PromptASTE: Prompting a Dataset from Pre-trained Language Models for Unsupervised Aspect Sentiment Triplet Extraction

Anonymous EMNLP submission

Abstract

Aspect sentiment triplet extraction (ASTE) is a sentiment analysis task that aims to extract views' sentiment polarity, expression, and target (aspect). This paper proposes the first unsupervised method for aspect sentiment triplet extraction. Based on the previous discovery of the pre-trained language model (PLM)'s awareness of sentiment, we further leverage the masked language model (MLM) to prompt an ASTE dataset with automatically annotated labels. Our method, PromptASTE, fills in a series of prompts to generate a dataset for related aspects and views. The dataset is then used to train an ASTE model for prediction. Training on PromptASTE results in models with an outstanding capability in discerning sentiment polarities and targeted aspects. Our model sets the first and strong baseline on unsupervised ASTE.

1 Introduction

Aspect sentiment triplet extraction (ASTE) is a type of sentiment analysis task. Compared to conventional sentiment analysis that classifies the sentence-level sentiment polarity, ASTE is interested in aspect-based sentiment and extracts the expression (view) and target (aspect) of sentiments, more than just the polarity.

Some instances for ASTE are shown in Figure 1, the view and aspect are represented by spans. Paired spans are labeled as the sentiment polarity of the view on its targeted aspect. While many previous works have been done for the supervised ASTE system, unsupervised ASTE remains a blank. As sentiment is a universal and cross-language phenomenon, unsupervised ASTE is appealing to reduce the burden for annotation, especially for low-source language with a limited number of skilled annotators.

However, unsupervised ASTE is challenging as ASTE data are structured in a complex form. The unsupervised system faces several essential problems for relationship understanding. **a) Polarity** How the model understands the sentiment polarity with no annotated knowledge? **b) Relationship** How the model learns paired feature that does not exist in sequential natural language with no annotation for relationships? **c) Boundary** How the model determines the span boundaries annotated by human when testing?

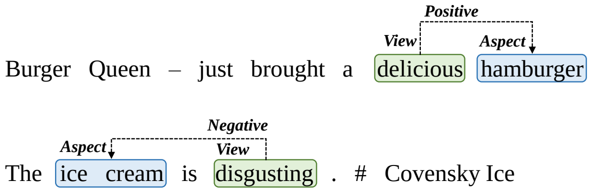

Positive
View
Aspect
Burger Queen – just brought a delicious hamburger
Negative
Aspect
View
The ice cream is disgusting . # Covensky Ice

Figure 1: Instances for the ASTE task.

The challenges above hinder the application of conventional unsupervised methods, like clustering. Moreover, clustering requires collecting unannotated data for unsupervised training, which is still unfriendly for low-source languages. We aim to step even further towards a method that is free from any ASTE-related data, no matter annotated or unannotated ones.

Thus, we cast our attention to pre-train language models (PLMs) (Radford et al., 2018; Devlin et al., 2019; Liu et al., 2019; Yang et al., 2019), which are competitive zero-shot learners (Radford et al., 2018) with strong scalability. PLMs, like Roberta (Liu et al., 2019), are trained on upstream masked language model (MLM) tasks that require the language model to fill in masked words in context. Recent studies have shown that pre-training endows PLMs with sentiment awareness to solve conventional sentiment analysis problems, suggesting the PLM is an admirable choice for unsupervised ASTE. By utilizing the MLM task, we fill in prompts to create an ASTE dataset from PLMs. A prompt combination is used to sample **kernel**

spans, which are spans consisting of aspect sentiment triplets, from PLMs.

The annotating system comprises three prompts for domain specification, aspect generation, and view generation. We also propose a contrastive prompt to prompt better sentiment expressions by contrasting positive and negative expressions. Based on the kernel span, PLMs are again used to supplement the contextual background via mask filling. The supplemented data finally form the PromptASTE dataset.

After the dataset is created, PromptASTE is used to train ASTE models following a supervised scenario. Spans and their relationships are annotated in graphs to train a parser for graphic pattern capturing. We test the trained parser on several ASTE datasets and compare the results with supervised results. Our method shows competitive performance on unsupervised ASTE and sets the first and strong baseline.

The contributions from our work are summarized as follows:

• We propose the first unsupervised method for ASTE and set a strong baseline for the task.

• We verify the plausibility of prompting a dataset for a task with a complex data structure.

• We implement a novel contrastive prompting procedure to generate sentiment expressions better.

2 Background and Related Work

Triplets in ASTE are formalized in (V, A, P) where V, A, P refer to view (expression) span, aspect (target) span, and sentiment polarity respectively. ASTE models are trained to determine the boundary of spans and label the polarity held by the view towards the aspect.

Since the annotation of a variety of ASTE datasets (Peng et al., 2020; Xu et al., 2020) based on aspect based sentiment analysis (ABSA) data (Pontiki et al., 2014, 2015, 2016), many supervised methods have been proposed for ASTE. (Peng et al., 2020) tests a wide range of previous triplet extracting method on ASTE and propose a new tagging model to set the first supervised baseline. (Xu et al., 2020) incorporates position information and CRF inference into the tagging system to boost the performance. (Wu et al., 2020) formalizes ASTE in a grid tagging scheme. Though supervised ASTE has been under heated discussion since the task's proposal, so far no attention has been cast to solve ASTE with no supervision.

However, unsupervised ASTE is a fairly challenging task. Besides its complex structured nature, the difficulty also comes from the incapability of existing unsupervised system to build a complete pipeline, from span extraction to relationship labeling. For unsupervised relation extraction, current models have only limited capability to label the relationships between paired already extracted spans (Tran et al., 2020; Yuan and Eldardiry, 2021). These methods use conventional unsupervised method like clustering to assign closely distributed span pairs the same labels. Thus, the prerequisite of annotated spans makes these unsupervised methods unfriendly to the real zero-shot learning.

Thus, we abandon the conventional unsupervised methods and turn towards leveraging PLMs, which are powerful zero-shot learners via training on super-large corpora. The long training procedure endows PLMs with the understanding of semantic relationships between tokens, which makes the PLM a desirable tool for unsupervised downstream tasks. Also, mask filling on prompts has been verified to be a power way to extract commonsense knowledge (Petroni et al., 2019), relationship understanding (Goswami et al., 2020), and sentiment awareness (Wu et al., 2019) of the PLM. Our work further leverages the endowed sentiment awareness in PLMs to build a complete unsupervised pipeline for ASTE.

