

Revisiting Structured Sentiment Analysis as Latent Dependency Graph Parsing

Chengjie Zhou¹, Bobo Li¹, Hao Fei², Fei Li¹, Chong Teng¹, Donghong Ji^{1*}

¹ Key Laboratory of Aerospace Information Security and Trusted Computing, Ministry of Education, School of Cyber Science and Engineering, Wuhan University, Wuhan, China

² National University of Singapore, Singapore

{zhoucj, boboli, lifei_csnlp, tengchong, dhji}@whu.edu.cn

Abstract

Structured Sentiment Analysis (SSA) was cast as a problem of bi-lexical dependency graph parsing by prior studies. Multiple formulations have been proposed to construct the graph, which share several intrinsic drawbacks: (1) The internal structures of spans are neglected, thus only the boundary tokens of spans are used for relation prediction and span recognition, thus hindering the model’s expressiveness; (2) Long spans occupy a significant proportion in the SSA datasets, which further exacerbates the problem of internal structure neglect. In this paper, we treat the SSA task as a dependency parsing task on partially-observed dependency trees, regarding flat spans without determined tree annotations as latent subtrees to consider internal structures of spans. We propose a two-stage parsing method and leverage TreeCRFs with a novel constrained inside algorithm to model latent structures explicitly, which also takes advantages of joint scoring graph arcs and headed spans for global optimization and inference. Results of extensive experiments on five benchmark datasets reveal that our method performs significantly better than all previous bi-lexical methods, achieving new state-of-the-art.¹

1 Introduction

Structured Sentiment Analysis (SSA) aims to extract the complete opinion tuple from a sentence. As shown in Figure 1(a), the complete opinion tuple includes an opinion expression e with sentiment polarity p , an opinion holder h , and the corresponding target t . Given the complexity of detecting three items and classifying one, SSA presents more challenges than other related tasks, such as Opinion Mining (Katiyar and Cardie, 2016; Xia et al., 2021), ABSA (Aspect-based Sentiment Analysis) (Pontiki et al., 2014, 2016; Wang et al., 2016), TOWE

(Target-oriented Opinion Words Extraction) (Fan et al., 2019; Mao et al., 2021), ASTE (Aspect Sentiment Triplet Extraction) (Peng et al., 2020; Mao et al., 2021; Zhai et al., 2022; Fei et al., 2022a; Li et al., 2023, 2024), etc.

Recent works of SSA mainly cast it as a problem of bi-lexical dependency graph parsing and propose multiple formulations: (1) Barnes et al. (2021) proposed formulations namely head-first/head-final as illustrated in Figure 1(b). Their method cannot resolve the problem because head-first/head-final treats the first/final word as the head of the span and strictly restricts any word inside the span directly head to span head, which decreases the height of the converted trees to 2 and excludes the latent structures completely. (2) Another label strategy was proposed by Shi et al. (2022), which simplifies the label set to only arcs linking spans boundaries, as shown in Figure 1(c). Despite the special label for discontinuous span decoding, Zhai et al. (2023) utilize the same label set. Without distinct formulation about inside words, they attempt to utilize the powerful neural models like Graph Attention Network (Velickovic et al., 2018) or Axial-based Attention Network (Huang et al., 2019; Wang et al., 2020) to implicitly encode the inside structure information, which is found lagging behind large with explicitly modeling with graph-based parsing methods (Wang and Tu, 2020; Fonseca and Martins, 2020; Zhang et al., 2020; Yang and Tu, 2022a). It is evident that previous work does not address the key challenge focused on the prediction of boundary words (First/Final/Both) of spans, and neglect the words and structures inside spans, which hinders the model expressiveness seriously.

Should we neglect the structure inside spans as previous works have done? Table 1 list the statistics of span length and the max length of spans in the benchmark datasets respectively. We present a real example to illustrate our point, focusing on the expression “conceded” and the target “US President

*Corresponding author

¹Our code is publicly available at <https://github.com/JYoen/latentssa>.

