TAGPRIME: A Unified Framework for Relational Structure Extraction

I-Hung Hsu[*†] **Kuan-Hao Huang**[*‡] **Shuning Zhang**[◇] **Wenxin Cheng**[‡]
Premkumar Natarajan[†] **Kai-Wei Chang**[‡] **Nanyun Peng**[‡]

[†]Information Science Institute, University of Southern California
[‡]Computer Science Department, University of California, Los Angeles
[◇]Computer Science and Technology Department, Tsinghua University

{ihunghsu, pnataraj}@isi.edu,
{khhuang, kwchang, violetpeng}@cs.ucla.edu
zhang-sn19@mails.tsinghua.edu.cn, wenxin0319@ucla.edu

Abstract

Many tasks in natural language processing require the extraction of relationship information for a given condition, such as event argument extraction, relation extraction, and task-oriented semantic parsing. Recent works usually propose sophisticated models for each task independently and pay less attention to the commonality of these tasks and to have a unified framework for all the tasks. In this work, we propose to take a unified view of all these tasks and introduce TAGPRIME to address relational structure extraction problems. TAGPRIME is a sequence tagging model that appends *priming words* about the information of the given condition (such as an event trigger) to the input text. With the self-attention mechanism in pre-trained language models, the priming words make the output contextualized representations contain more information about the given condition, and hence become more suitable for extracting specific relationships for the condition. Extensive experiments and analyses on three different tasks that cover ten datasets across five different languages demonstrate the generality and effectiveness of TAGPRIME.

1 Introduction

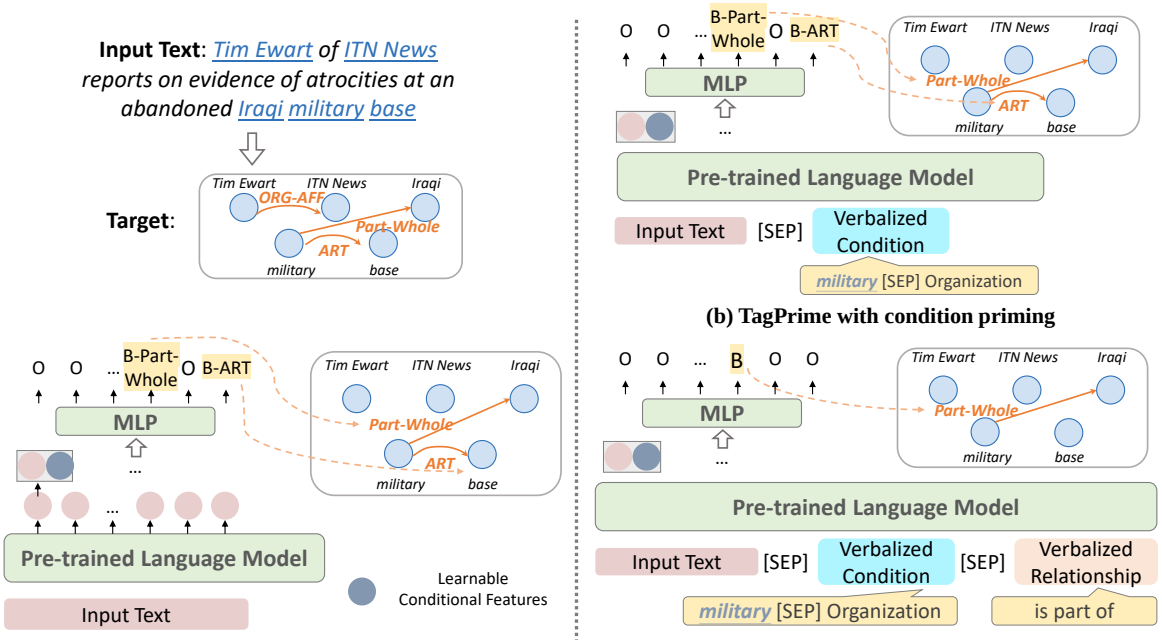
There are many tasks in natural language processing (NLP) that require extracting relational structures from texts. For example, the event argument extraction task aims to identify event arguments and *their corresponding roles* for a given event trigger (Huang et al., 2022; Wang et al., 2019). In entity relation extraction, the model identifies the tail-entities and head-entities that *forms specific relations* (Wei et al., 2020; Yu et al., 2020). In task-oriented semantic parsing, the model predicts the slots and *their semantic roles* for a given intent in an utterance (Tür et al., 2010; Li et al., 2021). These tasks are beneficial to a wide range of applications, such as dialog systems (Liu et al., 2018), question answering (Yasunaga et al., 2021), and narrative generation (Chen et al., 2019a). Prior works usually design models to specifically address each of the tasks (Sun et al., 2019; Miwa and Bansal, 2016; Han et al., 2019; Fu et al., 2019; Zhang et al., 2018). However, less attention is paid to the commonality among these tasks and having a unified framework to deal with them and provide a strong baseline for every task.

In this work, we take a unified view of these NLP tasks. We call them relational structure extraction (RSE) tasks and formulate them as a unified task that identifies arguments to a given condition and classifies their relationships. The condition could be a textual span, such as an event trigger for event argument extraction, or a concept, such as an intent for task-oriented semantic parsing.

We present TAGPRIME, a simple, unified, and strong model, which follows a sequence tagging paradigm with a *priming technique*, which is proposed by Fincke et al. (2022). TAGPRIME inherits the strength of sequence tagging models to unifiedly address RSE by converting the relational structure into a sequence of predictions by sequentially labeling tokens in the input passage. TAGPRIME further improves this framework's performance by better incorporating information about the given condition via priming. Traditional sequence tagging models usually leverage learnable feature embeddings to incorporate information about the given condition before the tags are assigned, as illustrated in Figure 1(a). With the priming mechanism, TAGPRIME augments the input text with condition-specific contexts, as illustrated in Figure 1(b) & (c). The main merit of the

[*]The authors contribute equally.

Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics
Volume 1: Long Papers, pages 12917–12932



(a) Sequence tagging model with conditional features

(b) TagPrime with condition priming

(c) TagPrime with condition & relationship priming

Figure 1: Illustrations of different models running on an instance for relation extraction, where the target is to predict the relations between the named entities. (a) **Sequence tagging model with conditional features**: A conventional sequence tagging model that embeds conditional information by adding learnable features to the output representation from a pre-trained language model. In the shown example, the conditional features contain two parts: one is the token embedding representing the conditional word *"military"*, and the other is an entity type embedding. (b) **TagPrime with condition priming**: The conditional information is further applied to the input sequence to induce the output representation from the pre-trained language model to become condition-aware. (c) **TagPrime with condition & relationship priming**: Our approach that further append the verbalized relationship to TagPrime with condition priming model. For this case, the goal of the tagging model is to make predictions specific to the relationship type in the input. We omit CRF layers after MLP layers in this figure for better readability.

priming technique comes from the nature of the self-attention mechanism in pre-trained language models. Augmenting input text with condition-specific contexts makes the sentence representations *condition-specific* directly. Thus, it unlocks the capability of sequence tagging methods for relational structure extraction better than the commonly used feature embedding approach, as shown in Section 5.