3 Prompting ASTE Dataset

3.1 The Pipeline

We first give a rough description of our method and how it deals with the challenges in unsupervised ASTE before introducing the specific implementation. The pipeline comprises two main procedures: kernel span generation and context supplement.

Kernel span refers to the span that consists of the aspect sentiment triplet. To obtain kernel spans, our prompt involves masked view spans (v-mask) and masked aspect spans (a-mask). V-masks and a-masks are both common mask tokens used in the upstream MLM pre-training, and their only difference is representing views or aspects. The PLM fills the masked spans, and the kernel span is seized from the span for context supplement.

Polarity We add hints for polarity to the prompt in order to generate view expressions with the corresponding sentiment polarity.

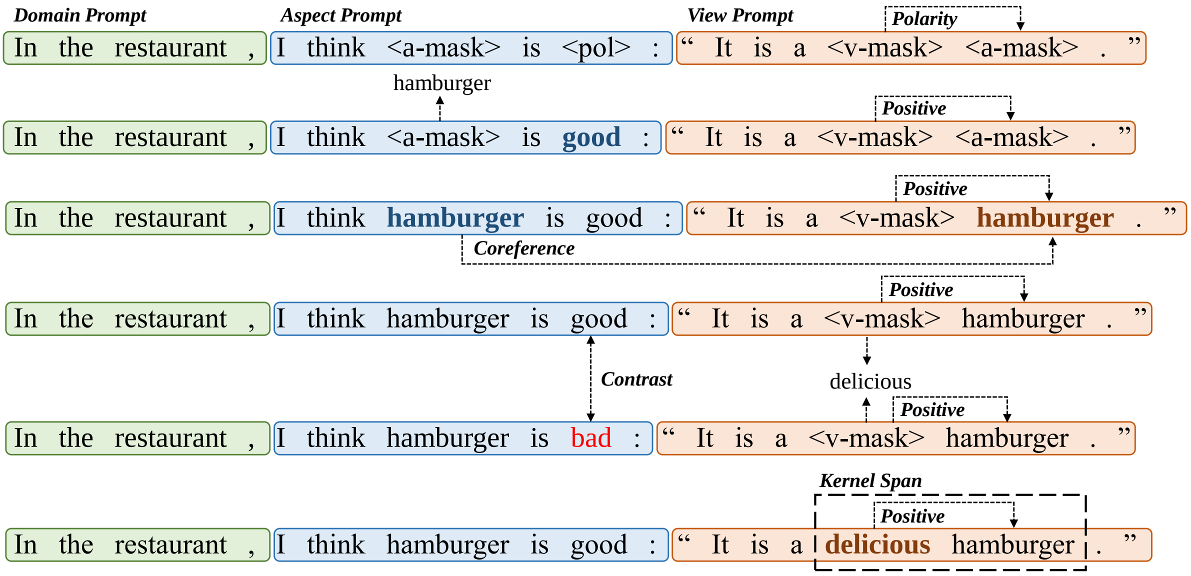

Domain Prompt
Aspect Prompt
View Prompt
Polarity
In the restaurant , I think <a-mask> is <pol> : " It is a <v-mask> <a-mask> . "
hamburger
Positive
In the restaurant , I think <a-mask> is good : " It is a <v-mask> <a-mask> . "
Positive
In the restaurant , I think hamburger is good : " It is a <v-mask> hamburger . "
Coreference
Positive
In the restaurant , I think hamburger is good : " It is a <v-mask> hamburger . "
Contrast
delicious
Positive
In the restaurant , I think hamburger is bad : " It is a <v-mask> hamburger . "
Kernel Span
Positive
In the restaurant , I think hamburger is good : " It is a delicious hamburger . "

Figure 2: Prompting steps for the generation of PromptASTE.

Relationship The relationships are pre-defined between views and aspects in the prompt.

Boundary Words near the span boundaries help control the generated span to have boundaries as pre-defined in the prompt.

Based on the kernel spans, we again use the PLM to supplement the contextual background for the sentiment via mask filling. The supplemented results are the final PromptASTE dataset.

3.2 Domain Prefix Prompt

The domain prefix prompt is used to specify the domain for kernel span generation. As in the green frame in Figure 2, the domain prefix prompt determines the contextual environment for the prompting generation. As the testing datasets are in different domains, the domain prefix prompt will help generate more relevant training data to improve the performance of trained models.

3.3 Aspect Prompt

The aspect prompt is the blue frame in Figure 2, which is responsible for polarity selection and aspect generation. The prompt contains a-masks and a polarity token <*pol*> that provides hints for the later generation.

After the polarity of triplets in the kernel span is selected, the polarity token is substituted by a token with sentiment information. In the instances in Figure 2, the word *good* substitutes <*pos*> and indicates the positive sentiment in the kernel span.

Then we fill in the a-masks via a beam search. Notice that the masked aspect span might consist of multiple mask tokens.

$$\begin{aligned}
&X = [x_1, \cdots, x_{i-1}, \textit{<mask>}, \cdots, \textit{<mask>}, x_{j+1}, \cdots, x_n] \\
&p(x_i, \cdots, x_j|X) = \prod_{t=i}^{j} p(x_t|X, x_i, \cdots, x_{t-1}) \\
&p(x_t|X, x_i, \cdots, x_{t-1}) = \text{softmax}(R_t/T) \\
&R = \text{PLM}(x_t|X, x_i, \cdots, x_{t-1})
\end{aligned} \quad (205)$$

where T refers to the temperature for sampling. $R \in \mathbb{R}^{n\times o}$ is the output representation from the PLM, and o refers to the dictionary size. We summarize the beam searching procedure as Beam($\cdot$). After we get the existing probability of each beam, we sample an aspect span following the predicted distribution.

3.4 Contrastive View Prompt

After generating the aspect span, we also fill in the coreferenced masked aspect span in the view prompt. Then we introduced our contrastive generation for view span.

For the prompt in this step X^{self}, we shift the word in the position of polarity token to create an opposite prompt X^{oppo}. We first use X^{self} to sample k view span beams by prompting and then calculate the probability distribution of the view span in X^{oppo}.

$$P^{self} = \text{Beam}(X^{self}), P^{oppo} = \text{Beam}(X^{oppo}) \quad (224)$$

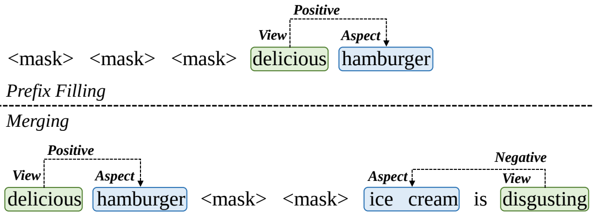

Positive
View
Aspect
<mask> <mask> <mask> delicious hamburger
Prefix Filling
Merging
Positive
View
Aspect
Negative
Aspect
View
delicious hamburger <mask> <mask> ice cream is disgusting

Figure 3: Supplement procedures that transform kernels into training data.