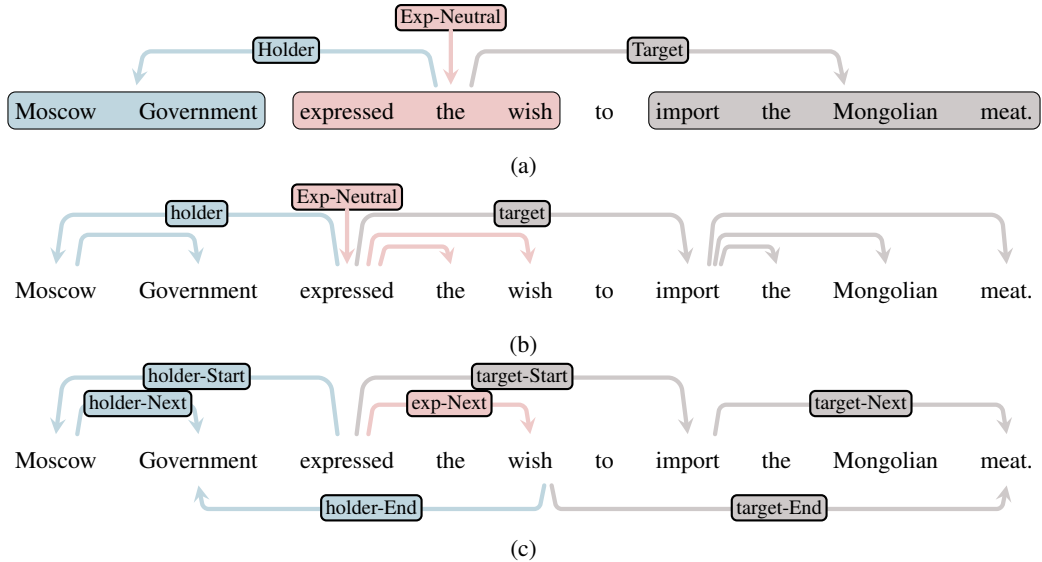


Figure 1: (a) An example of original structure sentiment analysis. (b) The head-first parsing graph proposed by Barnes et al. (2021). (c) The label strategy proposed by Shi et al. (2022). The label formulation proposed by Zhai et al. (2023) is similar to this.

Dataset	Span Length ≥ 4			Max
	Holder	Target	Exp.	
NoReC _{Fine}	1.1%	19.2%	56.8%	40
MultiB _{CA}	2.6%	18.4%	21.4%	19
MultiB _{EU}	1.1%	2.7%	15.3%	10
MPQA	19.9%	51.1%	14.5%	56
DS _{Unis}	1.3%	0.8%	13.7%	9

Table 1: Statistics of the proportion of span 1) with length greater than or equal to 4 and 2) the max length of span (both in tokens) for SSA datasets. We highlight the number that reveals that the long span is ubiquitous and the extraction of them seems to be the bottleneck of SSA problem.

... last year” for brevity.

Tang *conceded* there had been “twists and turns” following *US President George W. Bush’s accession to the US presidency in early last year*.

From this example, it is evident that the target span is considerably long, making it challenging for the boundaries (“US” and “year”) to effectively represent the entire span. Conversely, the internal word “accession” provides significant clues for identifying it as the target of the expression “conceded”. Furthermore, the other words within the target span act as modifiers, aiding in the detection of the span’s boundaries. Based on these observations, the necessity of addressing the internal structure of

spans in the SSA task is clear. It remains a significant challenge to develop an effective and unified method that can handle such structures.

To address this issue, we propose a novel approach to structured sentiment analysis: treating flat spans as latent subtrees. This perspective considers expression-holder/target structures as partially-observed trees, where the exact subtrees for each span are yet to be determined. We employ TreeCRF (Eisner, 1997; Eisner and Satta, 1999) to model these partially-observed trees. This involves enumerating all possible arcs and spans in a latent tree form of a flat span and calculating their probabilities using a constrained inside algorithm. The scores for all possible arcs and spans are determined using biaffine or triaffine attention methods (Dozat and Manning, 2017; Zhang et al., 2020), commonly applied in Dependency Parsing. During the decoding stage, we parse the highest-scoring dependency tree and reconstruct the SSA structure from it. Span boundaries are accurately inferred from the descendant words of a head word. Additionally, using the labels from the dependency tree, we can globally predict span-span relationships. This method not only explicitly accounts for internal span structures but also maintains the end-to-end nature characteristic of previous work.

We conduct extensive experiments on five benchmarks, including NoReC_{Fine} (Øvrelid et al., 2020), MultiB_{EU}, MultiB_{CA} (Barnes et al., 2018), MPQA

(Wiebe et al., 2005) and DS_{Unis} (Toprak et al., 2010). The results affirm that our model achieves new state-of-the-art in performance for SSA task.

Our contributions are summarized as follows:

- We cast SSA as a novel latent trees formulation to address the neglecting of span structures in prior work. Concretely, we treat flat spans as latent trees and marginalize the explicit structure via a novel constraint inside algorithm.
- We propose an effective two-stage parsing method to well collaborate with our latent tree formulation, which employs dependency parsing with high-order scoring and global optimization, modeling sentiment structures explicitly.
- The experimental results show that our model has achieved the SOTA performance in five datasets for structured sentiment analysis, especially in terms of long spans boundary detection and relation prediction.

2 Related Work

As a key topic in the Sentiment Analysis community (Fei et al., 2023a; Wu et al., 2021), Structured Sentiment Analysis (SSA) encompasses several sub-tasks, each targeting a specific component of the goal tuple (holder, target, expression, polarity). SSA also involves closely with structure predictions (Fei et al., 2022c) and relevant tasks (Fei et al., 2023b; Wu et al., 2023a,b).

OM Opinion Mining primarily aims to extract the (h, t, e) tuple. Existing OM studies generally adopt one of two approaches: 1) BIO-based approach, which views OM as a sequence labeling task (Katiyar and Cardie, 2016); and 2) span-based approach, which jointly predicts all span pairs and their interrelations (Xia et al., 2021). Additionally, Zhang et al. (2019); Wu et al. (2022) introduced a transition-based model for OM. However, these methodologies neglect the sentiment polarity classification sub-task.

ABSA Aspect-Based Sentiment Analysis is another important sentiment analysis task. Various methods have been proposed to address ABSA, including: 1) Pipeline (Peng et al., 2020), which sequentially predicts spans and their relationships; 2) End-to-End (Chen and Qian, 2020), utilizing interactive information from each pair of sub-tasks; and 3) MRC (Mao et al., 2021; Zhai et al., 2022), which

employ a machine reading comprehension framework to extract triplets. However, these methodologies neglect the sentiment holder extraction sub-task.

