TAGPRIME: A Unified Framework for Relational Structure Extraction

Anonymous ACL submission

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 of the tasks 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 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 that can deal with all of them.

In this work, we take a unified view of these tasks. We formulate them as a task that identifies arguments to a given condition and classifies their relationships, and call it the relational structure extraction (RSE) task. 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 (Fincke et al., 2022). TAGPRIME inherits the strength of sequence tagging models to unifyingly 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 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 will be shown in Section 5.
Our contributions can be summarized as follows. (1) We present a unified view of tasks that requires extracting relational structures, including end-to-end event extraction, end-to-end relation extraction, task-oriented semantic parsing, etc. (2) We present TAGPRIME, a unified sequence tagging model with priming to handle various relational structure extraction problems. (3) Thorough experiments on three different tasks show that TAGPRIME achieves comparable or even better performance than the current state-of-the-art on ten datasets in five different languages. (4) We propose a novel efficient approximation to speed up TAGPRIME during inference time without sacrificing too much performance.

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 Relational Structure Extraction with Sequence Tagging**

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

**3.1 A Unified Formulation of RSE**

Given an input text \( x = [x_1, x_2, \ldots, x_n] \) and a condition \( c \), the RSE tasks identify a list of spans \( s^c = [s^c_1, s^c_2, \ldots, s^c_j] \) and their corresponding relationships or attributes \( r^c = [r^c_1, r^c_2, \ldots, r^c_f] \) towards the condition \( c \), where \( r^c_i \in A \) and \( A \) 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 \( s^c \) and the relationships \( r^c \) 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 \( s^c \), with relations, \( r^c \).

**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 \( s^c \) as well as their semantic roles \( r^c \) 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 \( y = [y_1, y_2, \ldots, y_n] \), 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 \( s^c \) (and their relationships \( r^c \)).
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. 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 $y$, as illustrated in Figure 1(a).

### 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-specific information. 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 relationship $r$ 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 $s^*$ 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.

---

1 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).
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. All the implementation details can be found in Appendix A.1.

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 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.

### Table 1: Results of end-to-end event extraction.

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

* We reproduce the results using their released code. † indicates significant improvements per McNemar’s testing when compared to our reproduced OneIE.
Table 2: Results of end-to-end relation extraction. All values are micro F1-score with the highest value in boldface. TAGRIME achieves the best performance in ACE05-R and competitive results on ACE04-R despite we get slightly lower entity scores compared to PFN. ‡indicates the use of cross-sentence context information. †indicates significant improvements per McNemar’s testing when compared to PURE with sentence-level context.

<table>
<thead>
<tr>
<th>Model</th>
<th>ACE05-R</th>
<th></th>
<th>ACE04-R</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Ent</td>
<td>Rel</td>
<td>Rel+</td>
<td>Ent</td>
</tr>
<tr>
<td>Table-Sequence (Wang and Lu, 2020)</td>
<td>89.5</td>
<td>67.6</td>
<td>64.3</td>
<td>88.6</td>
</tr>
<tr>
<td>PFN (Yan et al., 2021)</td>
<td>89.0</td>
<td>-</td>
<td>66.8</td>
<td>89.3</td>
</tr>
<tr>
<td>Cascade-SRN (late fusion) (Wang et al., 2022)</td>
<td>89.4</td>
<td>-</td>
<td>65.9</td>
<td>-</td>
</tr>
<tr>
<td>Cascade-SRN (early fusion) (Wang et al., 2022)</td>
<td>89.8</td>
<td>-</td>
<td>67.1</td>
<td>-</td>
</tr>
<tr>
<td>PURE (Zhong and Chen, 2021)</td>
<td>89.7</td>
<td>69.0</td>
<td>65.6</td>
<td>88.8</td>
</tr>
<tr>
<td>PURE† (Zhong and Chen, 2021)</td>
<td>90.9</td>
<td>69.4</td>
<td>67.0</td>
<td>90.3</td>
</tr>
<tr>
<td>UniRE† (Wang et al., 2021)</td>
<td>90.2</td>
<td>-</td>
<td>66.0</td>
<td>89.5</td>
</tr>
<tr>
<td>TAGRIME w/ Cond. Priming</td>
<td>89.6</td>
<td>69.7</td>
<td>67.3†</td>
<td>89.0</td>
</tr>
<tr>
<td>TAGRIME w/ Cond. &amp; Rela. Priming</td>
<td>89.6</td>
<td><strong>70.4</strong></td>
<td><strong>68.1</strong>†</td>
<td>89.0</td>
</tr>
</tbody>
</table>

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. †indicates significant improvements per McNemar’s testing when compared to our reproduced JointBERT.