Our contributions can be summarized as follows. (1) We take a unified view of NLP tasks that requires extracting relational structures, including end-to-end event extraction, end-to-end relation extraction, and task-oriented semantic parsing. Then, we present TagPrime, a unified sequence tagging model with priming that can serve as a strong baseline to various relational structure extraction problems. (2) Thorough experiments on three different tasks show that TagPrime achieves competitive performance than the current state-of-the-art on ten datasets in five different languages. (3) We propose a novel efficient approximation to speed up TagPrime during inference time without sacrificing too much performance.

Our code will be publicly accessible at `https://github.com/PlusLabNLP/TagPrime`.

2 Related Work

Many natural language processing applications require extracting relational structures, including event extraction, relation extraction, coreference resolution, etc. The prevalence of these applications makes us hard to exhaustively list them in this short summary, hence, we mainly focus on related works for the applications we experiment on.

Event extraction. Early works in event extraction mostly consider a pipelined approach (Nguyen and Grishman, 2015; Wang et al., 2019; Yang et al., 2019) to deal with event extraction. Some follow-up works argue that pipelined design leads to error propagation issues and hence propose end-to-end approaches to better capture dependencies between

each prediction (Lin et al., 2020; Li et al., 2013; Nguyen et al., 2016; Hsu et al., 2022b; Lu et al., 2021; Huang and Peng, 2021). However, recently, some empirical studies (Hsu et al., 2022b; Zhong and Chen, 2021; Fincke et al., 2022) also show that when an abundant amount of data is used to learn representations for each pipelined task, it is hard to conclude that joint learning approaches always provide a stronger result. This aligns with our discovery in experiments — even though we apply a pipelined approach with a simple sequence tagging framework on event extraction, with the help of priming to learn more condition-aware contextualized representation, we can still achieve very strong performance on multiple datasets.

Relation extraction. End-to-end relation extraction can usually be solved using two categories of approaches. The first one is to directly perform joint inference on named entities and their relation(s) (Zheng et al., 2017; Wang and Lu, 2020; Katiyar and Cardie, 2017; Sun et al., 2019; Miwa and Bansal, 2016; Fu et al., 2019). The second category is to perform a pipeline that first extracts named entities, and then performs relation classification (Wu and He, 2019; Hsu et al., 2022a; Lyu and Chen, 2021; Peng et al., 2020; Zhou and Chen, 2021a; Lu et al., 2022), which assumes that both the head-entity and tail-entity are given. Yet, in our unified formulation for relational structure extraction tasks, we extract tail-entities and their corresponding relation types for a given head-entity, which is more similar to a less frequently studied framework called cascading approaches (Wei et al., 2020; Yu et al., 2020). Despite being a less popular formulation to deal with end-to-end relation extraction, TAGPRIME presents a strong performance compared to prior studies, showcasing the practicality and effectiveness of our unified formulation.

Task-oriented semantic parsing. Task-oriented semantic parsing, which focuses on intent classification and slot filling, has a long history of development (Tür et al., 2010; Gupta et al., 2018; Li et al., 2021; Zhang et al., 2018; Louvan and Magnini, 2020). Recently, some more advanced neural network-based approaches have been proposed, such as MLP-mixer (Fusco et al., 2022) or sequence-to-sequence formulation (Desai et al., 2021). Among them, JointBERT (Chen et al., 2019b), a sequence-tagging-based model that is trained to jointly predict intent and extract slots, serves as a widely-used baseline due to its simplicity. Our approach benefits from the same simplicity as JointBERT and can further improve its performance.

3 Method

We first introduce our view to unify RSE problems and then discuss how TAGPRIME approaches this problem under a unified framework of sequence tagging model with priming.

3.1 A Unified Formulation of RSE

Given an input text $\mathbf{x} = [x_1, x_2, ..., x_n]$ and a condition c, The RSE tasks identify a list of spans $\mathbf{s}^c = [s_1^c, s_2^c, ..., s_l^c]$ and their corresponding relationships or attributes $\mathbf{r}^c = [r_1^c, r_2^c, ..., r_l^c]$ towards the condition c, where $r_i^c \in \mathcal{A}$ and $\mathcal{A}$ is the set of all possible relationships or attributes. Many NLP tasks can be formulated as an RSE task. We showcase how this formulation can be applied to event extraction, entity relation extraction, and task-oriented semantic parsing below.

End-to-end event extraction. End-to-end event extraction aims to extract events from given texts (Ma et al., 2020; Hsu et al., 2022b; Yang et al., 2019). An event contains a trigger, which is the textual span that best represents the occurrence of an event, and several arguments, which are the participants involved in the event with different argument roles. We consider a pipeline solution — after the event triggers are identified, an argument extraction model extracts the event arguments and their corresponding roles for each given event trigger. Under the RSE formulation, the condition c is the given event trigger, and the target spans $\mathbf{s}^c$ and the relationships $\mathbf{r}^c$ are the arguments and their argument roles, respectively.

End-to-end relation extraction. Relation extraction identifies entities and their relations from texts, and it is usually solved by pipeline approaches — first extracting named entities and then predicting relations for each entity-pair (Wu and He, 2019; Zhong and Chen, 2021). Under the new formulation, an RSE model is used to predict *tail-entities* and the relations for each extracted named entity that serves as the *head-entity*. For example, in Figure 1(b), we extract the tail-entities (*"Iraqi"* and *"base"*) and their relation (*"Part-Whole"* and *"ART"*) for the head-entity, *"military"*. In this way,

each given head-entity is the condition c, and the extracted tail-entities are $\mathbf{s}^c$, with relations, $\mathbf{r}^c$.

Task-oriented semantic parsing. Task-oriented semantic parsing aims to classify the intent and parse the semantic slots in an utterance (to a task-oriented dialog system) (Li et al., 2021; Gupta et al., 2018). To fit into our formulation, we first predict the intent and then use a *relational structure extraction* model to predict the slots ($\mathbf{s}^c$) as well as their semantic roles ($\mathbf{r}^c$) for the given intent (c).

3.2 Sequence Tagging Model for RSE

We hereby introduce the typical way of applying a sequence tagging model to unifiedly solve relational structure extraction. The goal of our sequence tagging model for relational structure extraction is to predict the BIO-tag sequence $\mathbf{y} = [y_1, y_2, ..., y_n]$, where each y_i is the corresponding tag for each token x_i in the input text. The BIO-tag sequence can then be decoded to represent the extracted spans $\mathbf{s}^c$ (and their relationships $\mathbf{r}^c$).

