameIndex

[8] Another option could be to use existing resources that connect PropBank and FrameNet, such as SemLink (Palmer, 2009; Stowe et al., 2021). Nevertheless, their coverage is still slightly lacking and we thus take a data-driven method, which could map every frame that has data.

[9] This classifier is similar to our CLF querying model except that no extraction is needed. Its accuracy on the FrameNet dev set is around 0.7. Notice that even when the classifier does not hit the most suitable label, the predicted ones may still be reasonable for our usage.

[10] The criterion is that the argument's head word has a dependency relation of "case" to a child whose POS is "ADP".

Notice that this semi-automatic approach is far from perfect and there can be noises and inconsistencies, as shown in some of the examples in Table 1. Nevertheless, the above three pieces provide complementary information for role specification: the role names provide semantic information, the role orders include syntactic word order information, and the prepositions give further hints. In practice, we find that most of the generated templates are reasonably close to natural language. In this way, we are able to form similar queries for both SRL and EAE, tackling the label mismatch problem between different frame schemes.

2.3 Argument Augmentation

In addition to label mismatch, another discrepancy between SRL and EAE is that arguments in traditional SRL are syntactically constrained whereas event arguments can be extracted from any place in the context. Therefore, SRL models will have difficulties in predicting syntactically distant arguments. To mitigate this problem, we apply data augmentation (Feng et al., 2021) and knowledge distillation (Hinton et al., 2015) to augment distant arguments for SRL instances.

Firstly, we apply a simple data augmentation method by shuffling the input contexts. Since the SRL arguments are constrained by syntax, we hypothesize that by distorting syntax in some way, the model can be trained to focus more on the semantic relations between the predicates and arguments. This may allow it to predict more distant arguments. To distort syntax, we randomly chunk the input sentence with sizes randomly chosen from one to three at each time. Then these text chunks are shuffled, re-concatenated, and fed to the pre-trained model for contextualized encoding. Since our model only selects argument head words, there is no change to the later processing except for word position re-indexing. We only apply this procedure during training and simply mix vanilla unshuffled data with the shuffled ones by a 1:1 ratio.

Moreover, we seek signals of distant arguments from question answering (QA)[11] datasets, such as SQuAD (Rajpurkar et al., 2016, 2018). In QA annotations, the answers are not constrained by syntax and can be freely picked from the full context, providing valuable resources for distant arguments (Liu et al., 2021a). Motivated by this, we train a QA model with the MRC strategy and predict the missing arguments for SRL instances. Instead of hard predictions, we store a soft probabilistic distribution over the context words for each role and utilize these for SRL training with a standard cross-entropy objective:

$$\mathcal{L}_{\text{distill}}(r) = -\sum_{w \in \mathbf{s} \cup \{\epsilon\}} p_r^{\text{qa}}(w) \log p_r^{\text{m}}(w)$$

Here, for the querying of each role r, $p_r^{\text{qa}}(w)$ denotes the argument probabilities among the context words according to the QA model, while $p_r^{\text{m}}(w)$ indicates the current model's outputs. To avoid noise from the QA predictions, we adopt two filters. Firstly, we only apply distillation for the unfilled roles according to SRL annotations. This is intuitive since the filled roles already have gold annotations. Moreover, we apply distillation only when the prediction is confident enough. We perform calibration to the QA model by temperature scaling (Guo et al., 2017) and adopt a probability threshold of 0.5. In this way, we could borrow the signals of distant arguments from the QA datasets to enhance SRL instances with potential missing distant arguments.

3 Experiments

3.1 Settings

We conduct our main experiments with English ACE[12] (Walker et al., 2006) and ERE (LDC, 2015) event datasets. We adopt the preprocessing scripts[13] from OneIE (Lin et al., 2020). For the target event frames, we manually specify extra information such as templates, adopting those of Li et al. (2021b). Unless otherwise specified, we assume that gold event triggers are given and focus on the extraction of event arguments. We also provide results with predicted event triggers in Appendix B.3. We evaluate arguments by labeled F1 scores, which require both argument spans and roles to match the gold ones. We run with five random seeds and report average results.

For external data, we take PropBank, NomBank 1.0, and FrameNet 1.7 as our main SRL resources. We prepare the SRL templates by the semi-automatic process described in §2.2. For QA datasets, we take SQuAD 2.0 (Rajpurkar et al., 2018), QA-SRL 2.1 (FitzGerald et al., 2018),

[11] Specifically we adopt the extractive QA-MRC data. To avoid confusion, we use "QA" when denoting data resources while using "MRC" for the querying strategy.

[12] We adopt ACE05-E$^+$ (Lin et al., 2020).

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

Method	ACE P%	ACE R%	ACE F1%	ERE P%	ERE R%	ERE F1%
Super.	$68.93_{\pm 1.07}$	$68.94_{\pm 0.95}$	$68.93_{\pm 0.95}$	$72.75_{\pm 1.69}$	$71.80_{\pm 1.29}$	$72.24_{\pm 0.34}$
GPT-3	29.10	34.25	31.47	25.09	26.76	25.90
QA	$32.77_{\pm 3.70}$	$47.43_{\pm 1.17}$	$38.62_{\pm 2.58}$	$32.68_{\pm 2.78}$	$48.13_{\pm 4.08}$	$38.74_{\pm 2.09}$
SRL_{CLF}	$47.97_{\pm 1.47}$	$25.37_{\pm 0.86}$	$33.18_{\pm 0.92}$	$50.17_{\pm 1.72}$	$25.60_{\pm 0.65}$	$33.89_{\pm 0.88}$
SRL_{MRC}	$58.27_{\pm 0.75}$	$39.54_{\pm 1.60}$	$47.08_{\pm 0.89}$	$62.02_{\pm 1.15}$	$45.31_{\pm 1.74}$	$52.32_{\pm 0.83}$
SRL_{GEN}	$55.77_{\pm 0.61}$	$45.31_{\pm 1.26}$	$49.99_{\pm 0.93}$	$58.37_{\pm 0.66}$	$52.68_{\pm 0.63}$	$55.38_{\pm 0.62}$
SRL_{TSQ}	$57.74_{\pm 0.95}$	$49.61_{\pm 0.80}$	$53.36_{\pm 0.53}$	$59.93_{\pm 0.68}$	$55.84_{\pm 0.78}$	$57.81_{\pm 0.34}$
+shuf.	$58.36_{\pm 0.53}$	$51.70_{\pm 0.52}$	$54.82_{\pm 0.44}$	$59.70_{\pm 0.89}$	$57.42_{\pm 1.26}$	$58.54_{\pm 1.05}$
+distill	$54.53_{\pm 0.97}$	$55.85_{\pm 0.67}$	$55.17_{\pm 0.42}$	$55.27_{\pm 0.72}$	$60.90_{\pm 0.85}$	$57.95_{\pm 0.65}$
+both	$55.68_{\pm 1.26}$	$57.04_{\pm 0.93}$	$\mathbf{56.35}_{\pm 1.07}$	$56.63_{\pm 0.78}$	$61.48_{\pm 0.18}$	$\mathbf{58.96}_{\pm 0.48}$

Table 2: Zero-shot EAE results on event test sets. Except for GPT-3, all results are averaged over five runs.

QANom (Klein et al., 2020) and QAMR (Michael et al., 2018). For the training of SRL or QA models, we simply adopt the concatenation of all the corresponding datasets. Except for those that have manual syntactic annotations, we utilize Stanza (Qi et al., 2020) to parse the texts to obtain the syntactic head words of the arguments.