Finally, the log probability of P^{self} is subtracted by the weighted log probability of P^{self} and passed through a softmax function for the contrastive distribution.

$$P^{contrast} = \text{softmax}(\log(P^{self}) - w\log(P^{oppo}))$$

The view span is likely sampled following the predicted distribution as the aspect span.

After aspect and view spans are completely filled, we seize the kernel span and build the triplets using pre-defined relationships.

3.5 Context Supplement

Based on the collected kernel spans, we supplement the contextual background for them by continuing to utilize mask filling. We use two supplement scenarios in our experiments: prefix filling and kernel merging as in Figure 3.

Prefix filling is to attach several mask tokens to the beginning of the sentence. Then the PLM fills in the masks following a greedy strategy.

Kernel merging is to merge multiple kernel spans together. We insert several mask tokens between two collected kernels and use the PLM to fill in the mask, still following the greedy strategy.

We avoid adding mask tokens after the kernel span since the generated contents are more likely to break the aspect boundary and generate data with low quality. As a result, we do not apply postfix filling for the context supplement.

4 ASTE Model

4.1 Graph Annotation

We formalized the collected data as parsing graphs to train the ASTE model. We attach a question as the prefix prompt to each sentence, like in the dataset prompting step.

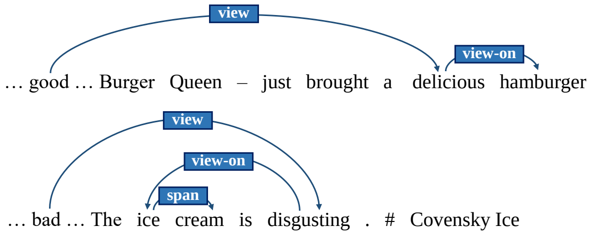

view
view-on
… good … Burger Queen – just brought a delicious hamburger
view
view-on
span
… bad … The ice cream is disgusting . # Covensky Ice

Figure 4: Transformed parsing graphs from ASTE instances.

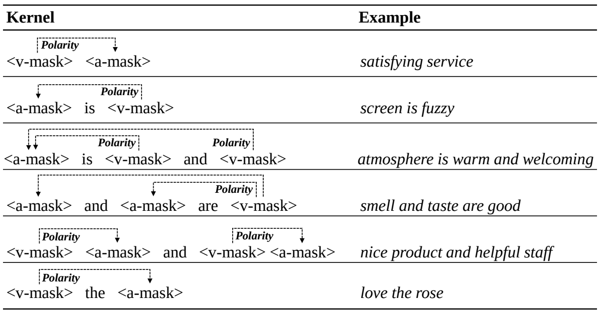
Is this comment ***good***, ***bad*** *or* ***common****? [SEP]*

For each triplet (V, A, P), we first build an edge from a sentiment token in the prefix prompt to the syntactic headword of the view span. We select the word with minimal depth in the syntactic dependency tree as the headword. The connected sentiment word indicates the polarity of view. Then, an edge with the *view-on* label is built from the headword of the view span to the headword of the aspect span, indicating the relationship between spans. Finally, for spans with more than one token, edges are built from the headword to the boundaries of the span. We show some transformed instances in Figure 4.

4.2 The Parser

We describe the training procedure of our parser in this section. For an input sentence, we concatenate the pre-trained word embedding and representation from the PLM to build the initial representation. Then we pass the representation through a multi-layer bidirectional long short term memory (BiLSTM) network (Hochreiter and Schmidhuber, 1997) for contextualization. The contextualized representations are then fed into four MLPs to get the head and dependent representations for edges $X^{head,edge}$, $X^{dep,edge}$ and labels $X^{head,label}$, $X^{dep,label}$.

The representations above are in shape $\mathbb{R}^{b\times m\times d}$ where b, m, d respectively represent the batch size, the sentence length, and the hidden dimension. To produce edge and label scores, we pass the representations through two scorers with second-order CRF inference (Wang et al., 2019). Instead of conventional biaffine (Dozat and Manning, 2017) and triaffine (Wang et al., 2019) scorers, we use the AOI scorer (Anonymous, 2021), a newly-proposed dot product scorer with global attention. The spe-



Kernel	Example
<v-mask> <a-mask>	*satisfying service*
<a-mask> is <v-mask>	*screen is fuzzy*
<a-mask> is <v-mask> and <v-mask>	*atmosphere is warm and welcoming*
<a-mask> and <a-mask> are <v-mask>	*smell and taste are good*
<v-mask> <a-mask> and <v-mask> <a-mask>	*nice product and helpful staff*
<v-mask> the <a-mask>	*love the rose*

Figure 5: Kernel spans used in our experiments.

cific scoring process is omitted here and can be found in Appendix A.

$$S^{edge} = \text{Scorer}(X^{head,edge}, X^{dep,edge})$$
$$S^{label} = \text{Scorer}(X^{head,label}, X^{dep,label})$$

After getting the scores $S^{edge} \in \mathbb{R}^{b\times m\times m}$, $S^{label} \in \mathbb{R}^{b\times c\times m\times m}$, we calculate the training loss by the cross entropy function. Here c refers to the number of label classes.

$$\mathcal{L}_{edge} = \sum_{i,j} \text{CrossEntropy}(S^{edge}_{i,j}, S^{edge,gold}_{i,j})$$
$$\mathcal{L}_{label} = \sum_{i,j,S^{edge,gold}_{i,j}=1} \text{CrossEntropy}(S^{label}_{i,j}, S^{label,gold}_{i,j})$$
$$\mathcal{L} = (1-\lambda)\mathcal{L}_{edge} + \lambda\mathcal{L}_{label}$$

5 Experiment

5.1 Testing Data and Metric

We use the ASTE datasets annotated in (Xu et al., 2020) for testing. The datasets include three restaurant review datasets and a laptop review dataset. To compare with previous supervised methods, we use the test datasets for evaluation. Besides, we also create a subset without boundary determination and neutral views to test the model's understanding of relationship and polarity. We first drop all triplets with neutral sentiment polarity. Then, we remove triplets that consist of spans in length > 1.