SSA Barnes et al. (2021) propose the conversion named head-first/head-final and applied first-order parsing method. Shi et al. (2022) proposed a new label strategy and apply Graph Attention Network for aggregation on span boundaries for decoding, Samuel et al. (2022) apply Transformer to predict the graph directly from the text, Zhai et al. (2023) add new labels to model the boundaries of discontinuous spans and apply axial-attention encoder and table filling scheme to decode the relations. However, all these works neglect the internal structure of the spans and rely on powerful encoders to implicitly incorporate internal span structure information.

Different from previous work, we are the first to cast SSA task as partially-observed dependency tree and apply dependency parsing method to explicitly model the internal structures of spans. Owing to the structural similarities, leveraging NLP tasks such as parsing proves to be an effective strategy for structured prediction. Our research builds upon the successes observed in partially-observed tree reduction and parsing methods across various NLP tasks including named entity recognition (NER) (Yu et al., 2020), nested NER (Fu et al., 2021; Lou et al., 2022; Li et al., 2022; Fei et al., 2022b) and semantic role labeling (Zhang et al., 2022).

3 SSA as Latent Graph Parsing Scheme

In the formulation of SSA as Latent Dependency Graph, we treat each sentiment span as latent tree. To build the Dependency Graph, we deal with each expression span **separately** and assumes each of them corresponds to a single-root tree. In this tree, the sentiment head word of the expression span serves as the dependency root, with each subtree of the corresponding holder/target span attaching to it. Consequently, our proposed latent dependency parsing task can be divided into two sub-tasks: (1) expression extraction; (2) corresponding holder/target extraction. Note that both of these subtasks are solved by a consistent graph-based and headed-span-based parsing method, trained jointly and decoded step-by-step, which is named

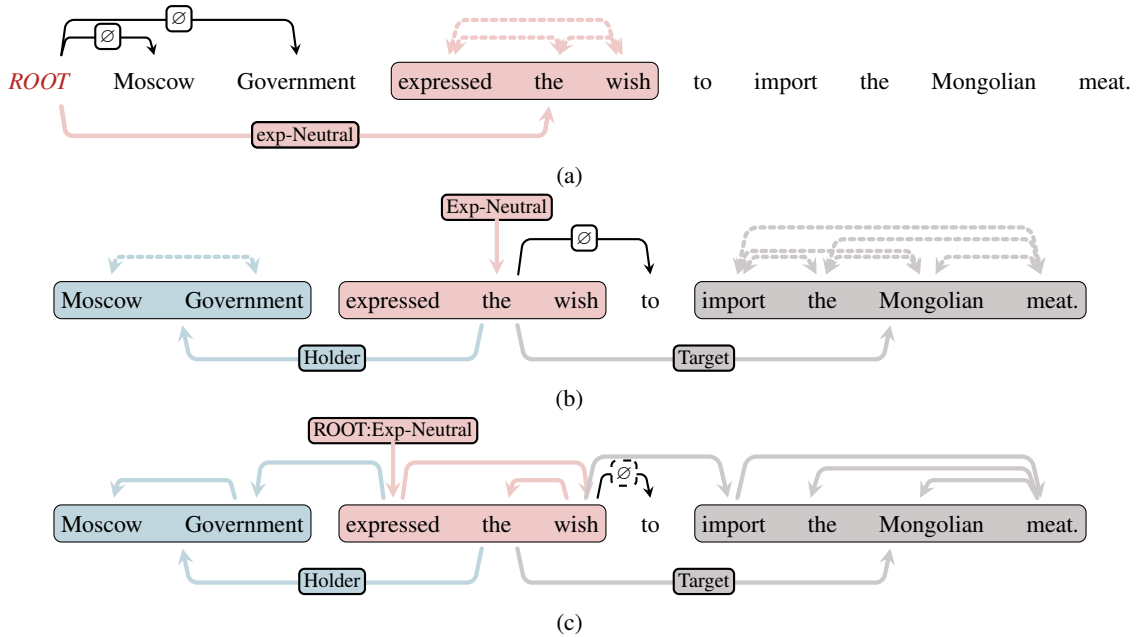


Figure 2: Illustration of our SSA as Latent Dependency Graph and Two-stage Parsing. (a) The conversion and training of Stage I-expression extraction: convert the ROOT-expression structure to a dependency tree with (dotted) latent annotations; non-expression spans are assigned a label of \emptyset for clarity. We omit the \emptyset edge from ROOT to “to import ...” for brevity. (b) The conversion and training of corresponding Stage-II-holder/target extraction: with the given expression, convert the expression-holder/target structure to a dependency tree with (dotted) latent annotations and irrelevant spans assigned with \emptyset as well. (c) The decoding and recovery of the converted latent dependency tree: Parsing the best dependency tree (above the sentence) and recovery the tree to SSA structure (below). We combine the separate two-stage decoding and recovery in one figure for brevity.

as **Two-stage Parsing**².

For the rest of this section, we show how our method : (1) converting SSA to latent dependency graph; (2) training to get span/tree structural representation; (3) decoding to find the best tree; (4) recovering resulting tree to sentiment tuple.

3.1 Conversion and Training on Latent Tree

Formally, given an input sentence $x = x_1, \dots, x_n$, our object is to obtain the corresponding tree structures for each expression $e \in \mathcal{E}$ and compose them to construct the Latent Dependency Graph ultimately.