Specifically, given an input text, we obtain the contextualized representation z_i for each token x_i by passing the passage to a pre-trained language model.[1] To embed the information of the condition c, one commonly-used technique is to add conditional features to z_i (Ma et al., 2020; Wei et al., 2020; Yang et al., 2019; Yu et al., 2020), as shown in Figure 1(a). For example, in Ma et al. (2020), they use a token embedding of the given event trigger word and a *learnable* event type feature as the conditional features for the task of event argument extraction. In such case, the feature of c will contain the contextualized word representation z_j, if x_j is the token that represents the given condition, i.e., event trigger. In our experimental setup, if the given condition can be represented as an input span, we will include the span embeddings as the conditional features together with the type embeddings, such as the cases for event extraction and relation extraction. If the condition is only a concept, such as the task-oriented semantic parsing case, the conditional features will only contain type embeddings. Augmented with these conditional features, the final representation for token x_i is further fed into multi-layer perceptrons and a conditional random field (CRF) layer (Lafferty et al., 2001) to predict the BIO-tag sequence $\mathbf{y}$, as illustrated in Figure 1(a).

[1] If a token x_i is split into multiple word pieces, we use the average embeddings of all its word pieces to be z_i, following the practice of Lin et al. (2020).

3.3 TAGPRIME

TAGPRIME follows the sequence tagging paradigm but utilizes the priming technique for better leverage information about the input condition.

Condition Priming. Motivated by previous work (Fincke et al., 2022), we consider priming to inject the information of the condition c to further improve the sequence tagging model. The priming mechanism informs the model of the conditional information by directly appending conditional information to the input text. However, unlike Fincke et al. (2022) that uses an integer string to represent features in a categorical style, we use a natural-language-styled indicator to better exploit the semantics of the condition. The indicators can be obtained by verbalizing the conditional information.

Take Figure 1(b) as an example, when extracting the tail-entities and the relationships for the *"military"* head-entity (condition c), we first verbalize the entity type of *"military"*, i.e., from *"Org"* to *"Organization"*. Then, the string *"military"* and *"Organization"* are appended to the input text, which serves as the information about the condition c.

The priming technique leverages the self-attention mechanism in pre-trained language models and makes the token representation z_i condition-aware. Hence, the representation of every z_i is more *task-specific* than the one in the model described in Section 3.2. More precisely, for tagging models without priming, the representation z_i usually captures more general information that focuses on the context of input text. For models with priming, the representation z_i is affected by the additional verbalized words when computing attention. Hence, z_i becomes more task-specific and more suitable for addressing the task (Zheng and Lapata, 2022; Zhong and Chen, 2021). Additionally, the priming method can be easily combined with conditional features described in Section 3.2. More discussion on this will be shown in Section 5.

Relationship Priming. The same idea of condition priming can also be extended to relationship. Specifically, we decompose a relational structure extraction task into several extraction subtasks, each of them only focusing on one single relationship r ($r \in \mathcal{A}$). Similar to the condition priming, we verbalize the relationship information and append related strings to the input text as well. Therefore, the representation z_i is aware of the relation-

ship r and specific for predicting spans with relationship r to the condition c.

For example, in Figure 1(c), for the given relationship *"Part-Whole"*, we first verbalized it into *"is part of"*. Then, the string *"is part of"* is appended to the input text together with the condition priming strings. The BIO-tag sequence can be decoded into those tail-entities $\mathbf{s}^c$ that form *"Part-Whole"* relationship(s) with the given head-entity *"military"*.

Discussion. A similar idea of appending tokens in the pre-trained language model's input to affect the output text representation has also been leveraged in Zhou and Chen (2021b); Zhong and Chen (2021). Yet, different from their works that only focus on relation classification and apply *instance-specific* information, our TAGPRIME with relationship priming method focuses on using *task-specific information*, because we decompose relational extraction into sub-tasks. We want that different task-specific representation can be learned for different sub-tasks, hence proposing relationship priming.

An underlying advantage of TAGPRIME with relationship priming is its ability to handle cases containing multi-relationships. After we decompose a relational structure extraction task into several extraction subtasks, we do not perform any filtering to address conflict relationship predictions between the same condition and extracted span. This is to enlarge our model's generality to different scenarios.

4 Experiments

To study the effectiveness of TAGPRIME, we consider three NLP tasks: (1) end-to-end event extraction, (2) end-to-end relation extraction, and (3) task-oriented semantic parsing. All the results are the average of five runs with different random seeds.

4.1 End-to-End Event Extraction

Datasets. We consider the two most widely-used event extraction datasets, ACE-2005 (Doddington et al., 2004) and ERE (Song et al., 2015). For ACE-2005 (ACE05-E), we experiment on the English and Chinese portions and keep 33 event types and 22 roles, as suggested in previous works (Wadden et al., 2019; Hsu et al., 2022b). For ERE, we consider the English and Spanish annotations and follow the preprocessing of Lin et al. (2020) to keep 38 event types and 21 roles.

Baselines. We consider the following end-to-end event extraction models, including DyGIE++ (Wadden et al., 2019), TANL (Paolini et al., 2021), Text2Event (Lu et al., 2021), OneIE (Lin et al., 2020), and DEGREE (Hsu et al., 2022b). Since TAGPRIME requires trigger predictions, we simply take the trigger predictions made by a simple sequence tagging model trained with multi-tasking on trigger detection and named entity recognition.

For TAGPRIME, DyGIE++, and OneIE, we consider BERT-large (Devlin et al., 2019) for ACE05-E (en) and ERE (en), and consider XLM-RoBERTa-large (Conneau et al., 2020) for ACE05-E (zh) and ERE (es). For generation-based models, we consider BART-large (Lewis et al., 2020) for DEGREE, T5-base (Raffel et al., 2020) for TANL, and T5-large (Raffel et al., 2020) for Text2Event, as suggested by their original papers.

Implementation details. The followings are the training details for all baselines:

- **DyGIE++** (Wadden et al., 2019): we use the released training script[2] with the default parameters.
- **TANL** (Paolini et al., 2021): we report the numbers from the original paper.
- **Text2Event** (Lu et al., 2021): we report the numbers from the original paper.
- **OneIE** (Lin et al., 2020): we use the released training script[3] with the default parameters.
- **DEGREE** (Hsu et al., 2022b): we report the numbers from the original paper.
- **TAGPRIME** (ours): We fine-tune pre-trained language models with the dropout rate being 0.2. We use AdamW optimizer. For parameters that are not pre-trained we set the learning rate to 10^{-3} and the weight decay to 10^{-3}. For parameters that are not pre-trained we set the learning rate to 10^{-5} and the weight decay to 10^{-5}. We consider the linear scheduler with a warm-up, where the warm-up epoch is 5. The number of epochs is 90. The training batch size is set to 6. For conditional token features and learnable features, the dimension is set to 100. It takes around 6 hours to train with a NVIDIA RTX A6000 with 48GB memory.