For evaluation, we use the F1 score that considers the exact matching of triplets as applied for previous supervised ASTE models. A triplet matches the golden triplet only when their views, aspects, and sentiment polarities are all matched.

5.2 Dataset Configuration

To build the PromptASTE dataset, we design six kernel spans as shown in Figure 5. The whole prompts for kernel construction are shown in Appendix B. Considering the domain variation in the testing dataset, we create two PromptASTE datasets with two different domain prefix prompts as follows.

In the restaurant, ...
For the laptop, ...

The contrastive prompting for neutral view span is a little different from positive and negative view. The neutral sentiment does not have a semantically opposite sentiment. Thus, we set both the positive and negative sentiment as the opposite to eliminate the view's polarity. The formula for contrastive generation is rewritten for the neutral view as follows.

$$P^{contrast} = \text{softmax}(\log(P^{self}) - \frac{w}{2}\log(P^{pos})) - \frac{w}{2}\log(P^{neg}))$$

For the generation, we use *Roberta-large* as the PLM. The beam size is set to 256 to cover a wide range of candidates. Tokens *good*, *bad*, and *average* are used to substitute the polarity token to indicate positive, negative and neutral sentiment polarities. We set temperature T to 1.0 for aspect span generation and 2.5 for context supplement. The temperature for view span generation varies from kernels to kernels to balance the generation's diversity and correctness. The specific setup for these temperatures is included in Appendix C. The weight w for contrastive prompting is 0.6. The max length of mask token series for context supplement is 6.

5.3 Model Configuration

Model We use *BERT-large-uncased* to produce the contextual representation. The pre-trained representation (projected to 600 hidden dimensions) is then concatenated by pre-trained word embedding from GloVe (Pennington et al., 2014) with 100 dimensions. A 2-layer BiLSTM with 400 hidden dimensions for each direction is applied for contextualization. The hidden size for edge and label representations are 600 and 300. A more detailed configuration for scorers and second-order CRF inference is in Appendix C.

Training The hyperparameter λ for loss weight balancing is 0.1. We apply the Adam optimizer (Kingma and Ba, 2015) for parameter updating. The learning rate is set to 3×10^{-4} initially, with a 3×10^{-9} weight decay.

Method	14res			14lap			15res			16res		
	P.	R.	F1	P.	R.	F1	P.	R.	F1	P.	R.	F1
(supervised)												
CMLA+	39.18	47.13	42.79	30.09	36.92	33.16	34.56	39.84	37.01	41.34	42.10	41.72
RINANTE+	31.42	39.38	34.95	21.71	18.66	20.07	29.88	30.06	29.97	25.68	22.30	23.87
Li-unified-R	41.04	67.35	51.00	40.56	44.28	42.34	44.72	51.39	47.82	37.33	54.51	44.31
(Peng et al., 2020)	43.24	63.66	51.46	37.38	50.38	42.87	48.07	57.51	52.32	46.96	64.24	54.21
OTE-MTL	63.07	58.25	60.56	54.26	41.07	46.75	60.88	42.68	50.18	65.65	54.28	59.42
JET^{t}	63.44	54.12	58.41	53.53	43.28	47.86	68.20	42.89	52.66	65.28	51.95	57.85
JET^{o}	70.56	55.94	62.40	55.39	47.33	51.04	64.45	51.96	57.53	70.42	58.37	63.83
GTS	71.76	59.09	64.81	57.12	53.42	55.21	54.71	55.05	54.88	65.89	66.27	66.08
(Huang et al., 2021)	63.59	73.44	68.16	57.84	59.33	58.58	54.53	63.30	58.59	63.57	71.98	67.52
(unsupervised)												
PromptASTE (res)	**68.12**	32.54	44.05	38.46	**19.89**	26.22	**55.97**	**33.88**	**42.21**	63.09	38.99	48.19
PromptASTE (lap)	52.95	31.24	39.30	**51.49**	18.81	27.55	44.00	29.55	35.35	55.71	38.01	45.19
PromptASTE (res + lap)	64.04	**34.73**	**45.03**	47.81	19.71	**27.91**	54.05	33.06	41.03	**64.72**	**41.13**	**50.30**

Table 1: Main results from our experiments on PromptASTE

Method	14res			14lap			15res			16res		
	P.	R.	F1	P.	R.	F1	P.	R.	F1	P.	R.	F1
Supervised	86.07	78.54	82.14	74.88	71.03	72.90	75.83	71.43	73.56	80.60	78.83	79.70
PromptASTE (res)	**76.07**	**51.67**	**61.54**	55.62	**43.93**	**49.09**	67.58	**54.91**	**60.59**	69.26	**64.96**	**67.04**
PromptASTE (lap)	61.98	49.58	55.09	51.10	43.46	46.97	57.78	46.43	51.49	63.64	56.20	59.69
PromptASTE (res + lap)	75.61	45.21	56.58	**61.59**	39.72	48.30	**74.13**	47.32	57.77	**72.46**	54.74	62.37

Table 2: Experiment results on the testing data in sampled subsets.

5.4 Experiment Result

The results from our experiments are presented in Tables 1 and 2. We report the highest results in the experiment. As no unsupervised baseline has been built before, we retrieve results from supervised baselines to evaluate our method's effectiveness.

Main result As in Table 1, we train and test parsers on PromptASTE datasets constructed in different domains. The experiment results verify the effectiveness of our method. PromptASTE achieves precision comparable to supervised results on all ASTE test datasets. For F1 score, PromptASTE outperforms supervised baselines like CMLA+ and RINANTE+. The recall is the weakness of PromptASTE as the limited patterns of kernel spans only endow the parser with partial recognition of aspect sentiment triplets. This weakness results from the trade-off with generality and simplicity and can be overcome by involving more patterns during prompting. But we want to propose a more general paradigm to prompt unsupervised datasets. Though there still exists a gap between PromptASTE and the highest supervised baseline, the outstanding performance establishes our method as a strong unsupervised baseline.

Domain analysis The main results also show how domain specification in dataset prompting affects the training result. Table 1 presents that the parser trained on restaurant PromptASTE dataset performs better on restaurant test datasets, and the phenomenon remains the same for the laptop domain. According to the comparison between parsers trained on datasets with different domain prefix prompts, parsers perform better on in-domain test datasets. Thus, the effect of domain specification for parser training is convinced. Moreover, the merging of PromptASTE datasets in different domains can result in better performance on some datasets. Thus, the combination of more domains during prompting might result in a further improvement in the performance.