We adopt pre-trained language models for initialization and fine-tune the full models during training. Specifically, we use $\text{RoBERTa}_{\text{base}}$ (Liu et al., 2019) for encoder-only models (CLF, MRC, TSQ) and $\text{BART}_{\text{base}}$ (Lewis et al., 2020) for encoder-decoder models (GEN). Please refer to Appendix B.1 for more detailed experimental settings.

3.2 Main Transfer Experiments

We conduct our main experiments with English ACE and ERE datasets. Thanks to the unified querying framework, we can conduct experiments in a zero-shot setting (§3.2.1), where models trained on external data are directly evaluated on EAE. We also investigate low-resource settings where some amounts of EAE annotations are available for further fine-tuning (§3.2.2).

3.2.1 Zero-shot

In the zero-shot setting, we further compare with two methods in addition to SRL: 1) GPT-3 (Brown et al., 2020), where we form prompts[14] for each role and use GPT-3 to generate the answers; 2) QA, where we train QA models[15] with the QA datasets. We also provide the fully-supervised results[16] (Super.) as references.

[14]Please refer to Appendix B.2 for more GPT-3 details.

[15]Notice that we can only use the MRC strategy for QA models because of the task-specific format.

[16]We take those of the TSQ model. More details of the supervised results are provided in Appendix B.5.

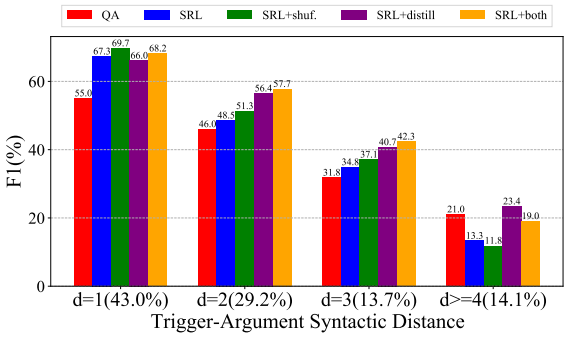

Figure 3: Breakdowns on trigger-argument syntactic distances (on ACE dev set). Numbers in the parentheses denote the percentages in the gold annotations.

Results The main results are shown in Table 2. Except for the one with the CLF strategy, SRL models perform generally better than QA and GPT-3, showing the effectiveness of utilizing SRL resources. Among the SRL models, the TSQ strategy generally performs the best, indicating the effectiveness of this contextualized querying strategy. Further improvements can be obtained with the argument augmentation techniques. Interestingly, if only using shuffling augmentation (+shuf.), precision remains roughly the same while recall increases. If only using distillation (+distill), recall increases but at the expense of precision. Finally, if both are utilized (+both), precision and recall both improve relative to the distillation-only case. This leads to the overall best F1 scores, reaching around 80% of the supervised results.[17]

Analysis As shown in Figure 3, we further perform breakdowns on the syntactic distances between triggers and arguments. We especially compare the QA model and the four SRL_{TSQ} models. Firstly, the QA model performs worse than SRL

[17]Please refer to Appendix B.6 for manual analysis.

models except for the long distant ones ($d \geq 4$). This is due to SRL annotations mainly capturing syntactically local arguments while QA is not constrained by this. Within the SRL models, when adding shuffling (+shuf.) or distillation (+distill), the middle-ranged arguments consistently obtain improvements. One interesting pattern is that shuffling benefits $d = 1$ but hurts $d \geq 4$, while distillation seems to have the opposite effects. This may indicate that shuffling enhances more robust predictions of short- and middle-ranged arguments while distillation encourages longer-ranged ones. Finally, when combining these two techniques (+both), the model can reach a good balance, achieving the best overall results. Due to its overall better performance, we use the "SRL_{TSQ}+both" strategy for our SRL models in the remainder of this work.

3.2.2 Low-resource

We further investigate scenarios where we have some amount of target EAE annotations. With target data, we can directly train an EAE model (from pre-trained language models). We further apply a simple intermediate-training scheme (Phang et al., 2018; Wang et al., 2019a) to transfer the knowledge from SRL. We take the SRL-trained model and further fine-tune it on the target event data. A similar scheme can also be adopted with the QA model. Figure 4 shows the results with different amounts of training instances. Generally, SRL intermediate training is beneficial, especially for middle- and low-resource cases, again showing that SRL annotations can be valuable transfer sources for the extraction of event arguments. Note that when using full target data, the external SRL data is less helpful. We think this is probably because there is already enough supervision to learn most of the target patterns, and there might be less further information that SRL could provide beyond the already rich target resources.

3.3 Further Extensions

In the previous experiments, we take ACE and ERE as the targets, which are still relatively similar to the SRL annotations. In this sub-section, we further investigate scenarios where there are larger discrepancies between the source and the target. Specifically, we examine the transfer from SRL to EAE in cross-domain (§3.3.1), multi-lingual (§3.3.2) and multi-sentence (§3.3.3) cases.

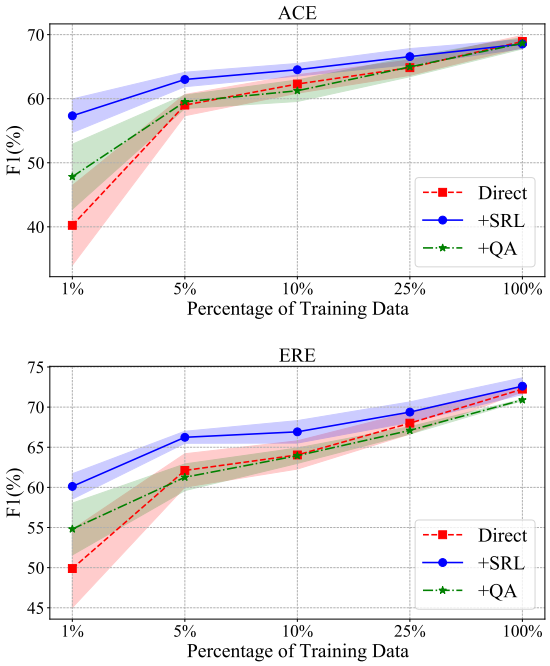

Figure 4: Model performance with direct or intermediate training. Here x-axis (drawn in log scale) denotes the percentage of utilized training data. The shaded areas indicate the ranges of standard deviations.

3.3.1 Cross-domain

We first investigate the biomedical domain, utilizing the GENIA BioNLP-11 benchmark (Kim et al., 2011). The GENIA events are quite different than general SRL frames and mainly describe detailed bio-molecule behavior (Kim et al., 2008). Still focusing on the argument extraction step, we take the event triggers predicted by the supervised system BEESL (Ramponi et al., 2020). We perform zero-shot argument extraction and evaluate the QA and SRL models, with manually compiled role questions and templates. We adopt the official evaluation metric of approximate recursive matching. Please refer to Appendix C.1 for more details.

Our main comparison is between the QA and SRL models, while we also include the supervised results of BEESL as references. We further adopt a self-training approach to adapt to the target domain. Specifically, we take the texts from the original GENIA training set, ignore the original labels, predict SRL frames on these texts with our SRL model and train a final SRL model with both these predicted structures and external SRL resources.