A directed dependency tree t is defined by assigning a head $h \in \{x_0, x_1, \dots, x_n\}$, accompanied by a relation label $l \in \mathcal{L}$ to each modifier $m \in x$. Here, x_0 is typically positioned before x , serving as the root node.

For an expression $e \in \mathcal{E}$ within a consecutive

²The rationale behind Two-stage parsing is grounded in the relative independence of these subtasks: Different from the relation prediction between spans and the boundary identification of span, the inside structure of expression span is minimally influenced by the inside structure of corresponding holder/target spans and vice versa.

word span x_i, \dots, x_j and assigned a sentiment label $l \in \mathcal{L}$, we constraint all potential subtrees within this span to be single-rooted at a potential headword h , which is not realized yet. This concept is illustrated in Figure 2(a), where the sentiment label l is allocated as the label of the dependency (i.e., the expression label with polarity) originating from x_0 to the headword. A parallel approach is adopted for non-expression spans, with the distinction of setting the label to \emptyset and omitting the single-root constraint. Subsequently, we designate all corresponding latent span subtrees as descendants of T_e .

For a corresponding holder/target span with a consecutive word span x_i, \dots, x_j and a sentiment label $l \in \mathcal{L}$, we impose a similar single-rooted constraint as with expression, as shown in Figure 2(b). Accordingly, the sentiment label l denotes the label of the dependency extending from each word in e to the headword. Spans deemed irrelevant are treated akin to non-expression spans.

By enumerating all possible subtrees and accumulating them together, the resultant tree sets T_e and $T_{h/t}$ expand exponential in size. To manage

this during training, we develop a constrained Inside algorithm to perform the enumeration (§ 4.4), designed to prevent the formation of illegal structures that cannot be decoded into meaningful trees.

3.2 Decoding and Recovery on Latent Tree

Assuming that we have trained a parser, our next step involves recovering expression and corresponding holder/target structures after decoding/parsing the highest-scoring dependency tree.

We give illustration in Figure 2(c), we initially identify all expression spans by obtaining the highest-scoring expression tree t^* rooted at x_0 through our parsing method:

$$t^* = \arg \max s(\mathbf{x}, t) \quad (1)$$

Following this, we determine the highest-scoring corresponding tree t^{**} for e using the same algorithm:

$$t^{**} = \arg \max_{t: x_0 \xrightarrow{p} e \in t^*} s(\mathbf{x}, t) \quad (2)$$

where $s(\mathbf{x}, t)$ represents the score of the tree, with p denoting the polarity label of expression spans relative to x_0 . The tree is constrained to have its root in one of the words in e . We then recover corresponding holder/target spans of the expression by transforming all subtrees headed by e into flat spans. If the label l of the dependency $e \rightarrow h$ is not “ \emptyset ” (indicating irrelevant spans), then a complete span is formed, comprising h and its descendants, and is assigned l as its sentiment label. The final SSA output consists of a compilation of all recovered expression spans and their respective holder/target spans.

4 Methodology

Following previous work on dependency parsing (Dozat and Manning, 2017; Zhang et al., 2020; Yang and Tu, 2022a), our model consists of a contextualized encoder and scoring modules. We further propose a constraint TreeCRF to compute the probabilities of the partially-observed trees of SSA.

4.1 Encoder

For a given sentence $\mathbf{x} = x_1, x_1, \dots, x_n$, we introduce special tokens $\langle \text{bos} \rangle$ and $\langle \text{eos} \rangle$ as x_0 and x_{n+1} , respectively. The vector representation for each token $x_i \in \mathbf{x}$ is an amalgamation of five distinct components:

$$\mathbf{e}_i = \left[\mathbf{e}_i^{\text{word}}; \mathbf{e}_i^{\text{lemma}}; \mathbf{e}_i^{\text{pos}}; \mathbf{e}_i^{\text{char}}; \mathbf{e}_i^{\text{BERT}} \right]$$

In this composition, $\mathbf{e}_i^{\text{word}}$, $\mathbf{e}_i^{\text{pos}}$, and $\mathbf{e}_i^{\text{lemma}}$ represent word, part-of-speech (POS), and lemma embeddings, respectively. $\mathbf{e}_i^{\text{char}}$ is derived from the outputs of a CharLSTM layer (Lample et al., 2016). Lastly, $\mathbf{e}_i^{\text{BERT}}$ constitutes the word-level embeddings obtained through mean-pooling at the last layer of BERT (Devlin et al., 2019), specifically by averaging all subword embeddings.

Then we obtain the hidden representation and the of each vectorial token representations x_i via a deep BiLSTMs (Gal and Ghahramani, 2016) encoder.

$$\begin{aligned} \mathbf{h}_0, \mathbf{h}_1, \dots, \mathbf{h}_n &= \text{BiLSTMs}(e_0, e_1, \dots, e_n) \\ \mathbf{c}_0, \mathbf{c}_1, \dots, \mathbf{c}_n &= \text{BiLSTMs}(e_0, e_1, \dots, e_n) \end{aligned} \quad (3)$$

where f_i and b_i are the forward and backward hidden states of the last BiLSTM layer at position i respectively, $h_i = [f_i, b_i]$ is the token representation of x_i , $c_i = [f_i, b_{i+1}]$ is the boundary representation for the i th boundary lying between x_i and x_{i+1} .