Subset result Table 2 presents the results tested on the sampled datasets. PromptASTE achieves much higher results on the subset due to the difficulty of the unsupervised method to determine boundaries annotated by human. Free from boundary determination, the gap between PromptASTE and the supervised method is narrowed down in the subset, which better reflects the potential of PLMs for sentiment understanding.

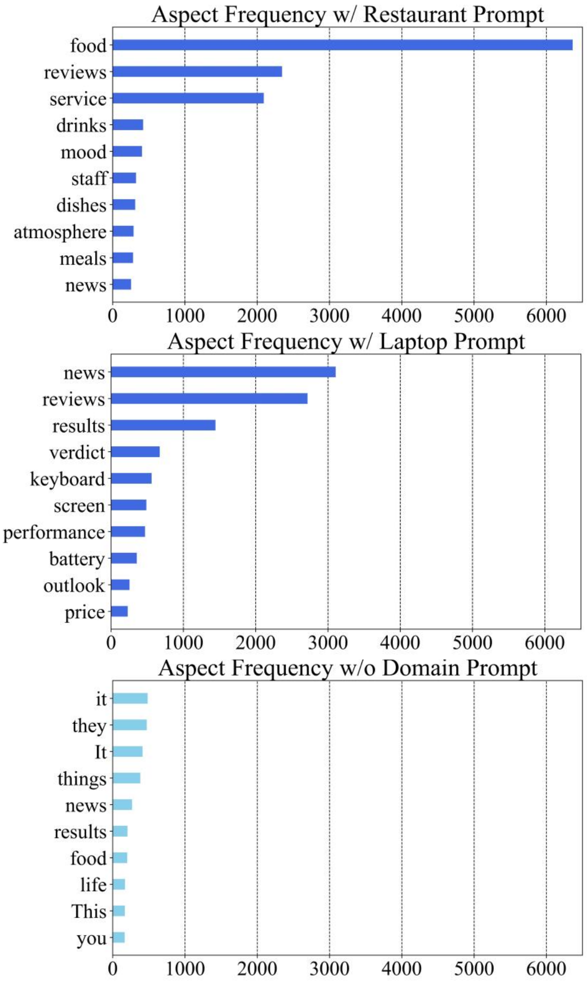


Figure 6: Sampled aspect (top 10) distribution with different domain prefix prompts.

6 Further Analysis

In this section, we conduct further experiments to analyze the components in our prompting pipeline. Our aim is to figure out how our design affects the generated results and trained models. Thus, we analyze our pipeline via ablation experiments and statistics. We also use case studies to discuss the capacity and limitation of our method.

How domain prefix affects aspect prompting? To analyze the contribution of the domain prefix prompt, we sample 20000 instances by using our pipeline with different prefix prompts. We count the frequency of the collected aspects and present the statistics in Figure 6.

Method	P.	R.	F1
PromptASTE	**76.07**	**51.67**	**61.54**
w/o Domain Prefix	58.60	45.42	51.17
w/o Contrastive Prompting	59.26	50.00	54.24
w/ Postfix Filling	72.31	48.96	58.39

Table 3: Ablation Study on PromptASTE. The subset of res14 is selected as the test dataset.

The results verify the capability of the prefix prompts for domain determination. With the prompt, our pipeline will generate more in-domain aspects, like *food*, *service*, *drinks* for the restaurant domain and *keyboard*, *screen*, *battery* for the laptop domain. The results above also verify the capability of our system to adapt to different domains by adjusting the prefix prompt.

In contrast, without the domain prefix prompt, the generated aspects are mostly some trivial pronouns that are not even considered by the ASTE task. We thus conclude that the prefix prompt activates the sampling for non-trivial data interested by the task.

Metrics from the ablation study in Table 3 also supports our conclusion. We reconstruct a dataset without the domain prefix prompt and train a parser on it. With the removal of domain prefix prompts, both precision and recall drop sharply. The phenomenon verifies the quality improvement on the generated data from the application of our domain prefix prompt.

How contrast affects view prompting? Contrastive prompting is a key component in our pipeline, which guarantees the polarity of generated views. To analyze how contrast affects view prompting, we respectively sample 10000 instances for positive and negative sentiment, with or without contrastive prompting. We depict the statistics of collected views in Figure 7, comparing between sampled results with or without contrast.

The comparison shows that the views from the contrastive prompt enjoy higher quality. First, the 10 most common views sampled by contrastive prompts are all in correct polarities, whereas the views sampled from conventional prompts include non-sentiment words, like + and *the*. Also, considering the 10 most common views, generation from contrastive prompt have 31.94% and 32.31% probabilities of falling in a correct view for positive and negative sentiments, respectively. In comparison, the probabilities for conventional prompts are only 18.55% and 15.96%.

The ablation result in Table 3 further support our conclusion. The removal of contrastive prompting leads to a dramatic drop in precision, recall, and F1 score.