The results on the test set are shown in Table 3. SRL generally outperforms QA for most of the types. This may be due to the difficulty of asking proper questions. For example, for the "Regula-

Types	QA	SRL	SRL_{+self}	Super.
Expression	70.71	76.66	77.59	80.90
Transcription	63.64	55.63	59.72	69.46
Catabolism	62.07	66.67	66.67	74.07
Phosphorylation	75.95	78.98	83.64	89.52
Localization	53.28	66.89	67.33	69.51
– **Simple** –	68.26	73.63	75.27	79.31
Binding	39.41	34.90	35.10	50.19
Regulation	33.80	38.95	38.52	45.90
Pos. regulation	31.85	38.96	39.95	49.41
Neg. regulation	36.62	44.84	44.51	47.17
– **Complex** –	33.36	40.44	40.88	48.32
– **All** –	47.42	51.95	52.76	60.22

Table 3: BioNLP-11 event extraction results (F1%).

tion" event, we ask "What is regulated?" for the role of "Theme" and "What causes the regulation?" for "Cause". These questions may be unrelated to the actual contexts, while for the SRL models, extra hints from the query templates may be helpful. This may also explain why QA is better on some of the types where it is relatively easy to ask questions. For example, for "Transcription", the question "What is transcribed?" would be accurate for most contexts. For the SRL models, the self-training method is beneficial overall, showing the effectiveness of utilizing unlabeled corpus from the target domain.[18] Finally, our best zero-shot model could recover more than 80% of the overall performance of the supervised model, showing that general SRL resources can still be helpful in the biomedical domain. The main gaps between the zero-shot and supervised systems are in the "Binding" and "Complex" events where there are complicated and even nested structures. One future direction is to investigate ways to better handle these complex structures.

3.3.2 Multi-lingual

We next explore a multi-lingual setting, taking ACE05 Arabic and Chinese datasets as our targets. We follow Huang et al. (2022) and utilize their pre-precessing scripts[19] for data preparation[20]. We further include multi-lingual external resources. For SRL, we utilize Arabic and Chinese PropBank annotations from OntoNotes (Hovy et al., 2006; Weischedel et al., 2013). For the role names in SRL frames, we again adopt a statistical approach:

Model	Arabic	Chinese
Zero-shot results without any EAE annotations.		
QA_{en}	$22.56_{\pm 1.48}$	$26.58_{\pm 2.61}$
QA_{en+tgt}	$23.54_{\pm 1.43}$	$27.08_{\pm 1.79}$
SRL_{en}	$37.75_{\pm 0.52}$	$39.37_{\pm 1.45}$
SRL_{en+tgt}	$\mathbf{40.64}_{\pm 1.49}$	$\mathbf{41.50}_{\pm 1.04}$
Multi-lingual results with English EAE annotations.		
GATE†	44.5	49.2
X-Gear†	44.8	54.0
En_{MRC}	$37.44_{\pm 3.02}$	$51.86_{\pm 0.92}$
$+QA_{en}$	$39.06_{\pm 2.86}$	$53.36_{\pm 1.06}$
$+QA_{en+tgt}$	$44.27_{\pm 1.37}$	$53.97_{\pm 1.41}$
En_{TSQ}	$37.64_{\pm 1.96}$	$53.54_{\pm 0.65}$
$+SRL_{en}$	$41.86_{\pm 0.92}$	$53.96_{\pm 0.85}$
$+SRL_{en+tgt}$	$\mathbf{51.51}_{\pm 1.32}$	$\mathbf{58.90}_{\pm 0.76}$
Supervised results with target EAE annotations.		
Super.	$58.09_{\pm 1.51}$	$65.11_{\pm 0.94}$

Table 4: Results (F1%) of ACE05 Arabic and Chinese. "†" denotes reported results from Huang et al. (2022).

predicting with a FrameNet classifier based on a multilingual pre-trained encoder and adopting the mostly predicted label for each role. Due to differences in word order and usage of prepositional words in non-English languages, we exclude preposition words and simply order the roles by their ARG numbers.[21] We also include QA datasets for the target languages, adopting CMRC-2018 (Cui et al., 2019) for Chinese and the Arabic portion of TyDiQA (Clark et al., 2020) for Arabic. All our models in this experiment are based on the pre-trained XLM-R_{base} (Conneau et al., 2020).

The results are shown in Table 4. In the first group, we compare zero-shot performance without any EAE training resources. Similar to the previous trends, SRL models are better than QA models, while including annotations in the target language could provide further benefits. In the second group, we assume access to English EAE training data. Similar to §3.2.2, we adopt an intermediate-training scheme by further fine-tuning the QA or SRL model on the English EAE data. Compared with the results of directly training in English, intermediate training with external resources could bring improvements. Again we see that models enhanced with SRL resources obtain the overall best results, which are quite promising when compared with the supervised ones.

[18] We also tried a masked-language-model objective but did not find obvious improvements.

[19] https://github.com/PlusLabNLP/X-Gear

[20] We further re-tokenize Chinese data with CoreNLP (Manning et al., 2014) to align with segmentation in OntoNotes.

[21] The Arabic and Chinese frames adopt similar schemes as in English, specifying roles of {ARG0, ARG1, ...}. We find it reasonable by simply ordering them by the role numbers and forming templates of "ARG0 V ARG1 ARG2 ...".

Model	Overall	Same-Sent.	Cross-Sent.
QA	$28.23_{\pm 0.74}$	$35.16_{\pm 1.42}$	$11.66_{\pm 0.69}$
SRL	$48.03_{\pm 0.30}$	$53.36_{\pm 0.30}$	$2.81_{\pm 0.78}$
SRL+pseudo	$48.00_{\pm 0.14}$	$53.50_{\pm 0.16}$	$11.17_{\pm 1.88}$
Super.	$57.38_{\pm 0.84}$	$63.45_{\pm 0.86}$	$25.52_{\pm 1.31}$

Table 5: Argument head F1(%) on RAMS test set.

3.3.3 Multi-sentence

Finally, we investigate multi-sentence event arguments, which are not constrained to the same sentence of the event trigger but can come from the document-level contexts. To investigate this phenomenon, we evaluate[22] on the RAMS dataset (Ebner et al., 2020), which annotates event arguments within five-sentence windows around the triggers. We similarly extend contexts to five-sentence windows if available in our training of QA and SRL models for this experiment.

The zero-shot results are shown in the first group of Table 5. Consistent with our previous findings, SRL performs better than QA for same-sentence arguments. Nevertheless, it predicts very few cross-sentence arguments. This is not surprising because there are no such signals in the SRL training data. Inspired by previous works on coreference and anaphora resolution (Varkel and Globerson, 2020; Konno et al., 2021), we create pseudo SRL data with cross-sentence arguments by surface-string matching. Specifically, for each nominal argument in an SRL instance, we search for words in nearby sentences that have the same lemma as the argument's head word. If there are, we delete the original true argument and add pseudo cross-sentence argument links to those matched words. Although deletion may create ungrammatical instances, we find it better than other schemes; such as replacing the original argument with a "[MASK]" token. With the additional synthetic data, the model can recover certain cross-sentence arguments while keeping similar same-sentence performance. Multi-sentence argument extraction is still a difficult task, where even the supervised system can only obtain an F1 score of around 25%. This calls for further exploration, and an investigation of how best to use auxiliary data (such as from SRL) may be a promising direction.

[22]Since our head-expanding heuristic does not cover the argument span annotation conventions of RAMS, for simplicity we only evaluate argument head words.

4 Related Work

Utilizing shallow semantics for event-centric information extraction tasks has been explored previously. Liu et al. (2016) leverage FrameNet frames to enhance event detection. Wang et al. (2021) conduct contrastive pre-training with AMR structures to enhance event extraction. Several works utilize predicted shallow semantic structures as inputs to help low-resource event extraction (Peng et al., 2016; Huang et al., 2018; Lyu et al., 2021) and event schema induction (Huang et al., 2016). Moreover, SRL has been utilized for implicit argument linking or implicit semantic role labeling (iSRL) in many previous works (Chen et al., 2010; Laparra and Rigau, 2012, 2013; Feizabadi and Padó, 2015; O'Gorman, 2019). This work follows these directions and shows that SRL can be a valuable direct training resource for EAE.