Evaluation metrics. Following previous works (Wadden et al., 2019; Lin et al., 2020), we measure the correctness of arguments based on whether the offsets of the argument span match or not. We

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

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

Model	ACE05-E (en)			ACE05-E (zh)			ERE (en)			ERE (es)		
	Tri-C	Arg-I	Arg-C	Tri-C	Arg-I	Arg-C	Tri-C	Arg-I	Arg-C	Tri-C	Arg-I	Arg-C
DyGIE++* (Wadden et al., 2019)	69.7	53.0	48.8	72.3	63.0	59.3	58.0	51.4	48.0	65.8	49.2	46.6
TANL (Paolini et al., 2021)	68.4	50.1	47.6	-	-	-	54.7	46.6	43.2	-	-	-
Text2Event (Lu et al., 2021)	71.9	-	53.8	-	-	-	59.4	-	48.3	-	-	-
OneIE* (Lin et al., 2020)	74.7	59.2	56.8	73.3	63.4	60.5	57.0	50.1	46.5	66.5	54.5	52.2
DEGREE (Hsu et al., 2022b)	73.3	-	55.8	-	-	-	57.1	-	49.6	-	-	-
TAGPRIME w/ Cond. Priming	74.6	**60.0**	56.8	71.9	63.2	60.5	57.3	52.1	49.3	66.3	**55.2**	52.6
TAGPRIME w/ Cond. & Rela. Priming	74.6	59.8	**58.3**	71.9	**64.7**	**62.4**	57.3	**52.4**	**49.9**	66.3	55.1	**53.6**

Table 1: Results of end-to-end event extraction. All values are micro F1-score, and we highlight highest scores with boldface. TAGPRIME with conditional and relationship priming achieves more than 1.4 Arg-C F1-score improvements in three out of four datasets. *We reproduce the results using their released code.

consider argument identification F1-score (Arg-I), which cares about only the offset correctness, and argument classification F1-score (Arg-C), which cares about both offsets and the role types. We also report trigger classification F1-score (Tri-C), although it is not our main focus as the triggers are provided via other models and we just use their predictions to simulate the end-to-end scenarios.

Results. Table 1 shows the results of end-to-end event extraction on various datasets and languages. Although simple, TAGPRIME surprisingly has decent performance and achieves better results than the state-of-the-art models in terms of argument F1-scores. We attribute the good performance to the design of priming, which leverages the semantics of the condition and makes the representations more task-specific. It is worth noting that considering relationship priming further improves the results, which again shows the importance of task-specific representations.

4.2 End-to-End Relation Extraction

Datasets. We consider two popular end-to-end relation extraction datasets, ACE04 and ACE05 (Doddington et al., 2004), denoted as ACE04-R and ACE05-R. Both datasets consider 7 named entity types and 6 different relations. We follow the same procedure in Zhong and Chen (2021) to preprocess the data and split the datasets. We refer readers to their papers for more details about the datasets.

Baselines. We compare to the following end-to-end relation extraction models: Table-Sequence (Wang and Lu, 2020), PFN (Yan et al., 2021), and Cascade-SRN (both late fusion and early fusion) (Wang et al., 2022). Additionally, we consider PURE (Zhong and Chen, 2021), which also takes a pipelined approach to solve end-to-end relation extraction. To fairly compare with prior works, we use PURE's named entity predictions on the test set for TAGPRIME to perform relational structure extraction.[4] In order to be consistent with our other tasks, we adopt the single sentence setting (Zhong and Chen, 2021) for our model. However, we also list baselines with cross-sentence settings, such as PURE's and UniRE (Wang et al., 2021)'s results with cross-sentence context as input. All the models use ALBERT-xxlarge-v1 (Lan et al., 2020) as the pre-trained language models.

Implementation details. The followings are the training details for all baselines:

- **Table-Sequence** (Wang and Lu, 2020): we report the numbers from the original paper.
- **Cascade-SRN** (Wang et al., 2022): we report the numbers from the original paper.
- **PURE** (Zhong and Chen, 2021): we report the numbers from the original paper.
- **PFN** (Yan et al., 2021): we report the numbers from the original paper.
- **UniRE** (Wang et al., 2021): we report the numbers from the original paper.
- **TAGPRIME** (ours): We fine-tune pre-trained language models with the dropout rate being 0.2. We use AdamW optimizer. For parameters that are not pre-trained we set the learning rate to 10^{-3} and the weight decay to 10^{-3}. For parameters that are not pre-trained we set the learning rate to 2×10^{-5} and the weight decay to 10^{-5}. We consider the linear scheduler with a warm-up, where the warm-up epoch is 5. The number of epochs is 30. The training batch size is set to 32. For conditional token features and learnable features, the dimension is set to 100.

[4] We get PURE's named entity recognition predictions by retraining PURE's named entity recognition model.

Model	ACE05-R			ACE04-R		
	Ent	Rel	Rel+	Ent	Rel	Rel+
Table-Sequence (Wang and Lu, 2020)	89.5	67.6	64.3	88.6	63.3	59.6
PFN (Yan et al., 2021)	89.0	-	66.8	89.3	-	62.5
Cascade-SRN (late fusion) (Wang et al., 2022)	89.4	-	65.9	-	-	-
Cascade-SRN (early fusion) (Wang et al., 2022)	89.8	-	67.1	-	-	-
PURE (Zhong and Chen, 2021)	89.7	69.0	65.6	88.8	64.7	60.2
PURE$^{\diamond}$ (Zhong and Chen, 2021)	90.9	69.4	67.0	90.3	66.1	62.2
UniRE$^{\diamond}$ (Wang et al., 2021)	90.2	-	66.0	89.5	-	**63.0**
TAGPRIME w/ Cond. Priming	89.6	69.7	67.3	89.0	65.2	61.6
TAGPRIME w/ Cond. & Rela. Priming	89.6	**70.4**	**68.1**	89.0	**66.2**	62.3

Table 2: Results of end-to-end relation extraction. All values are micro F1-score with the highest value in boldface. TAGPRIME achieves the best performance in ACE05-R and competitive results on ACE04-R despite we get slightly lower entity scores compared to PFN. $^{\diamond}$indicates the use of cross-sentence context information.

It takes around 20 hours to train with a NVIDIA RTX A6000 with 48GB memory.