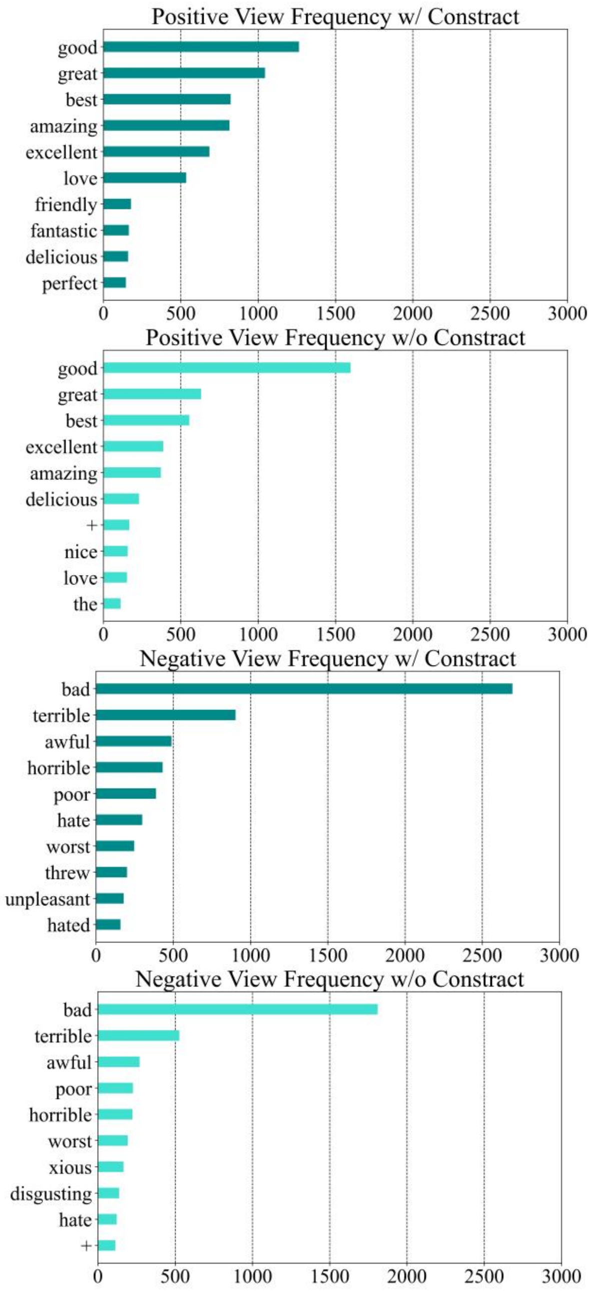


Figure 7: Sampled view (top 10) distribution with and without contrastive prompting.

How postfix filling affects training result? We test a pipeline with postfix filling. The performance drop in the ablation study suggests postfix filling is not a beneficial context supplement method.

What is current boundary of PromptASTE's capability in ASTE? We enumerate and analyze several cases in Figure 8 to answer the question.

In the first case, the instance pattern is covered by our prompting pipeline. The instance can be generated by the prompt via kernel merging between two defined kernel spans. As a result, the instance is easily solved by the parser trained with PromptASTE.

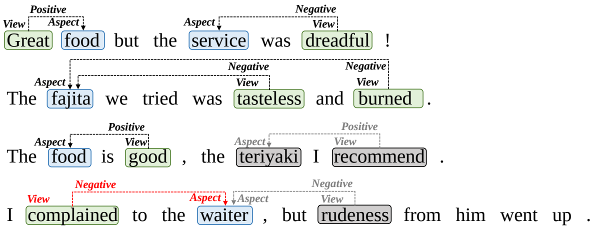

Positive
View
Aspect
Aspect
Negative
View
Great food but the service was dreadful !
Negative
Negative
Aspect
View
View
The fajita we tried was tasteless and burned .
Positive
Positive
Aspect
View
Aspect
View
The food is good , the teriyaki I recommend .
Negative
Negative
View
Aspect
Aspect
View
I complained to the waiter , but rudeness from him went up .

Figure 8: Case Study for the capability boundary of PromptASTE. Grey arrow: Missing triplet (negative false). Red arrow: Incorrect triplet (negative true).

The second case shows the scalability of PromptASTE as the pattern of the instance is not covered by prompting. The parser stays robust against the noise from the adjective component *we tried*. Thus, the triplets are successfully extracted from the sentence.

The limitation of PromptASTE is presented in the third case. While the parser correctly extracts the first triplet, the *recommend-teriyaki* relationship is ignored. As the relationship is in a casual pattern that is very different from our pre-defined ones, the parser fails to capture it. Incorporating this casual pattern into kernel spans might well solve the problem.

The last case includes inference based on coreference, a thorny problem for our parse trained on data with fixed patterns. The case also shows our method to suffer from shortcut learning (Geirhos et al., 2020). The word *complained* is directly recognized as a negative view on the word *waiter*, without understanding the semantic relationships between them. Solving these problems might require pre-trained models for a stronger inference capability.

From the cases, we conclude that our method has some basic understanding of ASTE and enjoys some scalability from the PLM. However, hyperlinguistic phenomena like coreference still remain the problem for us to solve in future studies.

7 Conclusion

We propose a novel method, PromptASTE, for ASTE, which is also the first unsupervised method. We utilize the PLM's understanding of sentiment and apply a series of prompts to construct a training dataset from the PLM. Various prompting mechanisms guarantee the quality of the generated dataset and trained parser to set a strong baseline for unsupervised ASTE.

References

Anonymous. 2021. Counting what deserves to be counted for graph parsing. In *ACL ARR 2021 November Blind Submission*. OpenReview.net.

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 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Timothy Dozat and Christopher D. Manning. 2017. Deep biaffine attention for neural dependency parsing. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard S. Zemel, Wieland Brendel, Matthias Bethge, and Felix A. Wichmann. 2020. Shortcut learning in deep neural networks. *Nat. Mach. Intell.*, 2(11):665–673.

Ankur Goswami, Akshata Bhat, Hadar Ohana, and Theodoros Rekatsinas. 2020. Unsupervised relation extraction from language models using constrained cloze completion. In *Findings of the Association for Computational Linguistics: EMNLP 2020, Online Event, 16-20 November 2020*, volume EMNLP 2020 of *Findings of ACL*, pages 1263–1276. Association for Computational Linguistics.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.*, 9(8):1735–1780.

Lianzhe Huang, Peiyi Wang, Sujian Li, Tianyu Liu, Xiaodong Zhang, Zhicong Cheng, Dawei Yin, and Houfeng Wang. 2021. First target and opinion then polarity: Enhancing target-opinion correlation for aspect sentiment triplet extraction. *CoRR*, abs/2102.08549.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

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

Haiyun Peng, Lu Xu, Lidong Bing, Fei Huang, Wei Lu, and Luo Si. 2020. Knowing what, how and why: A near complete solution for aspect-based sentiment analysis. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8600–8607. AAAI Press.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick S. H. Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander H. Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 2463–2473. Association for Computational Linguistics.

Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, Ion Androutsopoulos, Suresh Manandhar, Mohammad Al-Smadi, Mahmoud Al-Ayyoub, Yanyan Zhao, Bing Qin, Orphée De Clercq, Véronique Hoste, Marianna Apidianaki, Xavier Tannier, Natalia V. Loukachevitch, Evgeniy V. Kotelnikov, Núria Bel, Salud María Jiménez Zafra, and Gülsen Eryigit. 2016. Semeval-2016 task 5: Aspect based sentiment analysis. In *Proceedings of the 10th International Workshop on Semantic Evaluation, SemEval@NAACL-HLT 2016, San Diego, CA, USA, June 16-17, 2016*, pages 19–30. The Association for Computer Linguistics.

Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, Suresh Manandhar, and Ion Androutsopoulos. 2015. Semeval-2015 task 12: Aspect based sentiment analysis. In *Proceedings of the 9th International Workshop on Semantic Evaluation, SemEval@NAACL-HLT 2015, Denver, Colorado, USA, June 4-5, 2015*, pages 486–495. The Association for Computer Linguistics.