For the EAE task, most previous works adopt a classification-based strategy where each role is assigned static querying parameters (Chen et al., 2015; Nguyen et al., 2016; Wang et al., 2019b; Pouran Ben Veyseh et al., 2020; Ma et al., 2020; Ebner et al., 2020). Recently, two interesting alternative strategies have been explored to enable extraction in more flexible ways: MRC-based methods cast the problem as answering role questions (Liu et al., 2020; Du and Cardie, 2020; Li et al., 2020; Feng et al., 2020; Lyu et al., 2021; Liu et al., 2021a), while generation-based methods adopt sequence-to-sequence generation schemes (Paolini et al., 2021; Li et al., 2021b; Hsu et al., 2022; Lu et al., 2021; Du et al., 2021; Huang et al., 2022). We cover all these strategies within a unified role querying framework and further explore a template-based role querying strategy. This strategy is also related with prompt-based learning (Liu et al., 2021b; Schick and Schütze, 2021; Li and Liang, 2021; Petroni et al., 2019), but differs in the extraction-targeted paradigm. Concurrently, Ma et al. (2022) adopt a similar idea, while this work differs mainly in our focus on transfer learning.

5 Conclusion

In this work, we explore transfer learning from semantic roles to event arguments. With unified role querying strategies, we show that SRL annotations are a valuable resource for event argument extraction. The SRL model also obtains promising results when extended to new scenarios with domain and language differences.

Limitations

This work has several limitations. Firstly, we only focus on the event argument extraction step and assume given event triggers. Though the first step of event detection is also important for event extraction, we do not cover it in this work mainly due to two reasons: 1) the annotation of event triggers is generally less laborious than argument annotation since word-level tagging instead of pairwise linking is required; 2) Event detection is highly specific to the target scheme, which is different than argument extraction where there are more sharings between semantic roles and event arguments. Secondly, in this work, SRL templates are created heuristically and do not cover syntactic and language variations. For example, we only construct English-styled templates in active voice, which might not be ideal for all cases. We mainly aim to show that the template-based method is a promising way to perform argument extraction, especially in transferring scenarios, but surely there could be better ways to construct the querying templates. Finally, though the application of argument augmentation recovers certain amounts of distance arguments, it is still far from an ideal solution to the problem. This calls for more future investigations in this direction, researching toward deeper and more comprehensive semantic understanding of natural languages.

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.

Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. The Berkeley FrameNet project. In *36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 1*, pages 86–90, Montreal, Quebec, Canada. 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, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Desai Chen, Nathan Schneider, Dipanjan Das, and Noah A. Smith. 2010. SEMAFOR: Frame argument resolution with log-linear models. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 264–267, Uppsala, Sweden. Association for Computational Linguistics.

Yubo Chen, Liheng Xu, Kang Liu, Daojian Zeng, and Jun Zhao. 2015. Event extraction via dynamic multi-pooling 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 1: Long Papers)*, pages 167–176, Beijing, China. Association for Computational Linguistics.

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, Online. Association for Computational Linguistics.

Jonathan H. Clark, Eunsol Choi, Michael Collins, Dan Garrette, Tom Kwiatkowski, Vitaly Nikolaev, and Jennimaria Palomaki. 2020. TyDi QA: A benchmark for information-seeking question answering in typologically diverse languages. *Transactions of the Association for Computational Linguistics*, 8:454–470.

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

Yiming Cui, Ting Liu, Wanxiang Che, Li Xiao, Zhipeng Chen, Wentao Ma, Shijin Wang, and Guoping Hu. 2019. A span-extraction dataset for Chinese machine reading comprehension. 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 5883–5889, Hong Kong, China. Association for Computational Linguistics.

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

Xinya Du, Alexander Rush, and Claire Cardie. 2021. GRIT: Generative role-filler transformers for document-level event entity extraction. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 634–644, Online. Association for Computational Linguistics.

Seth Ebner, Patrick Xia, Ryan Culkin, Kyle Rawlins, and Benjamin Van Durme. 2020. Multi-sentence argument linking. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8057–8077, Online. Association for Computational Linguistics.

Parvin Sadat Feizabadi and Sebastian Padó. 2015. Combining seemingly incompatible corpora for implicit semantic role labeling. In *Proceedings of the Fourth Joint Conference on Lexical and Computational Semantics*, pages 40–50, Denver, Colorado. Association for Computational Linguistics.

Rui Feng, Jie Yuan, and Chao Zhang. 2020. Probing and fine-tuning reading comprehension models for few-shot event extraction. *arXiv preprint arXiv:2010.11325*.

Steven Y. Feng, Varun Gangal, Jason Wei, Sarath Chandar, Soroush Vosoughi, Teruko Mitamura, and Eduard Hovy. 2021. A survey of data augmentation approaches for NLP. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 968–988, Online. Association for Computational Linguistics.

Nicholas FitzGerald, Julian Michael, Luheng He, and Luke Zettlemoyer. 2018. Large-scale QA-SRL parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2051–2060, Melbourne, Australia. Association for Computational Linguistics.

Daniel Gildea and Daniel Jurafsky. 2002. Automatic labeling of semantic roles. *Computational Linguistics*, 28(3):245–288.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. 2017. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1321–1330. PMLR.

Geoffrey Hinton, Oriol Vinyals, Jeff Dean, et al. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2(7).

Eduard Hovy, Mitchell Marcus, Martha Palmer, Lance Ramshaw, and Ralph Weischedel. 2006. OntoNotes: The 90% solution. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, pages 57–60, New York City, 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.

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

Lifu Huang, Taylor Cassidy, Xiaocheng Feng, Heng Ji, Clare R. Voss, Jiawei Han, and Avirup Sil. 2016. Liberal event extraction and event schema induction. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 258–268, Berlin, Germany. Association for Computational Linguistics.

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.

Jin-Dong Kim, Tomoko Ohta, and Jun'ichi Tsujii. 2008. Corpus annotation for mining biomedical events from literature. *BMC bioinformatics*, 9(1):1–25.

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

Ayal Klein, Jonathan Mamou, Valentina Pyatkin, Daniela Stepanov, Hangfeng He, Dan Roth, Luke Zettlemoyer, and Ido Dagan. 2020. QANom: Question-answer driven SRL for nominalizations. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3069–3083, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Ryuto Konno, Shun Kiyono, Yuichiroh Matsubayashi, Hiroki Ouchi, and Kentaro Inui. 2021. Pseudo zero pronoun resolution improves zero anaphora resolution. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3790–3806, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Egoitz Laparra and German Rigau. 2012. Exploiting explicit annotations and semantic types for implicit argument resolution. In *2012 IEEE Sixth International Conference on Semantic Computing*, pages 75–78. IEEE.

Egoitz Laparra and German Rigau. 2013. ImpAr: A deterministic algorithm for implicit semantic role labelling. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1180–1189, Sofia, Bulgaria. Association for Computational Linguistics.

LDC. 2005. ACE (automatic content extraction) english annotation guidelines for events version 5.4.3. *Linguistic Data Consortium*.

LDC. 2015. Deft Rich ERE annotation guidelines: Events version 3.0. *Linguistic Data Consortium*.

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.

Fayuan Li, Weihua Peng, Yuguang Chen, Quan Wang, Lu Pan, Yajuan Lyu, and Yong Zhu. 2020. Event extraction as multi-turn question answering. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 829–838, Online. Association for Computational Linguistics.