4.2 Tree Scoring

We decompose a tree t into two distinct components: \mathbf{y} , representing an unlabeled skeletal tree, and \mathbf{l} , signifying the corresponding sequence of labels. The process of scoring an unlabeled skeletal tree involves the aggregation of arcs and head-span scores. For each head-modifier pair $h \rightarrow m \in \mathbf{y}$, we score them using two MLPs followed by a Biaffine layer (first-order scorer on arcs in the tree):

$$\begin{aligned} \mathbf{r}_i^{\text{head/mod}} &= \text{MLP}^{\text{head/mod}}(\mathbf{h}_i) \\ \mathbf{s}_{h \rightarrow m}^{\text{arc}} &= \left[\mathbf{r}_h^{\text{head}}; 1 \right]^T W \left[\mathbf{r}_m^{\text{mod}}; 1 \right] \end{aligned} \quad (4)$$

The scoring of the dependency $h \rightarrow m$ with label $l \in \mathcal{L}$ is calculated in a similar manner. We use two additional MLPs and $|\mathcal{L}|$ Biaffine layers to compute all label scores.

Enhancing the first-order biaffine parser for the unlabeled tree, we leverage adjacent-sibling information as mentioned in McDonald and Pereira (2006) and headed-span information as described in Yang and Tu (2022b). Additional MLPs and biaffine/ triaffine layers are included to perform the

scoring,

$$\mathbf{r}_i^{\text{head/mod/sib}} = \text{MLP}^{\text{head/mod/sib}}(\mathbf{h}_i) \quad (5)$$

$$\mathbf{s}_{h \rightarrow s, m}^{\text{sib}} = \text{TriAff}(\mathbf{r}_s^{\text{sib}}, \mathbf{r}_h^{\text{head}}, \mathbf{r}_m^{\text{mod}}) \quad (6)$$

$$\mathbf{r}_i^{\text{left/right/head}} = \text{MLP}^{\text{left/right}}(\mathbf{c}_i), \text{MLP}^{\text{head}}(\mathbf{h}_i) \quad (7)$$

$$\mathbf{s}_{k, i/j}^{\text{left/right}} = \left[\mathbf{r}_{i/j}^{\text{left/right}}; 1 \right]^T W \left[\mathbf{r}_k^{\text{head}}; 1 \right] \quad (8)$$

where s and m are two adjacent modifiers of h , s populates between h and m , i and j are left / right boundary of the span whose head is k .

Ultimately, the scoring of unlabeled tree contains the accumulation of scores of all first-order arcs, second-order adjacent-sibling arcs and headed-spans in the tree as follow,

$$\begin{aligned} s(\mathbf{x}, \mathbf{y}) = & \sum_{(h \rightarrow m) \in \mathbf{y}} \mathbf{s}_{h, m}^{\text{arc}} + \sum_{(h \rightarrow \{s, m\}) \in \mathbf{y}} \mathbf{s}_{h, s, m}^{\text{sib}} \\ & + \sum_{(l, r, i) \in \mathbf{y}} (\mathbf{s}_{i, l_i}^{\text{left}} + \mathbf{s}_{i, r_i}^{\text{right}}) \end{aligned} \quad (9)$$

We parameterize the probabilities of skeletal tree \mathbf{y} as:

$$P(\mathbf{y} | \mathbf{x}) = \frac{\exp(s(\mathbf{x}, \mathbf{y}))}{Z(\mathbf{x}) \equiv \sum \exp(s(\mathbf{x}, \mathbf{y}'))} \quad (10)$$

The term $Z(\mathbf{x})$ represents the partition function, which is calculable via the Inside algorithm within the TreeCRF framework. Parameterization of the related label sequence l is as follows:

$$P(l | \mathbf{x}, \mathbf{y}) = \prod_{h \xrightarrow{l} m \in \mathbf{t}} P(l | \mathbf{x}, h \rightarrow m) \quad (11)$$

Each label l operates independently of the tree \mathbf{y} and other labels. In conclusion, the probability of the labeled tree \mathbf{t} is defined as the product of the probabilities of two components, namely, the unlabeled tree and the associated labels.

$$P(\mathbf{t} | \mathbf{x}) = P(\mathbf{y} | \mathbf{x}) \cdot P(l | \mathbf{x}, \mathbf{y}) \quad (12)$$

4.3 Training Objective

During training, the objective is to maximize the probability of tree T_e and $T_{h/t}$ for each expression $e \in \mathcal{E}$. Consequently, we formulate the loss function in the following manner:

$$\mathcal{L} = - \sum_{e \in \mathcal{E}} \log P(T_e | \mathbf{x}) \cdot P(T_{h/t} | \mathbf{x}) \quad (13)$$

In this equation, the term $P(T_e | \mathbf{x})$ and $P(T_{h/t} | \mathbf{x})$ are expanded as:

$$\begin{aligned} P(T | \mathbf{x}) &= \sum_{\mathbf{t} \in T} P(\mathbf{y} | \mathbf{x}) \cdot P(l | \mathbf{x}, \mathbf{y}) \\ &= \frac{1}{Z(\mathbf{x})} \sum_{\mathbf{t} \in T} \exp(s(\mathbf{x}, \mathbf{y})) \cdot P(l | \mathbf{x}, \mathbf{y}) \end{aligned} \quad (14)$$

4.4 Inside Algorithm

The calculation of the partition function $Z(\mathbf{x})$ in Eq. (10) can be resolved by the Inside algorithm of TreeCRF. Follow (Zhang et al., 2022), we logarithm the scores and define the labeled tree score as:

$$s(\mathbf{x}, \mathbf{t}) = s(\mathbf{x}, \mathbf{y}) + \log P(l | \mathbf{x}, \mathbf{y}) \quad (15)$$

Consequently, the score represents the summation of the exponential scores of all legal labeled trees, as the logarithmic label probability of illegal trees is set to 0 in our conversion formulation.