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014. Semeval-2014 task 4: Aspect based sentiment analysis. In *Proceedings of the 8th International Workshop on Semantic Evaluation, SemEval@COLING 2014, Dublin, Ireland, August 23-24, 2014*, pages 27–35. The Association for Computer Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2018. Language models are unsupervised multitask learners.

Thy Thy Tran, Phong Le, and Sophia Ananiadou. 2020. Revisiting unsupervised relation extraction. In *Pro-*

ceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020, pages 7498–7505. Association for Computational Linguistics.

Xinyu Wang, Jingxian Huang, and Kewei Tu. 2019. Second-order semantic dependency parsing with end-to-end neural networks. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 4609–4618. Association for Computational Linguistics.

Xing Wu, Tao Zhang, Liangjun Zang, Jizhong Han, and Songlin Hu. 2019. Mask and infill: Applying masked language model for sentiment transfer. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 5271–5277. ijcai.org.

Zhen Wu, Chengcan Ying, Fei Zhao, Zhifang Fan, Xinyu Dai, and Rui Xia. 2020. Grid tagging scheme for aspect-oriented fine-grained opinion extraction. *CoRR*, abs/2010.04640.

Lu Xu, Hao Li, Wei Lu, and Lidong Bing. 2020. Position-aware tagging for aspect sentiment triplet extraction. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 2339–2349. Association for Computational Linguistics.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 5754–5764.

Chenhan Yuan and Hoda Eldardiry. 2021. Unsupervised relation extraction: A variational autoencoder approach. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 1929–1938. Association for Computational Linguistics.

A Scoring Procedure

We elaborate on the scoring processing of our parser in this section. For the representations for edges $X^{head,edge}$, $X^{dep,edge}$ and labels $X^{head,label}$, $X^{dep,label}$. We calculate the first-order scores by the accumulative operation-based induction (AOI) scorer (Anonymous, 2021) and the second-order scores by TriAOI, which is implemented based on accumulative multi-head attention as in AOI.

A.1 First-order: AOI Scorer

We first describe the first-order scoring procedure. For the paired representations X^{dep} and X^{head}, two linear transformations are used to get representations for specific labels.

$$\begin{aligned}\hat{X}^{head} &= W^{head}X^{head} + b^{head}\\ \hat{X}^{dep} &= W^{dep}X^{dep} + b^{dep}\end{aligned}$$

where weights $W^{dep}, W^{head} \in \mathbb{R}^{d\times c\times d}$, and biases $b^{dep}, b^{head} \in \mathbb{R}^{c\times d}$. The transformed $\hat{X}^{head}$, $\hat{X}^{dep}$ are in the shape $\mathbb{R}^{b\times c\times d}$.

Based on the label-specific representations, AOI uses a label-wise dot product to get the self attention scores.

$$S^{SelfAttn} = \hat{X}^{head} \cdot \hat{X}^{dep}$$

Then the hidden dimensions of $\hat{X}^{head}$, $\hat{X}^{dep}$ are split into a attention heads with d' dimensions, where d' and a satisfy $a \times d' = d$. The split representations $\tilde{X}^{head}$, $\tilde{X}^{dep}$ are then concatenate with the sequentially average pooled results.

$$\begin{aligned}\tilde{X}^{head,glob} &= \tilde{X}^{head} \oplus \text{MeanPool}(X^{head})\\ \tilde{X}^{dep,glob} &= \tilde{X}^{dep} \oplus \text{MeanPool}(X^{dep})\end{aligned}$$

The last hidden dimensions of $\tilde{X}^{head,glob}$ and $\tilde{X}^{dep,glob}$ are then projected to 1 by linear layers. A softmax layer is applied for the second sequential dimension to score the attention on heads. The final accumulative attention score is the max pooling on different attention heads.

$$\begin{aligned}A^{head} &= \text{Softmax}(\text{MLP}_{head,attn}(\tilde{X}^{head}))\\ A^{dep} &= \text{Softmax}(\text{MLP}_{dep,attn}(\tilde{X}^{dep}))\\ \hat{A}^{head} &= \text{MaxPool}(A^{head}), \hat{A}^{dep} = \text{MaxPool}(A^{dep})\end{aligned}$$

The product between self attention and accumulative attention produces the final first-order scores.

$$S_{i,j} = A^{SelfAttn}_{i,j} \times (\hat{A}^{head}_i \times \hat{A}^{dep}_j \times N)$$

where i, j refer to the element's position in graphs and sequences. N is a modifier that controls the density of attention.

A.2 Second-order: TriAOI Scorer

For second-order CRF inference, we also involve scorers that produce scores Q in $\mathbb{R}^{n\times n\times n}$ for inference. TriAOI takes three binary representations $B^x, B^y, B^z \in \mathbb{R}^{b\times n\times d}$ as the input.

$$Q = \text{TriAOI}(B^x, B^y, B^z)$$

Like in AOI, Q is also the product between self attention scores and accumulative attention scores.

$$\begin{aligned}Q_{i,j,k} &= A^{SelfAttn}_{i,j,k} \times (\hat{A}^x_i \times \hat{A}^y_j \times \hat{A}^z_k \times N)\\ A^{SelfAttn}_{i,j,k} &= \sum_u^d B^x_{i,u} \times B^x_{j,u} \times B^z_{k,u}\end{aligned}$$

where accumulative attention scores $\hat{A}^x$, $\hat{A}^y$, $\hat{A}^z$ are scored as in AOI.

We project the contextualized representation to representations for head B^{head}, dependent B^{cop}, and middle B^{mid} and use TriAOIs for second-order scoring.

$$\begin{aligned}Q^{sib} &= \text{TriAOI}(B^{head}, B^{dep}, B^{dep})\\ Q^{cop} &= \text{TriAOI}(B^{head}, B^{dep}, B^{head})\\ Q^{grd} &= \text{TriAOI}(B^{head}, B^{mid}, B^{dep})\end{aligned}$$