Manling Li, Sha Li, Zhenhailong Wang, Lifu Huang, Kyunghyun Cho, Heng Ji, Jiawei Han, and Clare Voss. 2021a. The future is not one-dimensional: Complex event schema induction by graph modeling for event prediction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5203–5215, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Sha Li, Heng Ji, and Jiawei Han. 2021b. Document-level event argument extraction by conditional generation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 894–908, Online. Association for Computational Linguistics.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, Online. Association for Computational Linguistics.

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

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

Jian Liu, Yufeng Chen, and Jinan Xu. 2021a. Machine reading comprehension as data augmentation: A case study on implicit event argument extraction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2716–2725, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021b. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586*.

Shulin Liu, Yubo Chen, Shizhu He, Kang Liu, and Jun Zhao. 2016. Leveraging FrameNet to improve automatic event detection. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2134–2143, Berlin, Germany. 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*.

Yaojie Lu, Hongyu Lin, Jin Xu, Xianpei Han, Jialong Tang, Annan Li, Le Sun, Meng Liao, and Shaoyi Chen. 2021. Text2Event: Controllable sequence-to-structure generation for end-to-end event extraction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2795–2806, Online. Association for Computational Linguistics.

Qing Lyu, Hongming Zhang, Elior Sulem, and Dan Roth. 2021. Zero-shot event extraction via transfer learning: Challenges and insights. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 322–332, Online. Association for Computational Linguistics.

Jie Ma, Shuai Wang, Rishita Anubhai, Miguel Ballesteros, and Yaser Al-Onaizan. 2020. Resource-enhanced neural model for event argument extraction. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3554–3559, Online. Association for Computational Linguistics.

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

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky.

2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60, Baltimore, Maryland. Association for Computational Linguistics.

Adam Meyers, Ruth Reeves, Catherine Macleod, Rachel Szekely, Veronika Zielinska, Brian Young, and Ralph Grishman. 2004. The NomBank project: An interim report. In *Proceedings of the Workshop Frontiers in Corpus Annotation at HLT-NAACL 2004*, pages 24–31, Boston, Massachusetts, USA. Association for Computational Linguistics.

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.

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.

Timothy J O'Gorman. 2019. *Bringing together computational and linguistic models of implicit role interpretation*. Ph.D. thesis, University of Colorado at Boulder.

Hiroki Ouchi, Hiroyuki Shindo, and Yuji Matsumoto. 2018. A span selection model for semantic role labeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1630–1642, Brussels, Belgium. Association for Computational Linguistics.

Martha Palmer. 2009. Semlink: Linking propbank, verbnet and framenet. In *Proceedings of the generative lexicon conference*, pages 9–15. GenLex-09, Pisa, Italy.

Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The Proposition Bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–106.

Martha Palmer, Daniel Gildea, and Nianwen Xue. 2010. Semantic role labeling. *Synthesis Lectures on Human Language Technologies*, 3(1):1–103.

Sinno Jialin Pan and Qiang Yang. 2009. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359.

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

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.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? 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 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Jason Phang, Thibault Févry, and Samuel R Bowman. 2018. Sentence encoders on stilts: Supplementary training on intermediate labeled-data tasks. *arXiv preprint arXiv:1811.01088*.

Amir Pouran Ben Veyseh, Tuan Ngo Nguyen, and Thien Huu Nguyen. 2020. Graph transformer networks with syntactic and semantic structures for event argument extraction. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3651–3661, Online. Association for Computational Linguistics.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 101–108, Online. Association for Computational Linguistics.

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, Melbourne, Australia. Association for Computational Linguistics.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

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

Sebastian Ruder, Matthew E. Peters, Swabha Swayamdipta, and Thomas Wolf. 2019. Transfer learning in natural language processing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Tutorials*, pages 15–18, Minneapolis, Minnesota. Association for Computational Linguistics.

Timo Schick and Hinrich Schütze. 2021. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 255–269, Online. Association for Computational Linguistics.

Kevin Stowe, Jenette Preciado, Kathryn Conger, Susan Windisch Brown, Ghazaleh Kazeminejad, James Gung, and Martha Palmer. 2021. SemLink 2.0: Chasing lexical resources. In *Proceedings of the 14th International Conference on Computational Semantics (IWCS)*, pages 222–227, Groningen, The Netherlands (online). Association for Computational Linguistics.

Yuval Varkel and Amir Globerson. 2020. Pre-training mention representations in coreference models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8534–8540, Online. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. *Advances in neural information processing systems*, 28.

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

Alex Wang, Jan Hula, Patrick Xia, Raghavendra Pappagari, R. Thomas McCoy, Roma Patel, Najoung Kim, Ian Tenney, Yinghui Huang, Katherin Yu, Shuning Jin, Berlin Chen, Benjamin Van Durme, Edouard Grave, Ellie Pavlick, and Samuel R. Bowman. 2019a. Can you tell me how to get past sesame street? sentence-level pretraining beyond language modeling. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4465–4476, Florence, Italy. Association for Computational Linguistics.

Xiaozhi Wang, Ziqi Wang, Xu Han, Zhiyuan Liu, Juanzi Li, Peng Li, Maosong Sun, Jie Zhou, and Xiang Ren. 2019b. 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)*, pages 5777–5783, Hong Kong, China. Association for Computational Linguistics.

Ziqi Wang, Xiaozhi Wang, Xu Han, Yankai Lin, Lei Hou, Zhiyuan Liu, Peng Li, Juanzi Li, and Jie Zhou. 2021. CLEVE: Contrastive Pre-training for Event Extraction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6283–6297, Online. Association for Computational Linguistics.

Ralph Weischedel, Martha Palmer, Mitchell Marcus, Eduard Hovy, Sameer Pradhan, Lance Ramshaw, Nianwen Xue, Ann Taylor, Jeff Kaufman, Michelle Franchini, et al. 2013. Ontonotes release 5.0. *Linguistic Data Consortium, Philadelphia, PA*, 23.

A Details of Methods

A.1 Modeling Details

This sub-section provides more details of the models and the querying strategies that are briefly described in §2.1.

We first introduce some common modeling settings before diving into specific querying strategies. As described in the main context, we adopt a unified role querying model for argument extraction with representations of the role queries $\mathbf{q}_r$ and the candidate words $\mathbf{h}_w$. The construction of word representations follows common practice: we feed the input sequence to a contextualized encoder and utilize each word's output hidden vector. When a word is split into multiple sub-words, the first sub-word is taken. To encode the trigger word, the input embedding of the trigger is added with a specific trigger embedding, which is randomly initialized and tuned together with the model. There are cases when the context does not have mentions for some roles (no arguments), where we adopt an all-zero dummy $\mathbf{h}_\epsilon$, which essentially fixes the no-argument scores to zero.

During training, we use the standard cross-entropy loss function. When there are more than one gold argument, we simply apply equal weights to them. In testing, for each role, we select the words whose score is larger than zero and ranks within the top two among all candidate words. One important aspect that we do not explicitly consider in the output modeling is the interactions between arguments as well as frame-level global features, which have been shown effective for event extraction (Lin et al., 2020) and event schema induction (Li et al., 2021a). Incorporating these for the transfer scenarios would be an interesting future direction. The selected words are further expanded to argument spans using the dependency-tree-based heuristic as described in the main content.

The main difference among the querying strategies is in the construction of the querying vectors, which is described in the following.