To enumerate legal trees, we introduce constraints to the Inside Algorithm as proposed by Eisner (1997); Li et al. (2016)³. The constraints are categorized into two groups, each defined by its specific purpose: (1) To prevent the arc $h \rightarrow m$ from crossing different spans, we apply a constraint to the rule (**R-LINK**), thereby prohibiting merging with the relevant incomplete span $I_{h, m}$. (2) To prevent the presence of multiple headwords within a single span, we restrict any word in expression spans e to merge solely with the completed span $F_{h, i}$ (**R-COMB**). Additionally, we permit a span to be considered complete only when i is positioned at the endpoint of a span (**R-FINISH**). We demonstrate the deduction rules via the parsing-as-deduction framework (Pereira and Warren, 1983) in Appendix B, specifically in Figure 4. For additional insights into the Eisner Algorithm, Appendix B provides further details.

5 Experiments

5.1 Datasets

Following the previous work, we conduct experiments on five benchmark datasets in four languages. **NoReC_{Fine}** (Øvrelid et al., 2020) is a multi-domain professional reviews dataset in Norwegian. **MultiB_{EU}** and **MultiB_{CA}** (Barnes et al., 2018) are

³We employ a modified form of Yang and Tu (2022a) to differentiate between finished and unfinished spans

Dataset	Model	Span			Sent. Graph	
		Holder F1	Target F1	Exp. F1	NSF1	SF1
NoReC_{Fine}	RACL-BERT (Chen and Qian, 2020)	-	47.2	56.3	-	-
	Head-first (Barnes et al., 2021)	51.1	50.1	54.4	37.0	29.5
	Head-final (Barnes et al., 2021)	60.4	54.8	55.5	39.2	31.2
	Frozen PERIN (Samuel et al., 2022)	48.3	51.9	57.9	41.8	35.7
	TGLS (Shi et al., 2022)	60.9	53.2	61.0	46.4	37.6
	USSA (Zhai et al., 2023)	66.3	54.3	61.4	47.7	39.6
	Ours [†]	67.4	54.5	62.7	49.5	41.5
MultiB_{EU}	RACL-BERT (Chen and Qian, 2020)	-	59.9	72.6	-	-
	Head-first (Barnes et al., 2021)	60.4	64.0	73.9	58.0	54.7
	Head-final (Barnes et al., 2021)	60.5	64.0	72.1	58.0	54.7
	Frozen PERIN (Samuel et al., 2022)	55.5	58.5	68.8	53.1	51.3
	TGLS (Shi et al., 2022)	62.8	65.6	75.2	61.1	58.9
	USSA (Zhai et al., 2023)	63.4	66.9	75.4	63.5	60.4
	Ours [†]	65.5	68.2	75.8	65.7	62.7
MultiB_{CA}	RACL-BERT (Chen and Qian, 2020)	-	67.5	70.3	-	-
	Head-first (Barnes et al., 2021)	43.0	72.5	71.1	62.0	56.8
	Head-final (Barnes et al., 2021)	37.1	71.2	67.1	59.7	53.7
	Frozen PERIN (Samuel et al., 2022)	39.8	69.2	66.3	60.2	57.6
	TGLS (Shi et al., 2022)	47.4	73.8	71.8	64.2	59.8
	USSA (Zhai et al., 2023)	47.5	74.2	72.2	67.4	61.0
	Ours [†]	50.3	75.2	74.7	69.7	62.8
MPQA	RACL-BERT (Chen and Qian, 2020)	-	20.0	31.2	-	-
	Head-first (Barnes et al., 2021)	43.8	51.0	48.1	24.5	17.4
	Head-final (Barnes et al., 2021)	46.3	49.5	46.0	26.1	18.8
	Frozen PERIN (Samuel et al., 2022)	44.0	49.0	46.6	30.7	23.1
	TGLS (Shi et al., 2022)	44.1	51.7	47.8	28.2	21.6
	USSA (Zhai et al., 2023)	47.3	58.9	48.0	36.8	30.5
	Ours [†]	51.2	60.2	48.2	40.1	32.4
DS_{Unis}	RACL-BERT (Chen and Qian, 2020)	-	44.6	38.2	-	-
	Head-first (Barnes et al., 2021)	28.0	39.9	40.3	31.0	25.0
	Head-final (Barnes et al., 2021)	37.4	42.1	45.5	34.3	26.5
	Frozen PERIN (Samuel et al., 2022)	13.8	37.3	33.2	24.5	21.3
	TGLS (Shi et al., 2022)	43.7	49.0	42.6	36.1	31.1
	USSA (Zhai et al., 2023)	44.2	50.2	46.6	38.0	33.2
	Ours [†]	44.4	51.0	48.2	40.1	35.7

Table 2: Main experimental results of our model and comparison with previous works. The score marked as bold means the best performance among all the methods. † means that the reported result is the performance of the high-order parser.

annotated hotel views in Basque and Catalan, respectively. **MPQA** (Wiebe et al., 2005) contains English news and the main content of **DS_{Unis}** (Toprak et al., 2010) is online university reviews in English as well.