<table>
<thead>
<tr>
<th>Model</th>
<th>MTOP (en)</th>
<th></th>
<th>MTOP (es)</th>
<th></th>
<th>MTOP (fr)</th>
<th></th>
<th>MTOP (de)</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Intent Slot-I Slot-C</td>
<td>Intent Slot-I Slot-C</td>
<td>Intent Slot-I Slot-C</td>
<td>Intent Slot-I Slot-C</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>JointBERT (Li et al., 2021)</td>
<td>96.7</td>
<td>-</td>
<td>92.8</td>
<td>-</td>
<td>95.2</td>
<td>-</td>
<td>89.9</td>
<td>-</td>
</tr>
<tr>
<td>JointBERT (reproduced)</td>
<td>97.1</td>
<td><strong>94.2</strong></td>
<td>92.7</td>
<td><strong>95.6</strong></td>
<td>91.6</td>
<td><strong>89.5</strong></td>
<td>90.7</td>
<td><strong>87.7</strong></td>
</tr>
<tr>
<td>TAGRIME + Cond. Priming</td>
<td>97.1</td>
<td><strong>94.8</strong></td>
<td>93.4†</td>
<td>96.6</td>
<td>91.6</td>
<td>90.3</td>
<td>95.8</td>
<td><strong>90.6</strong></td>
</tr>
<tr>
<td>TAGRIME + Cond. &amp; Rela. Priming</td>
<td>97.1</td>
<td><strong>94.7</strong></td>
<td>93.5†</td>
<td>96.6</td>
<td><strong>91.8</strong></td>
<td><strong>90.7</strong>†</td>
<td>95.8</td>
<td><strong>90.6</strong></td>
</tr>
</tbody>
</table>

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. Appendix A.3 includes more details about the implementation.

**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 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 pre-
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, 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

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.
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 require 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


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).


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


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


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).


A Implementation Details

All the models are trained with NVIDIA RTX A6000 with 48GB memory. The experiments take around 6 hours for event extraction models, 20 hours for relation extraction models, and 4 hours for task-oriented semantic parsing experiments.

A.1 End-to-End Event Extraction

The followings are the training details for all baselines:
- **DyGIE++** (Wadden et al., 2019): we use the released training script\(^4\) 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\(^5\) 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 total 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.

A.2 End-to-End Relation Extraction

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 $2 \times 10^{-5}$ and the weight decay to $10^{-5}$. We consider the linear scheduler with a warm-up, where the warm-up epoch is 5. The total 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.

A.3 Task-Oriented Semantic Parsing

The followings are the training details for all baselines:
- **JointBERT** (Chen et al., 2019b): we use the released training script\(^6\) 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 a warm-up, where the warm-up epoch is 5. The total 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.

B Detailed Results

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

---

\(^4\)https://github.com/dwadden/dygiepp
\(^5\)http://blender.cs.illinois.edu/software/oneie/
\(^6\)https://github.com/monologg/JointBERT
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.

<table>
<thead>
<tr>
<th>Model</th>
<th>ACE05-E (en) Arg-I</th>
<th>ACE05-E (zh) Arg-I</th>
<th>ERE (en) Arg-I</th>
<th>ERE (es) Arg-I</th>
</tr>
</thead>
<tbody>
<tr>
<td>DyGIE++ (Wadden et al., 2019)</td>
<td>53.0 ± 48.8</td>
<td>63.0 ± 59.3</td>
<td>51.4 ± 48.0</td>
<td>49.2 ± 46.6</td>
</tr>
<tr>
<td>TANL (Paolini et al., 2021)</td>
<td>50.1 ± 47.6</td>
<td>- -</td>
<td>46.6 ± 43.2</td>
<td>- -</td>
</tr>
<tr>
<td>Text2Event (Lu et al., 2021)</td>
<td>- -</td>
<td>53.8 ± 48.3</td>
<td>- -</td>
<td>- -</td>
</tr>
<tr>
<td>OneIE’ (Lin et al., 2020)</td>
<td>59.2 ± 56.8</td>
<td>63.4 ± 60.5</td>
<td>50.1 ± 46.5</td>
<td>54.5 ± 52.2</td>
</tr>
<tr>
<td>DEGREE (Hsu et al., 2022b)</td>
<td>- -</td>
<td>-</td>
<td>49.6 ± 43.2</td>
<td>- -</td>
</tr>
</tbody>
</table>

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. △ indicates the use of cross-sentence context information.

<table>
<thead>
<tr>
<th>Model</th>
<th>ACE04-R Rel</th>
<th>ACE04-R Rel+</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table-Sequence (Wang and Lu, 2020)</td>
<td>67.6</td>
<td>64.3</td>
</tr>
<tr>
<td>PFN (Yan et al., 2021)</td>
<td>-</td>
<td>66.8</td>
</tr>
<tr>
<td>Cascade-SRN (late fusion) (Wang et al., 2022)</td>
<td>-</td>
<td>65.9</td>
</tr>
<tr>
<td>Cascade-SRN (early fusion) (Wang et al., 2022)</td>
<td>-</td>
<td>67.1</td>
</tr>
<tr>
<td>PURE (Zhong and Chen, 2021)</td>
<td>69.0</td>
<td>65.6</td>
</tr>
<tr>
<td>PURE’ (Zhong and Chen, 2021)</td>
<td>69.4</td>
<td>67.0</td>
</tr>
<tr>
<td>UniRE’ (Wang et al., 2021)</td>
<td>- -</td>
<td>63.0</td>
</tr>
</tbody>
</table>

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.