1) CLF

For the traditional classification-based strategy, we allocate a specific vector for each role, which is included as model parameters. In the case where there is enough supervision, these vectors can be randomly initialized. To fit our goal of transfer learning, we take advantage of the natural language role names and encode them individually using a vanilla pre-trained model, with an input format of:

$$[CLS]\ \ role\_name\ \ [SEP]$$

For example, for the role of "artifact", the input is simply "$[CLS]\ \ artifact\ [SEP]$". We take the averagely-pooled output representations. These vectors are frozen for our main transferring experiments since we find this to be slightly better. Notice that since each role name is encoded by itself without any other contexts, the representations are non-contextualized, making this strategy almost the same as using a classifier.

Role	Question Template
Person	Who is the $[]_{\text{role\_name}}$ in the $[]_{\text{trigger\_text}}$ event?
Place	Where does the $[]_{\text{trigger\_text}}$ event take place?
Others	What is the $[]_{\text{role\_name}}$ of the $[]_{\text{trigger\_text}}$ event?

Table 6: Question templates from Liu et al. (2021a).

2) MRC

For the MRC-based strategy, we form a question for each role and dynamically obtain the query vectors by encoding the question together with the context. We adopt the question templates from Liu et al. (2021a), as shown in Table 6. For example, to query the "artifact" role of the "bought" event in Figure 1, we ask: "What is the artifact of the bought event?". One advantage of this strategy is that we only need role types, role names, and trigger texts to form a question, making it less difficult to extend to the SRL cases. In our preliminary experiments, we also tried role-specific questions for ACE utilizing those from Lyu et al. (2021), such as "What is bought?", and found similar results. Following standard MRC models, we concatenate the question and the context as the input sequence, which is fed to the encoder:

$$[CLS]\ \ role\_question\ \ [SEP]\ \ context\ \ [SEP]$$

Furthermore, instead of introducing extra parameters with an extra answer selection head, we simply take the contextualized representations of the question word[23] as the querying vector.

To apply MRC to the SRL data, we further need to know whether a role is person-related, to decide

[23] We choose the question word instead of the role name to allow easier transfer from QA datasets where there may be no specific querying roles. In preliminary experiments, we also tried average pooling over the question tokens to form querying vectors but did not find better results.

using the questions of "What" or "Who".[24] Similar to our template-construction strategy, we again take a counting-based method by checking how many times a role is filled by a personal pronoun. If this happens for a role with a frequency larger than 10%, we regard it as potentially person-related. Since there are cases where a role can be filled by either a person or an object, at training time we randomly pick "Who" or "What" questions for these potentially person-related roles, while only asking "Who" in testing.

3) GEN

The template-generation-based strategy requires a template for each event or SRL frame, which specifies a canonical realization of this frame in a natural language sentence. For example, for the "TransferOwnership" event, we have a template of "seller give artifact to buyer for beneficiary in place", where each role occupies a placeholder slot. In this strategy, a sequence-to-sequence encoder-decoder model is utilized. The context is encoded by the encoder while the filled template is generated by the decoder. We mostly follow Li et al. (2021b) but make some modifications to the output modeling. Instead of directly replacing the slots with actual argument words, we keep the role names and insert the actual arguments after the role slots. For example, we output[25] "seller [UNK] give artifact book to buyer he for beneficiary [UNK] in place store" instead of "[UNK] give book to he for [UNK] in store". We keep the role names for two reasons: firstly, the role names in the target sequence can act as a guide of the to-be-filled arguments; moreover, since the arguments are restricted to be words from the context, we can utilize the representations of the role names as queries to point to the context words. The second point allows us to form a pointer-network-styled model, which directly selects arguments from the context word representations, fitting in our unified role-querying framework.

4) TSQ

We do not need to fill in the template with actual argument words, since our target task is an extraction task where we only need to find the argument mentions in the context. Moreover, if no generation is required, we could merge the context and the template to allow bidirectional modeling. Motivated by this, we keep the unfilled but already natural-language-styled template as it is, concatenate it with the context and feed the full sequence to the encoder:

$$[CLS]\ template\ [SEP]\ context\ [SEP]$$

After the encoding, we take the output representations (first sub-token) of each role slot as its query vector and apply all the role queries parallelly to select the corresponding argument words. This can be viewed as a combination of MRC and GEN, taking advantages of both methods. As in MRC, we perform the extraction for the role queries, and no generation is needed, and as in GEN, the template allows us to embed all the role queries in one sequence rather than forwarding multiple times for different roles. Moreover, since all the role queries are performed parallelly without inter-dependencies, this can be viewed as a non-autoregressive method which is more efficient than GEN.

B Details of Main Experiments

B.1 Settings

The main experiments are conducted with English ACE[26] (ACE05-E$^+$) and ERE[27] (ERE-EN) datasets. The statistics of the event data are shown in Table 7. For SRL data, we take those from the latest PropBank[28] (EWT and OntoNotes), NomBank[29] and FrameNet[30]. For FrameNet, we utilize the lexicographic annotation sets since there are much more instances. We ignore SRL frames that do not have verbal predicates and only keep related non-core roles. We further split SRL arguments in coordination to align with EAE, which treats coordinated entities as separate arguments. For QA data, we include SQuAD[31], QA-SRL[32], QANom[33] and QAMR[34]. For the QA instances, we follow

[24] The "Where" question is designated to the role of PLACE.

[25] We utilize a specific [UNK] symbol to denote the case when there are no arguments in the context.

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

[27] LDC2015E29, LDC2015E68, and LDC2015E78.

[28] https://github.com/propbank/propbank-release

[29] https://nlp.cs.nyu.edu/meyers/NomBank.html

[30] https://framenet.icsi.berkeley.edu/fndrupal/

[31] https://rajpurkar.github.io/SQuAD-explorer/

[32] https://github.com/uwnlp/qasrl-bank

[33] https://github.com/kleinay/QANom

[34] https://github.com/uwnlp/qamr

Dataset	Split	Sent.	Event	Arg.	A/E
ACE	Train	192.2K	4.4K	6.6K	1.5
	Dev	0.9K	0.5K	0.8K	1.6
	Test	0.7K	0.4K	0.7K	1.6
ERE	Train	147.3K	6.2K	8.9K	1.4
	Dev	1.2K	0.5K	0.7K	1.4
	Test	1.2K	0.6K	0.8K	1.5

Table 7: Statistics of the ACE and ERE data. "A/E" denotes the averaged argument number per event.

Type	Dataset	Sent.	Inst.	Arg.	A/I
SRL	PropBank	77.7K	256.1K	374.5K	1.5
	NomBank	28.2K	56.9K	86.8K	1.5
	FrameNet	173.0K	173.4K	208.5K	1.5
QA	SQuAD	62.2K	130.3K	86.8K	0.7
	QA-SRL	64.0K	299.3K	299.3K	1.0
	QANom	7.1K	26.4K	26.4K	1.0
	QAMR	4.8K	88.3K	88.3K	1.0

Table 8: Statistics of the SRL and QA data. "Inst." denotes the number of SRL or QA instances, while "A/I" denotes the averaged number of arguments or answers per instance.

Method	ACE		ERE	
	gold	pred.	gold	pred.
Super.	68.93	53.98	72.24	49.77
QA	38.62	29.94	38.74	26.02
SRL_{CLF}	33.18	26.65	33.89	25.78
SRL_{MRC}	47.08	25.50	52.32	38.39
SRL_{GEN}	49.99	37.01	55.38	38.77
SRL_{TSQ}	53.36	39.41	57.81	40.08
+shuf.	54.82	41.15	58.54	40.78
+distill	55.17	41.50	57.95	40.01
+both	**56.35**	**42.16**	**58.96**	**41.15**

Table 9: Zero-shot EAE results (F1%) on event test sets with gold or predicted event triggers.