5.2 Baselines

We compare our proposed method with six state-of-the-art baselines. **RACL-BERT** (Chen and Qian, 2020) is a relation aware framework used to resolve subtasks of ABSA coordinately. **Head-first** and **Head-final** (Barnes et al., 2021) are symmetric bi-lexical dependency formulation with first-order biaffine parser. **Frozen PERIN** (Samuel et al., 2022) use graph-based parser to directly extraction sentiment tuple from text. **TGLS** (Shi et al., 2022) is a bi-lexical dependency parsing method applying

Graph attention network to make prediction. **USSA** (Zhai et al., 2023) is the same bi-lexical method with table filling scheme for prediction. About the details we make fair comparison, refer to Appendix A please.

5.3 Evaluation Metrics

Following the previous work (Zhai et al., 2023), we use **Holder F1**, **Target F1** and **Exp. F1** for the token extraction of Holders, Targets and Expressions. For structure prediction, we use **Sentiment Graph F1 (SF1)** and **Non-polarity Sentiment Graph F1 (NSF1)** to evaluate the model, which perform exact match evaluation on the full sentiment tuple (h, t, e, p) and non-polarity tuple (h,t,e).

	Method	Sents/s
Head-first (Barnes et al., 2021)	1O	170
TGLS (Shi et al., 2022)	TF	79
USSA (Zhai et al., 2023)	TF	50
Ours	1O	212
	2O & SPAN	174

Table 3: Speed comparison on MPQA Test dataset. The dataset contains 1681 sentences with average 24 tokens. TF:Table-filling method, 1O:first-order parser, 2O:second-order parser.

5.4 Main Results

Table 2 shows the comparison of our method against other baselines across multiple evaluation metrics. In terms of the Span F1 metric, our method demonstrates superior performance on all datasets, including a notable 7.2% F1 score increase in holder extraction on the MPQA dataset. Furthermore, when evaluating the Sentiment Graph metric, designed to assess both span extraction and relation prediction accuracy, our method consistently outperforms over competing approaches in both NSF1 and SF1 scores. Notably, it surpasses the USSA baseline by an average of 2.12 NSF1 and 2.08% SF1. This enhancement is primarily attributed to our SSA as a latent tree formulation and the effective parser, which accurately models the internal structure of spans.

5.5 Efficiency Comparison

Table 3 presents a comparative analysis of different models based on their processing speeds. The speed metrics for previous works were derived by rerunning their publicly available code. Our models, the first-order scoring only method and the comprehensive scoring method, demonstrate superior performance, processing approximately 212 and 174 sentences per second, respectively. This rate significantly surpasses that of prior models. TGLS (Shi et al., 2022) and USSA (Zhai et al., 2023) employ deeper networks and engage in computationally intensive tasks, which contributes to their slower processing speeds.

6 Discussion

6.1 Can SSA as latent tree formulation handle overlap and discontinuous cases?

SSA as a latent tree formulation effectively addresses not only latent tree modeling but also the management of overlap and discontinuous scenarios, as highlighted by Zhai et al. (2023). We offer a

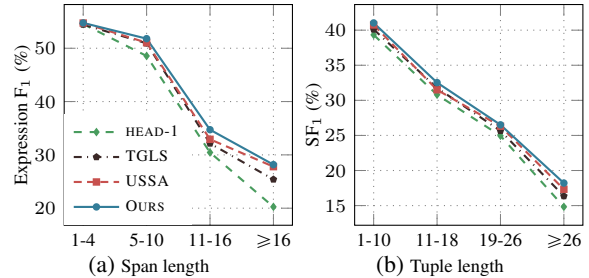


Figure 3: Analysis on long span identification in NoReC Test Dataset. The dataset contains 8448 expressions with up to 30 token and average length of 4.9 tokens. The Head-1 represents head-first method. (a) Expression F1 scores breakdown by span length (b) SF1 scores breakdown by tuple length.

succinct proof as follows: (1) **Overlap Case**: Our approach handles overlaps efficiently by treating each expression separately. This allows different expressions to share the same target/holder without conflict in our framework. (2) **Discontinuous Case**: This more complex scenario, predominantly found in expressions⁴, is also addressed in our scheme. We introduce a new label, “**exp-incomplete**”, during the expression extraction subtask. This label is based on the observation that discontinuous cases often include modifiers like adverbs of degree, affirmation, probability, or negation, which can be identified independently. For instance, in “It is **by no means** a **diploma mill**”, the discontinuous expression (“by no means” and “diploma mill”) is handled by recognizing “by no means” as an incomplete negation, which, when attached to a complete expression like “diploma mill” (negative), alters the overall sentiment polarity to positive in our framework.⁵

6.2 Does the latent tree structure benefit for long spans/tuples?

We address the question through results from two key experiments: (1) Experiment on **NoReC_{Fine}** (Figure 3) investigates the model’s performance in longer spans/tuples. The Expression F1 and Tuple F1 metrics indicate our method’s superiority over baselines, with an increasing performance gap correlating with longer span/tuple lengths. (2) To assess the importance of modeling spans as latent

⁴Discontinuity occurs in approximately 2% of targets and holders and 4%-9% of expressions across five benchmarks, excluding the MPQA dataset, which lacks discontinuous cases. See Zhai et al. (2023) for detailed statistics.