Evaluation metrics. We follow the standard evaluation setting with prior works (Bekoulis et al., 2018; Zhong and Chen, 2021) and use micro F1-score as the evaluation metric. For named entity recognition, a predicted entity is considered as a correct prediction if its span and the entity type are both correct. We denote the score as "Ent" and report the scores even though it is not our main focus for evaluation. For relation extraction, two evaluation metrics are considered: (1) Rel: a predicted relation is considered as correct when the boundaries of head-entity span and tail-entity span are correct and the predicted relation type is correct; (2) Rel+: a stricter evaluation of Rel, where they additionally required that the entity types of head-entity span and tail-entity must also be correct.

Results. The results of end-to-end relation extraction are presented in Table 2. From the table, we observe that TAGPRIME has the best performance on ACE05-R and outperforms most baselines on ACE04-R. This shows the effectiveness of TAGPRIME. Similar to the results of event extraction, considering relationship priming makes the representations more relationship-aware and leads to performance improvement.

4.3 Task-Oriented Semantic Parsing

Datasets. We choose MTOP (Li et al., 2021), a multilingual dataset on semantic parsing for task-oriented dialog systems. We specifically consider data in English (en), Spanish (es), French (fr), and German (de).

Baselines. We consider JointBERT (Chen et al., 2019b), the commonly used baseline for task-oriented semantic parsing. We directly use the predicted intents by JointBERT as the condition of TAGPRIME for a fair comparison. Both TAGPRIME and JointBERT are trained with XLM-RoBERTa-large (Conneau et al., 2020). Unlike event extraction and relation extraction, the condition of task-oriented semantics parsing (intent) does not include the word span, therefore, only a type feature embedding is contained in the conditional features for TAGPRIME in this experiment.

Implementation details. The followings are the training details for all baselines:

- **JointBERT** (Chen et al., 2019b): we use the training script[5] with the default parameters.
- **TAGPRIME** (ours): We fine-tune pre-trained language models with the dropout rate being 0.2. We use AdamW optimizer. For parameters that are not pre-trained we set the learning rate to 10^{-3} and the weight decay to 10^{-3}. For parameters that are not pre-trained we set the learning rate to 10^{-5} and the weight decay to 10^{-5}. We consider the linear scheduler with warm-up, where the warm-up epoch is 5. The number of epochs is 90. The training batch size is set to 6. For conditional token features and learnable features, the dimension is set to 100. It takes around 4 hours to train with a NVIDIA RTX A6000 with 48GB memory.

Evaluation metrics. We following MTOP (Li et al., 2021) to consider slot identification (Slot-I) and slot classification (Slot-C) F1-scores. Even though we focus on the performance of slot filling, we also report the intent classification accuracy.

Results. As demonstrated in Table 3, TAGPRIME

[5] https://github.com/monologg/JointBERT

Model	MTOP (en)			MTOP (es)			MTOP (fr)			MTOP (de)		
	Intent	Slot-I	Slot-C	Intent	Slot-I	Slot-C	Intent	Slot-I	Slot-C	Intent	Slot-I	Slot-C
JointBERT (Li et al., 2021)	96.7	-	92.8	95.2	-	89.9	94.8	-	88.3	95.7	-	88.0
JointBERT (reproduced)	97.1	94.2	92.7	96.6	91.6	89.5	95.8	90.2	87.7	96.5	89.2	87.6
TAGPRIME + Cond. Priming	97.1	**94.8**	93.4	96.6	91.6	90.3	95.8	**90.6**	88.6	96.5	**89.6**	87.9
TAGPRIME + Cond. & Rela. Priming	97.1	94.7	**93.5**	96.6	**91.8**	**90.7**	95.8	**90.6**	**89.1**	96.5	89.5	**88.1**

Table 3: Results of task-oriented semantic parsing. Intend scores are measured in accuracy(%) and slot scores are micro-F1 scores. The highest value is in bold.

Case	Cond.		Rela.		ACE05-E (en)		ACE05-E (zh)		MTOP (es)		MTOP (fr)		ACE05-R (en)		ACE04-R (en)		Average
	Feat.	Prim.	Feat.	Prim.	Arg-I	Arg-C	Arg-I	Arg-C	Slot-I	Slot-C	Slot-I	Slot-C	Rel	Rel+	Rel	Rel+	
1	✗	✗	✗	✗	57.8	54.2	60.2	57.2	91.8	90.2	90.5	88.4	67.8	65.5	62.2	58.9	69.1
2	✓	✗	✗	✗	58.1	55.3	60.4	58.1	**92.0**	90.4	90.6	88.6	67.5	65.2	61.8	58.4	69.4
3	✗	✓	✗	✗	59.6	56.7	62.0	59.7	91.8	90.4	**90.7**	88.8	69.6	67.2	64.7	60.7	70.6
4	✓	✓	✗	✗	**60.0**	56.8	63.2	60.5	91.6	90.3	90.6	88.7	69.7	67.3	65.2	61.6	70.9
5	✓	✗	✓	✗	57.3	55.3	61.4	59.4	91.7	90.5	90.2	88.5	68.0	65.6	61.6	58.3	69.6
6	✗	✓	✗	✓	59.3	57.6	63.0	61.2	91.7	90.5	90.5	88.9	**70.6**	**68.2**	66.0	62.2	71.4
7	✓	✓	✗	✓	59.8	**58.3**	**64.7**	**62.4**	91.8	**90.7**	90.6	**89.1**	70.4	68.1	**66.2**	**62.3**	**71.8**
8	✓	✓	✓	✓	59.7	58.0	64.3	**62.4**	91.5	90.4	90.6	**89.1**	70.5	68.1	65.8	62.2	71.7

Table 4: The ablation study results for three different tasks. The average column calculates the average scores of the stricter evaluation metrics (i.e, Arg-C, Slot-C, and Rel+) for each dataset. From the table, we demonstrate priming technique is the key attribute that make our sequence tagging model stronger than models with learnable features, which is the typical way of using sequence tagging models for relational structure extraction.

achieves a better performance than the baselines. Again, considering relationship priming leads to further improvement. It is worth noting that TAGPRIME is effective for different languages, which shows the generality of TAGPRIME.

4.4 Summary

We show the superiority of TAGPRIME on three different tasks (including ten different datasets across five different languages). Although being a unified and simple model, the results suggest that TAGPRIME can achieve competitive results for tasks requiring extracting relational structures.

5 Analysis

In this section, we study two questions: (1) What is the effectiveness of priming techniques compared to learnable features? (2) Relationship priming boosts the performance of TAGPRIME, but the task decomposition could slightly slow down the inference speed. Can we mitigate this issue?