Michael et al. (2018) and use a question-context alignment heuristic to find a predicate in the context for each question. Since the external data is mainly utilized as training resources, we simply concatenate all the available data portions for training while splitting a small subset for development. Data statistics of SRL and QA are shown in Table 8.

We utilize pre-trained language models ($RoBERTa_{base}$ for encoder-only models (CLF, MRC, TSQ) and $BART_{base}$ for encoder-decoder models (GEN)) to initialize our models and fine-tune the full models in all the experiments. The model parameter numbers are 125M and 139M, for those with RoBERTa and BART respectively. For the hyper-parameter settings, we mostly follow common practices. Adam is utilized for optimization. The learning rate is initially set to 2e-5 and linearly decayed to 2e-6 throughout the training process. The models are trained for 50K steps with a batch size of 16 for event and SRL and 32 for QA. We pick models by the performance on the development set of each task. In low-resources cases, the original event development set is also down-sampled accordingly as the training set. All the experiments can be conducted with one 1080 Ti GPU and the training can usually be finished within several hours.

B.2 Details of GPT-3 Prompting

To perform prompting with GPT-3, we utilize the OpenAI API.[35] We adopt the "Davinci" model and the Completion endpoint. We design the prompts with a strategy that is similar to MRC. The prompts consist of three parts: the context sentence, a question for the querying role and a partial answer to be completed. The context is simply the sentence where the event trigger appears, while the questions are those shown in Table 6 as in the MRC strategy. The to-be-completed answer sentence is a declarative repetition of the question.

[35] https://openai.com/api/

For example, we have the following prompt to query the "artifact" role with the context of "He went to the store and bought a book.":

```
He went to the store and bought a
book.
Q: What is the artifact of the
bought event?
A: The artifact of the bought
event is
```

We let the GPT-3 model greedily decode the remaining answer sentence and match the results to the tokens in the original context to obtain the arguments. When there are no matchings or the answer is "not specified", no arguments are predicted for the querying role. Since the answer should come from the context, we utilize the "logit_bias" parameter to constrain the model to adopt sub-tokens that appear in the context (or those from "not specified").

B.3 Results with Predicted Triggers

In our main experiments, we assume given gold event triggers. In this sub-section, we train a supervised sequence-labeling event detector and further utilize the predicted triggers to perform zero-shot

Model	ACE	ERE
encoder-only	56.35	58.96
encoder-decoder	55.36	58.44

Table 10: Comparisons between encoder-only and encoder-decoder TSQ models.

Method	Gold	Predicted
OneIE (Lin et al., 2020)	-	54.8
EEQA (Du and Cardie, 2020)	63.34	-
GenIE (Li et al., 2021b)	66.67	53.71
CLF	66.96	52.62
MRC	66.55	52.43
GEN	66.76	52.81
TSQ	68.93	53.98

Table 11: Comparisons of fully-supervised ACE05-E$^+$ test results (F1%) (with gold or predicted triggers).

argument extraction. The results are shown in Table 9. The event detectors could obtain labeled F1 scores of 71.0 and 58.4 for ACE and ERE, respectively. With the predicted triggers, the results drop correspondingly against those with gold triggers. Nevertheless, the overall trends are similar. The TSQ strategy performs the best while the argument augmentation is also helpful with predicted triggers. One interesting direction to explore is full event extraction in the zero-shot and low-resource scenarios, which we leave to future work.

B.4 Model Choice for TSQ

Concurrently, Ma et al. (2022) explore an idea that is similar to TSQ, while taking sequence-to-sequence encoder-decoder model to perform argument extraction. Specifically, they encode the contexts with the encoder while putting the template on the decoder side. We also compare this encoder-decoder scheme with our encoder-only TSQ in the transfer scenario. The results are shown in Table 10, where the encoder-only model is slightly better. Therefore, we utilize the encoder-only model for the TSQ strategy.

B.5 Supervised Results

Although our main focus is on the transfer scenarios, we also conduct purely supervised experiments on the target EAE datasets. We first compare fully-supervised results with previous works. As shown in Table 11, our results are generally comparable to those in previous works, which validates the quality of our implementation.

Furthermore, we compare the four querying

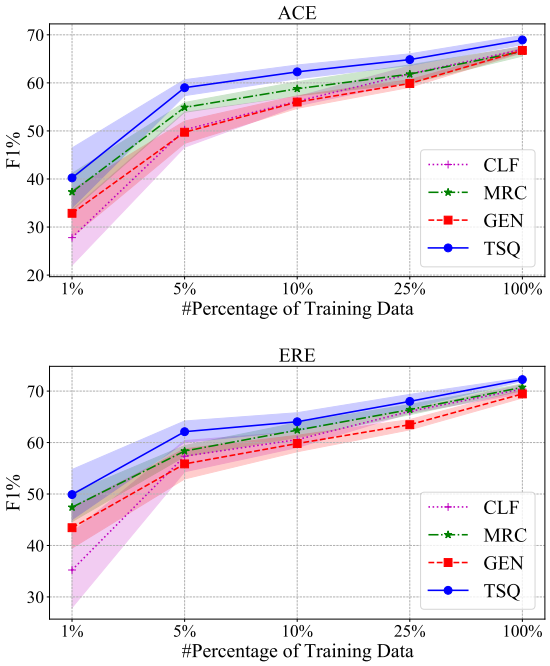

Figure 5: Argument F1(%) scores on ACE and ERE test sets with different amounts of training data. Here x-axis (drawn in log scale) denotes the percentage of original training data sampled. The shaded areas indicate the ranges of standard deviations.

strategies with different amounts of training data. The results are shown in Figure 5. The overall trend is similar in both datasets. In high-resource scenarios, different querying strategies could obtain similar results. In low-resource cases, the methods that capture more contextual information in the queries can generally perform better. The CLF strategy with non-contextualized queries obtains worse results than the others, while TSQ is the overall best-performing strategy. This is also consistent with the results in the zero-shot transfer scenarios.

B.6 Manual Analysis

We further perform a manual error analysis to investigate what the main error types are. We randomly take 100 event frames that contain prediction errors from the ACE development set and categorize the errors. We perform this analysis for both our best zero-shot SRL model and the supervised model to examine where the main gaps are. We specify eight error categories:

- **Ambiguous** cases, where there are annotation errors or ambiguities, and the predictions could be regarded as correct in some way.
- **Coreference**, where predicted and gold arguments are co-referenced in some way.

Category	Example	SRL	Super.
Correct	—	194 (50.52%)	238 (63.47%)
Role	Actually, they **paid**$_{\text{TransferMoney}}$ for [it]$_{\text{Beneficiary}}$.	45 (11.72%)	22 (5.87%)
Local	My$_{\text{Buyer}}$ plan is to **pay**$_{\text{TransferOwnership}}$ off my car.	40 (10.42%)	23 (6.13%)
Head	They **fired**$_{\text{Attack}}$ mortars in the [direction]$_{\text{Target}}$ of the 7th Cavalry$_{\text{Target}}$.	24 (6.25%)	8 (2.13%)
Global	"We condemned the **attack**$_{\text{Attack}}$," he said, adding that his messages to the terrorists$_{\text{Attacker}}$ is: Their efforts will not be successful.	24 (6.25%)	17 (4.53%)
Others	—	14 (3.65%)	10 (2.67%)
Ambiguous	At least four [policeman]$_{\text{Attacker Victim}}$ were injured in **clashes**$_{\text{Attack}}$.	18 (4.69%)	20 (5.33%)
Span	The 1st [Brigade]$_{\text{Attacker Attacker}}$ took Karbala with a minimal **fight**$_{\text{Attack}}$.	12 (3.12%)	14 (3.73%)
Coreference	He$_{\text{Defendant}}$ skipped bail during [his]$_{\text{Defendant}}$ **trial**$_{\text{Hearing}}$.	13 (3.39%)	23 (6.13%)

Table 12: Examples of the categories and results of the manual error analysis. In the examples, the triggers are shown in **bold** texts with brown event types. The gold arguments are presented in underlined spans with red roles, while predicted ones are [bracketed] followed by blue roles. Results are denoted with number counts and (percentages). The rows of the error categories are sorted by the gap between SRL and supervised counts.