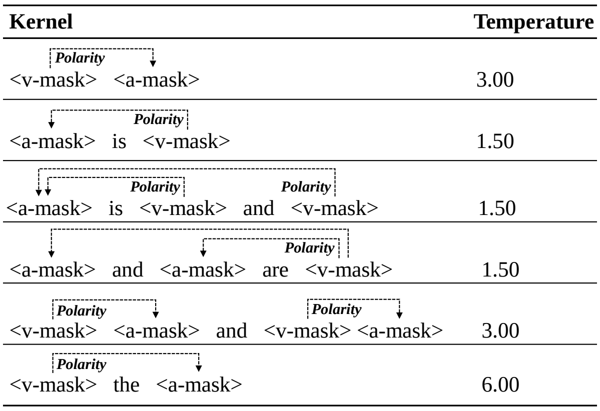
A.3 Mean Field Variational Inference

We follow the procedure in (Wang et al., 2019) for second-order CRF inference. For each iteration, we update the edge scores as follows.

$$\begin{aligned}\mathcal{G}^{(t-1)}_{i,j} = \sum_{k\neq i,j}\{&Q^{(t-1)}_{i,k}S^{(sib)}_{i\to j,i\to k} + Q^{(t-1)}_{k,j}S^{(cop)}_{i\to j,k\to j}\\ &+Q^{(t-1)}_{k,i}S^{(gp)}_{k\to i\to j} + Q^{(t-1)}_{j,k}S^{(gp)}_{i\to j\to k}\},\end{aligned}$$

$$Q^{(t)}_{i,j} = \begin{cases}\exp(S^{arc}_{i\to j} + \mathcal{G}^{(t-1)}_{i,j}), & \text{Arc } i \to j \text{ exist}\\ 1, & \text{Otherwise}\end{cases}$$

where t denotes the step for iteration.



Kernel	Temperature
<v-mask> <a-mask>	3.00
<a-mask> is <v-mask>	1.50
<a-mask> is <v-mask> and <v-mask>	1.50
<a-mask> and <a-mask> are <v-mask>	1.50
<v-mask> <a-mask> and <v-mask> <a-mask>	3.00
<v-mask> the <a-mask>	6.00

Figure 10: The configuration for temperature to generate view spans.

B Whole Prompt for Kernel Building

We present the whole prompts used in our experiments in Figure 9. Some special tokens are in the prompts. <*prefix*> refers to the domain prefix prompt. <*det*> refers to the determinative component. <*adv*> refers to the adverb component. <*be*> refers to words with the *be* lemma.

C Specific Configuration

C.1 Prompting Configuration

The temperature configuration for prompting is shown in Figure 10.

C.2 Parsing Configuration

The max epoch and patient are set to 200 and 20, respectively. The batch size is 5000. The dropout rates for BiLSTM, edge MLP, label MLP are 0.33, 0.25, 0.33. The representation from the PLM is the average of representations in the last 4 layers. The number of attention heads for AOI scorers is 4. To construct dependency trees, we use the parser provided by SpaCy[1].

For the second-order CRF inference, the number of binary representation's dimensions is 160, projected by MLPs with a 0.25 dropout rate. The number of attention heads for TriAOI scorers is also 4. The max number of iterations for second-order CRF inference is 3.

D Statistical Properties of Datasets

Prop.	14res	15res	16res	14lap
Sent. Num.	2.1k	1.1k	1.4k	1.5k
Sent. Len.	16.9	15.0	14.9	18.4
Span. Num.	6.8k	3.1k	4.0k	4.1k
Span. Len.	1.3	1.3	1.3	1.4
Rel. Num.	4.0k	1.7k	2.2k	2.4k

Table 4: Statistical properties of the triplet parsing datasets used in our experiments.

The statistical properties of the triplet parsing datasets in our experiments are presented in Table 4.

[1] https://spacy.io/

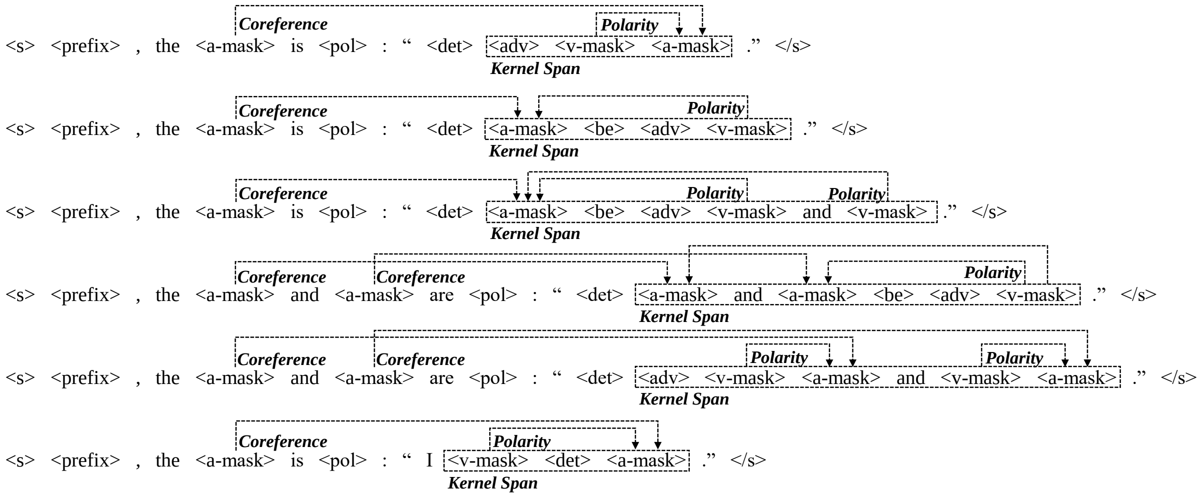

Coreference
Polarity
<s> <prefix> , the <a-mask> is <pol> : “ <det> <adv> <v-mask> <a-mask> .” </s>
Kernel Span
Coreference
Polarity
<s> <prefix> , the <a-mask> is <pol> : “ <det> <a-mask> <be> <adv> <v-mask> .” </s>
Kernel Span
Coreference
Polarity
Polarity
<s> <prefix> , the <a-mask> is <pol> : “ <det> <a-mask> <be> <adv> <v-mask> and <v-mask> .” </s>
Kernel Span
Coreference
Coreference
Polarity
<s> <prefix> , the <a-mask> and <a-mask> are <pol> : “ <det> <a-mask> and <a-mask> <be> <adv> <v-mask> .” </s>
Kernel Span
Coreference
Coreference
Polarity
Polarity
<s> <prefix> , the <a-mask> and <a-mask> are <pol> : “ <det> <adv> <v-mask> <a-mask> and <v-mask> <a-mask> .” </s>
Kernel Span
Coreference
Polarity
<s> <prefix> , the <a-mask> is <pol> : “ I <v-mask> <det> <a-mask> .” </s>
Kernel Span

Figure 9: The whole format of prompts used in our experiments.