To answer the first question, we conduct ablation experiments on sequence tagging models using different combinations of learnable features or/and adding information through the priming technique (Section 5.1). For the second question, we propose a simple modification to TAGPRIME so that we can flexibly control the number of layers to fuse priming information to contextualized representations. The modified TAGPRIME can serve as an efficient approximation of TAGPRIME (Section 5.2).

5.1 Ablation Study

We focus on the setting where we alter the choices on how to include the type information of the condition c and the relationship information r. Table 4 demonstrates our experimental results.

Comparing the first four cases in Table 4, we observe that the addition of type features is useful in general, and using the priming technique is a more effective way to incorporate conditional information. For models in case 5 to case 8, the relationship decomposition formulation described in Section 3.3 is applied. Comparing case 2 to case 5, we can see that simply applying the relationship decomposition formulation for solving relational structure extraction does not lead to improvements if the way to embed the relationship r is only through learnable features. However, comparing case 3 to case 6 and case 4 to case 7, we show that the relationship priming approach makes the representation z_i well capture the attribute of the queried relationship, thus, better exploiting the advantage of the relationship decomposition formulation and gaining improvements. Note that we conducted preliminary experiments that use pretrained language models' representations of the same verbalized token to be the initialization of the learnable type feature embedding, but the method

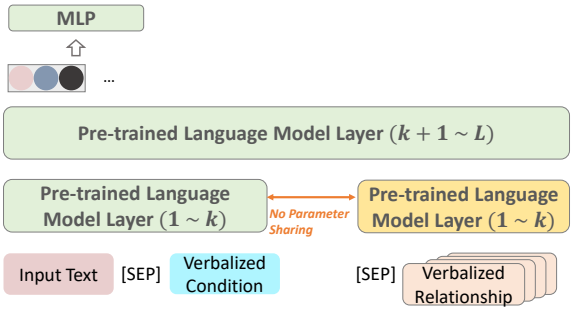


Figure 2: The illustration of our efficient approximation of TAGPRIME, which separates the pre-trained language models into two halves to enable parallel encoding in TAGPRIME, leading to faster inference.

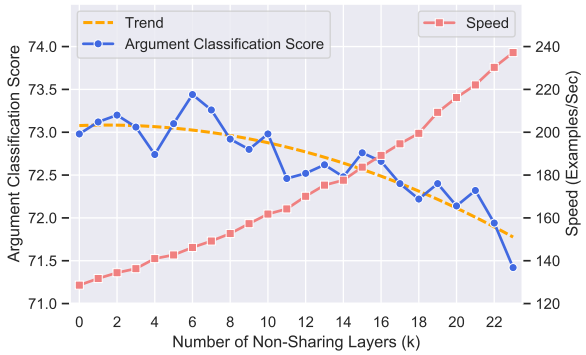


Figure 3: Analysis about the performance influence and inference speed impact of our efficient approximated TAGPRIME when the sharing layers vary. The blue line is the argument classification F1 score and the red line is the inference speed curve.

shows similar results with the random initialization, hence, we stick to random initialization on the learnable type features.

5.2 Efficient approximation of TAGPRIME

To make TAGPRIME to inference faster, we perform two modifications to TAGPRIME: (1) We first separate the pre-trained language model, which contains L layers, into two halves — one with the first k layers, the other one is the remaining layers. (2) We copy the first half of the language model to another module. When an input passage is fed into the model. We use the original first half to encode the input text as well as the verbalized condition, and we use the copied first half to encode the verbalized relation. Finally, the encoded representations will be fed into the second half layers, as illustrated in Figure 2. The value of k is adjustable, where when $k = 0$, it represents the TAGPRIME with condition and relationship priming, and when $k = L$, it is TAGPRIME with condition priming.

Since the encoding stage of the input text and the verbalized relationship is separated, we can accelerate the inference time of our modified TAGPRIME through parallel encoding. More precisely, our modified TAGPRIME can aggregate instances that share the same passage and verbalized condition. For those instances, TAGPRIME only needs to perform the encoding once on their input passage part,[6] and paired with several separated embedded verbalized relationships, which could be parallelly encoded together.

We conduct experiments on the ACE05-E (en) dataset to test our modification. In order to better analyze the results and isolate the influence from the pipelined errors, we report the results on the event argument extraction when gold event triggers are given. The experimental results are shown in Figure 3. First, we investigate the performance influence of our modification. We find that when $k \leq 10$, the performance of our modified TAGPRIME is strong in general and is comparable with TAGPRIME with the condition and relationship priming. To compare the efficiency of the model, we benchmark the inference time by performing inference on the whole testing dataset fifty times and calculate the average speed, which is measured by checking how many instances can be processed per second. The red line in Figure 3 shows the results. We observe that for our modified TAGPRIME with $k = 10$, its inference speed is around 30% faster than the TAGPRIME with the condition and relationship priming, but they perform similarly.

6 Conclusion

In this work, we take a unified view of tasks requiring extracting relational structures and present TAGPRIME, a simple, unified, effective, and general sequence tagging model. The key idea is applying priming, a small trick to make the representations task-specific by appending condition-related and relationship-related strings to the input text. Our experimental results demonstrate that TAGPRIME is general to different tasks in various languages and can serve as a strong baseline for future research on relational structure extraction.

[6]The string of verbalized relationship is usually much shorter than the input passage, hence, in most cases, the major part of the input for an instance is the input text and which requires more computations.

Acknowledgments

We thank anonymous reviewers for their helpful feedback. We thank the UCLA PLUS-Lab and UCLA-NLP group members for the valuable discussions and comments. This research was supported in part by AFOSR MURI via Grant #FA9550-22-1-0380, Defense Advanced Research Project Agency (DARPA) via Grant #HR00112290103/HR0011260656, the Intelligence Advanced Research Projects Activity (IARPA) via Contract No. 2019-19051600007, National Science Foundation (NSF) via Award No. 2200274, and a research award sponsored by CISCO.

Limitations

As we point out in Section 5, one of the limitations in TAGPRIME is the inference speed. When we perform TAGPRIME with condition and relationship priming, we requires more turns of sequence tagging processes than typical sequence tagging models. Observing this, we propose a simple way to mitigate such issue and increase the inference speed with only a small performance drop. Despite such effort, it is still slightly slower than the model requires only one pass of sequence labeling.

The other potential limitation of our method is that we assume the condition and relationship can be verbalized. However, in practice, there could be cases that the verbalization is hard to be done. Considering this, we do conduct preliminary experiments of applying TAGPRIME with special tokens priming rather than verbalized tokens. However, our preliminary results show that such method's performance is less stable and weaker than we can achieve with TAGPRIME.