- **Span** mismatch, where the main contents are captured with non-crucial boundary mismatches.
- **Head** mismatch, where the main contents are roughly captured but not with the exact annotated words. This happens mostly in appositions or noun modifiers with more specific content.
- **Role** misunderstanding, where the semantic meaning of a role is not correctly understood.
- **Local** inference, where correct predictions require semantic inference at the local clause.
- **Global** understanding, where correct predictions require global understanding of the full context.
- **Others**, where the error does not fall into any of the above categories.

Examples of these categories and the results are shown in Table 12. According to the statistics, the main gaps between the SRL and supervised models are in the categories of role misunderstanding, lacking of semantic inference as well as head mismatches. Head mismatches are due to the discrepancies between syntactic head and semantic core words, and might not cause severe problems. The first two are more semantic errors that are related to the essence of the EAE task. Role misunderstanding may be related to template mismatches, where roles in the SRL templates are different than those in target event ones. Lacking of semantic inference is mostly upon distant arguments. Though the argument augmentation techniques recover certain distant arguments for SRL frames, this problem is still far from being solved. Notice that these semantic errors reveal the main difficulties of the EAE task, which even supervised systems have not yet fully tackled. To solve these problems, more comprehensive semantic understanding is required.

Role	QA	SRL	+shuf.	+distill	+both
Place	51.73	47.10	46.94	56.46	**57.38**
Attacker	34.59	56.52	**59.03**	57.88	58.19
Entity	38.02	40.96	**43.97**	41.59	43.36
Target	33.15	38.62	38.66	37.59	**39.04**
Victim	66.98	80.58	**81.18**	79.27	79.49
Artifact	13.95	49.52	**62.82**	46.47	60.54
Person	58.53	71.82	72.16	73.46	**73.20**
Recipient	45.75	46.68	47.51	**50.64**	49.37
Destination	66.11	65.97	66.14	**68.02**	66.00
Instrument	34.28	40.00	47.05	43.54	**49.56**

Table 13: F1% score breakdowns by argument roles.

B.7 Role Breakdowns

We perform breakdowns on argument roles on ACE with the zero-shot models. Table 13 shows the results of the top-ten frequent roles. Interestingly, distillation generally helps more on the non-core roles, such as PLACE and DESTINATION, while shuffling enhances core roles, like ATTACKER and VICTIM. Finally, applying both could lead to the overall best results.

B.8 Speed Comparisons

We also perform decoding speed comparisons to examine the efficiency of different querying strategies. The results are shown in Table 15. There are no surprises that the simplest CLF strategy achieves the highest decoding speed since its input sequences are the shortest and there is no further complex query encoding. TSQ is only around 10% slower,

Event	Template	Role questions
Expression	agent express theme	What is expressed?
Transcription	agent transcribe theme	What is transcribed?
Catabolism	agent degrade theme	What is degraded?
Phosphorylation	agent phosphorylate theme	What is phosphorylated?
Localization	agent localize theme	What is localized?
Binding	agent bind theme1 to theme2	What is bound? What is something bound to?
Regulation	cause regulate theme	What causes the regulation? What is regulated?

Table 14: Manually specified templates and role questions for GENIA events ("agent" is a dummy role introduced to form the templates in active voice).

Method	Single-instance	Batched
CLF	**184**	**316**
MRC	106	146
GEN	28	144
TSQ	167	281

Table 15: Decoding speed (instances per second) comparisons of different role querying strategies. We evaluate both single-instance and batched decoding modes.

Language	Model	Pearson	Spearman
Arabic	w/o SRL	0.6050	0.6727
	w/ SRL	0.5157	0.1394
Chinese	w/o SRL	0.6910	0.5636
	w/ SRL	0.5025	0.2727

Table 16: Correlations between relative role order differences and performance gaps to supervised systems for multi-lingual EAE (with top-10 frequent roles).

but still efficient compared with the other two methods, where MRC suffers from multiple forwarding for different role queries and GEN requires autoregressive decoding at testing time.

C Further Extensions

C.1 GENIA Details

For the GENIA experiments, one more assumed input is the protein entities, following the settings of BioNLP shared task. Since our model-predicted argument head words might not match the protein entities, we perform a syntax-based post-processing heuristic. For a predicted argument word, we check its descendants in the dependency tree and relocate the argument to the highest node that belongs to an entity (or an event for the "Theme" of "Regulation"). If no such items can be found, the prediction is ignored. The evaluation metric is approximate recursive matching using the official online service.[36]

For the GENIA events, we manually specify templates, which are shown in Table 14. We also manually specify role questions since the templated questions mostly fail in this scenario. For the three regulation events, we simply adopt the same specifications since no obvious differences are found when adding modifiers of "positively" or "negatively". Since our SRL templates are all formed in active voice, we introduce a dummy "agent" role to form non-passive GENIA templates. The prediction of this dummy role is ignored in testing.

[36] http://bionlp-st.dbcls.jp/GE/2011/eval-test/

C.2 Multi-lingual Analysis

One interesting aspect of the multi-lingual scenario is how the predictions are influenced by the word order difference between the source and target languages. We analyze the influence by measuring the performance differences in different roles. We first calculate the directional statistics for each role in each language, specifically: for a role in a language, what percentage of its arguments appear after the trigger? For example, "Attacker" appears after the trigger 26.9% of the time in English, while this percentage is 72.7% in Arabic. Then for each role, we have a source-target order difference metric, which is the absolute value of the frequency difference. We further calculate the performance differences between a transfer model trained with English data and a supervised model directly trained on the target language. Finally, we measure the correlation between the order differences and performance differences for the top-ten frequent roles in each language. The results for the transfer model with or without (multi-lingual) SRL intermediate training are shown in Table 16. Interestingly, if directly transferring from English to the target languages,

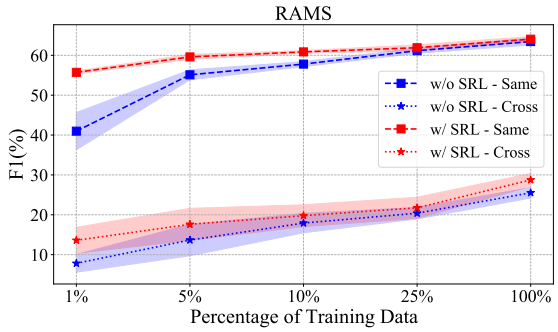

Figure 6: RAMS results (same- or cross-sentence argument F1%) with or without SRL intermediate-training.

there are at least moderate correlations between the order differences and performance gaps. While using SRL, the correlations decrease probably because of the extra signals about the target language order in the SRL data. This shows that order differences may be a major factor influencing the effectiveness of the cross-lingual transfer. Currently, our templates are all English-styled and it would be an interesting future direction to explore the influences of template specifications such as role orders.

C.3 More Multi-sentence Results

We also perform intermediate training on the RAMS dataset with different amounts of target training instances. The test results are shown in Figure 6. The same-sentence patterns are similar to those in previous ACE experiments, while SRL seems to be able to provide small but consistent benefits for cross-sentence arguments.