Ethics Considerations

TAGPRIME fine-tunes the pre-trained language models (Devlin et al., 2019; Lan et al., 2020). There have been works showing the potential bias in pre-trained language models. Although with a low possibility, especially after our finetuning, it is possible for our model to make counterfactual, and biased predictions, which may cause ethical concerns. We suggest carefully examining those potential issues before deploying the model in any real-world applications.

References

Giannis Bekoulis, Johannes Deleu, Thomas Demeester, and Chris Develder. 2018. Adversarial training for multi-context joint entity and relation extraction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Jiaao Chen, Jianshu Chen, and Zhou Yu. 2019a. Incorporating structured commonsense knowledge in story completion. In *The Thirty-Third AAAI Conference on Artificial Intelligence (AAAI)*.

Qian Chen, Zhu Zhuo, and Wen Wang. 2019b. BERT for joint intent classification and slot filling. *arXiv preprint arXiv:1902.10909*.

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, ACL*.

Shrey Desai, Akshat Shrivastava, Alexander Zotov, and Ahmed Aly. 2021. Low-resource task-oriented semantic parsing via intrinsic modeling. *arXiv preprint arxiv.2104.07224*.

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, (NAACL-HLT)*.

George R. Doddington, Alexis Mitchell, Mark A. Przybocki, Lance A. Ramshaw, Stephanie M. Strassel, and Ralph M. Weischedel. 2004. The automatic content extraction (ACE) program - tasks, data, and evaluation. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC)*.

Steven Fincke, Shantanu Agarwal, Scott Miller, and Elizabeth Boschee. 2022. Language model priming for cross-lingual event extraction. In *The Thirty-Sixth AAAI Conference on Artificial Intelligence, (AAAI)*.

Tsu-Jui Fu, Peng-Hsuan Li, and Wei-Yun Ma. 2019. Graphrel: Modeling text as relational graphs for joint entity and relation extraction. In *Proceedings of the 57th Conference of the Association for Computational Linguistics (ACL)*.

Francesco Fusco, Damian Pascual, and Peter Staar. 2022. pnlp-mixer: an efficient all-mlp architecture for language. *arXiv preprint arxiv.2202.04350*.

Sonal Gupta, Rushin Shah, Mrinal Mohit, Anuj Kumar, and Mike Lewis. 2018. Semantic parsing for task oriented dialog using hierarchical representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

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

I-Hung Hsu, Xiao Guo, Premkumar Natarajan, and Nanyun Peng. 2022a. Discourse-level relation extraction via graph pooling. In *The Thirty-Sixth AAAI Conference On Artificial Intelligence Workshop on Deep Learning on Graphs: Method and Applications (DLG-AAAI)*.

I-Hung Hsu, Kuan-Hao Huang, Elizabeth Boschee, Scott Miller, Prem Natarajan, Kai-Wei Chang, and Nanyun Peng. 2022b. 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 (NAACL-HLT)*.

Kuan-Hao Huang, I-Hung Hsu, Premkumar Natarajan, Kai-Wei Chang, and Nanyun Peng. 2022. Multilingual generative language models for zero-shot cross-lingual event argument extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (ACL)*.

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

Arzoo Katiyar and Claire Cardie. 2017. Going out on a limb: Joint extraction of entity mentions and relations without dependency trees. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL*.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning (ICML)*.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A lite BERT for self-supervised learning of language representations. In *8th International Conference on Learning Representations (ICLR)*.

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

Haoran Li, Abhinav Arora, Shuohui Chen, Anchit Gupta, Sonal Gupta, and Yashar Mehdad. 2021. MTOP: A comprehensive multilingual task-oriented semantic parsing benchmark. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*.

Qi Li, Heng Ji, and Liang Huang. 2013. Joint event extraction via structured prediction with global features. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL)*.

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

Shuman Liu, Hongshen Chen, ZhaDBLP:conf/acl/LiuFCRYL18ochun Ren, Yang Feng, Qun Liu, and Dawei Yin. 2018. Knowledge diffusion for neural dialogue generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Samuel Louvan and Bernardo Magnini. 2020. Recent neural methods on slot filling and intent classification for task-oriented dialogue systems: A survey. In *Proceedings of the 28th International Conference on Computational Linguistics (COLING)*.

Keming Lu, I-Hung Hsu, Wenxuan Zhou, Mingyu Derek Ma, and Muhao Chen. 2022. Summarization as indirect supervision for relation extraction. *arXiv preprint arXiv:2205.09837*.

Yaojie Lu, Hongyu Lin, Jin Xu, Xianpei Han, Jialong Tang, Annan Li, Le Sun, Meng Liao, and Shaoyi Chen. 2021. Text2event: Controllable sequence-to-structure generation for end-to-end event extraction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (ACL/IJCNLP)*.

Shengfei Lyu and Huanhuan Chen. 2021. Relation classification with entity type restriction. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*.

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-Findings)*.

Makoto Miwa and Mohit Bansal. 2016. End-to-end relation extraction using lstms on sequences and tree structures. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL*.

Thien Huu Nguyen, Kyunghyun Cho, and Ralph Grishman. 2016. Joint event extraction via recurrent neural networks. In *The 2016 Conference of the North*

American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT).

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

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

Hao Peng, Tianyu Gao, Xu Han, Yankai Lin, Peng Li, Zhiyuan Liu, Maosong Sun, and Jie Zhou. 2020. Learning from Context or Names? An Empirical Study on Neural Relation Extraction. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

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

Zhiyi Song, Ann Bies, Stephanie M. Strassel, Tom Riese, Justin Mott, Joe Ellis, Jonathan Wright, Seth Kulick, Neville Ryant, and Xiaoyi Ma. 2015. From light to rich ERE: annotation of entities, relations, and events. In *Proceedings of the The 3rd Workshop on EVENTS: Definition, Detection, Coreference, and Representation, (EVENTS@HLP-NAACL)*.

Changzhi Sun, Yeyun Gong, Yuanbin Wu, Ming Gong, Daxin Jiang, Man Lan, Shiliang Sun, and Nan Duan. 2019. Joint type inference on entities and relations via graph convolutional networks. In *Proceedings of the 57th Conference of the Association for Computational Linguistics (ACL)*.

Gökhan Tür, Dilek Hakkani-Tür, and Larry P. Heck. 2010. What is left to be understood in atis? In *2010 IEEE Spoken Language Technology Workshop (SLT)*.

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)*.

An Wang, Ao Liu, Hieu Hanh Le, and Haruo Yokota. 2022. Towards effective multi-task interaction for entity-relation extraction: A unified framework with selection recurrent network. *arXiv preprint arXiv:2202.07281*.

Jue Wang and Wei Lu. 2020. Two are better than one: Joint entity and relation extraction with table-sequence encoders. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

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

Yijun Wang, Changzhi Sun, Yuanbin Wu, Hao Zhou, Lei Li, and Junchi Yan. 2021. Unire: A unified label space for entity relation extraction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (ACL/IJCNLP)*.

Zhepei Wei, Jianlin Su, Yue Wang, Yuan Tian, and Yi Chang. 2020. A novel cascade binary tagging framework for relational triple extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Shanchan Wu and Yifan He. 2019. Enriching pre-trained language model with entity information for relation classification. In *Proceedings of the 28th ACM international conference on information and knowledge management*, pages 2361–2364.

Zhiheng Yan, Chong Zhang, Jinlan Fu, Qi Zhang, and Zhongyu Wei. 2021. A partition filter network for joint entity and relation extraction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

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

Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec. 2021. QA-GNN: Reasoning with language models and knowledge graphs for question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Bowen Yu, Zhenyu Zhang, Xiaobo Shu, Tingwen Liu, Yubin Wang, Bin Wang, and Sujian Li. 2020. Joint extraction of entities and relations based on a novel decomposition strategy. In *24th European Conference on Artificial Intelligence (ECAI)*.

Dongjie Zhang, Zheng Fang, Yanan Cao, Yanbing Liu, Xiaojun Chen, and Jianlong Tan. 2018. Attention-based RNN model for joint extraction of intent and word slot based on a tagging strategy. In *Artificial Neural Networks and Machine Learning (ICANN)*.

Hao Zheng and Mirella Lapata. 2022. Disentangled sequence to sequence learning for compositional generalization. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Suncong Zheng, Feng Wang, Hongyun Bao, Yuexing Hao, Peng Zhou, and Bo Xu. 2017. Joint extraction of entities and relations based on a novel tagging scheme. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017*.

Zexuan Zhong and Danqi Chen. 2021. A frustratingly easy approach for entity and relation extraction. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*.

Wenxuan Zhou and Muhao Chen. 2021a. An improved baseline for sentence-level relation extraction. *arXiv preprint arXiv:2102.01373*.

Wenxuan Zhou and Muhao Chen. 2021b. An improved baseline for sentence-level relation extraction. *arXiv preprint arXiv:2102.0137*.

A Detailed Results

Table 5, 6, and 7 lists the detailed results (mean and standard deviation) of TAGPRIME.

Model	ACE05-E (en)		ACE05-E (zh)		ERE (en)		ERE (es)	
	Arg-I	Arg-C	Arg-I	Arg-C	Arg-I	Arg-C	Arg-I	Arg-C
DyGIE++* (Wadden et al., 2019)	53.0	48.8	63.0	59.3	51.4	48.0	49.2	46.6
TANL (Paolini et al., 2021)	50.1	47.6	-	-	46.6	43.2	-	-
Text2Event (Lu et al., 2021)	-	53.8	-	-	-	48.3	-	-
OneIE* (Lin et al., 2020)	59.2	56.8	63.4	60.5	50.1	46.5	54.5	52.2
DEGREE (Hsu et al., 2022b)	-	55.8	-	-	-	49.6	-	-
TAGPRIME w/ Cond. Priming	**60.0**±0.47	56.8±0.54	63.2±0.74	60.5±0.73	52.1±0.15	49.3±0.28	**55.2**±0.79	52.6±1.11
TAGPRIME w/ Cond. & Rel. Priming	59.8±0.53	**58.3**±0.67	**64.7**±0.88	**62.4**±0.85	**52.4**±0.41	**49.9**±0.60	55.1±0.89	**53.6**±0.83

Table 5: Detailed results of end-to-end event extraction (mean±std). All values are micro F1-score, and we highlight the highest scores with boldface. *We reproduce the results using their released code.

Model	ACE05-R		ACE04-R	
	Rel	Rel+	Rel	Rel+
Table-Sequence (Wang and Lu, 2020)	67.6	64.3	63.3	59.6
PFN (Yan et al., 2021)	-	66.8	-	62.5
Cascade-SRN (late fusion) (Wang et al., 2022)	-	65.9	-	-
Cascade-SRN (early fusion) (Wang et al., 2022)	-	67.1	-	-
PURE (Zhong and Chen, 2021)	69.0	65.6	64.7	60.2
PURE$^{\diamond}$ (Zhong and Chen, 2021)	69.4	67.0	66.1	62.2
UniRE$^{\diamond}$ (Wang et al., 2021)	-	66.0	-	**63.0**
TAGPRIME w/ Cond. Priming	69.7±0.73	67.3±0.61	65.2±1.56	61.6±1.65
TAGPRIME w/ Cond. & Rela. Priming	**70.4**±0.64	**68.1**±0.64	**66.2**±1.51	62.3±1.19

Table 6: Detailed results of end-to-end relation extraction (mean±std). All values are micro F1-score with the highest value in bold. Note that in ACE04-R, the experiment was conducted and evaluated through 5-fold cross-validation, hence the variance is slightly larger compared to ACE05-R, which fixes the test set for every run with a different random seed. $^{\diamond}$indicates the use of cross-sentence context information.

Model	MTOP (en)		MTOP (es)		MTOP (fr)		MTOP (de)	
	Slot-I	Slot-C	Slot-I	Slot-C	Slot-I	Slot-C	Slot-I	Slot-C
JointBERT (Li et al., 2021)	-	92.8	-	89.9	-	88.3	-	88.0
JointBERT (reproduced)	94.2	92.7	91.6	89.5	90.2	87.7	89.2	87.6
TAGPRIME + Cond. Priming	**94.8**±0.27	93.4±0.30	91.6±0.43	90.3±0.15	**90.6**±0.22	88.6±0.24	**89.6**±0.15	87.9±0.07
TAGPRIME + Cond. & Rela. Priming	94.7±0.07	**93.5**±0.13	**91.8**±0.16	**90.7**±0.14	**90.6**±0.36	**89.1**±0.35	89.5±0.34	**88.1**±0.36

Table 7: Detailed results of task-oriented semantic parsing (mean±std). Intend scores are measured in accuracy(%) and slot scores are micro-F1 scores. The highest value is in bold.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
Please see the "Limitations" section.

☑ A2. Did you discuss any potential risks of your work?
Please see the "Ethics Considerations" section

☑ A3. Do the abstract and introduction summarize the paper's main claims?
Section 1

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☑ Did you use or create scientific artifacts?

Section 4

☑ B1. Did you cite the creators of artifacts you used?
Section 4

☑ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
Section 4

☑ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
Section 4

☐ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
Not applicable. Left blank.

☐ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Not applicable. Left blank.

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
Section 4

C ☑ Did you run computational experiments?

Section 4

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Section 4 and Appendix A

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Section 4 and Appendix A

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
Section 4 and Appendix B

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
Section 4 and Appendix A

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Left blank.

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
No response.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
No response.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
No response.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
No response.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
No response.