⁵It’s important to note that incomplete expression spans do **not** correspond to specific target/holder pairs; they are solely linked to complete expressions.

	Holder F1	Target F1	Exp. F1	SF1
Ours	51.2	60.2	48.2	32.4
First	48.5	57.4	48.1	27.8
Last	48.6	57.2	48.0	27.5
Flat	45.4	53.3	47.6	24.1

Table 4: Experimental results of our first-order method and three variants methods **Flat**, **First** and **Final** with various degrees of excluding internal structures in MPQA Test dataset.

trees, we examine three variants of our first-order method: 1) **FIRST**: Utilizes the first word as the span headword. 2) **LAST**: Employs the last word as the span headword. 3) **FLAT**: Similar to the head-first method by [Barnes et al. \(2021\)](#), attaches span words directly to the first word. Results in Table 4 reveal that both **FIRST** and **LAST** perform comparably but are consistently outperformed by our first-order method. **FLAT** records a significant drop in performance (24.1 SF1 on **MPQA**) compared to our method (32.4 SF1), indicating the substantial contribution of latent span representations to our model’s effectiveness.

7 Conclusion

In this study, we approach Structured Sentiment Analysis (SSA) through latent dependency graph parsing, conceptualizing flat sentiment spans as latent subtrees. We introduce an innovative parsing methodology grounded in TreeCRF, designed to effectively integrate span structures. Our experimental findings demonstrate that this method surpasses all previous approaches across five benchmark datasets. Comprehensive analyses validate the efficacy and consistency of our method in enhancing SSA.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (No. 62176187).

Limitations

We propose a two-stage parsing method to model the converted latent tree derived from the original SSA structure. The experiments demonstrate that our method outperforms the previous methods on benchmark datasets and proves its temporal efficiency. However, inevitable error propagation occurs in our decoding stage due to the sequential

nature of the two-stage decoding process. Another challenge is that considering each expression span separately to construct their corresponding trees incurs increased space requirements for storing intermediate results.

Ethics Statement

Our work on the Structured Sentiment Analysis (SSA) as Latent Graph Parsing Scheme adheres to ethical guidelines emphasizing transparency, fairness, and responsible AI development. We recognize the ethical implications of this work and have conducted our research with a commitment to minimizing biases, ensuring data privacy, and promoting the explainability of AI decisions. Our evaluations utilized publicly available or ethically sourced datasets, and we have made efforts to address and mitigate potential biases within these datasets to ensure fairness and objectivity in our findings.

The broader impact of Latent Graph Parsing Scheme, aimed at improving sentiment analysis for a more complete conceptualization, has the potential to contribute positively to various fields, including negation resolution, uncertainty/hedge detection, and event extraction. By introducing a graph parsing method, we foster more faithful, flexible, and explainable sentence-level sentiment analysis capabilities. We encourage the responsible use of our findings and technologies, and we commit to ongoing evaluation of our work’s societal and ethical implications.

References

- Jeremy Barnes, Toni Badia, and Patrik Lambert. 2018. [MultiBooked: A Corpus of Basque and Catalan Hotel Reviews Annotated for Aspect-level Sentiment Classification](#). In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).
- Jeremy Barnes, Robin Kurtz, Stephan Oepen, Lilja Øvrelid, and Erik Velldal. 2021. [Structured Sentiment Analysis as Dependency Graph Parsing](#). In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3387–3402, Online. Association for Computational Linguistics.
- Zhuang Chen and Tiejun Qian. 2020. [Relation-Aware Collaborative Learning for Unified Aspect-Based](#)

- Sentiment Analysis**. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3685–3694, Online. Association for Computational Linguistics.
- 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.
- Jason Eisner. 1997. **Bilexical Grammars and a Cubic-time Probabilistic Parser**. In *Proceedings of the Fifth International Workshop on Parsing Technologies*, pages 54–65, Boston/Cambridge, Massachusetts, USA. Association for Computational Linguistics.
- Jason Eisner and Giorgio Satta. 1999. **Efficient parsing for bilexical context-free grammars and head automaton grammars**. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics on Computational Linguistics, ACL '99*, pages 457–464, USA. Association for Computational Linguistics. Event-place: College Park, Maryland.
- Zhifang Fan, Zhen Wu, Xin-Yu Dai, Shujian Huang, and Jiajun Chen. 2019. **Target-oriented Opinion Words Extraction with Target-fused Neural Sequence Labeling**. 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 2509–2518. Association for Computational Linguistics.
- Hao Fei, Bobo Li, Qian Liu, Lidong Bing, Fei Li, and Tat-Seng Chua. 2023a. Reasoning implicit sentiment with chain-of-thought prompting. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 1171–1182.
- Hao Fei, Fei Li, Chenliang Li, Shengqiong Wu, Jingye Li, and Donghong Ji. 2022a. Inheriting the wisdom of predecessors: A multiplex cascade framework for unified aspect-based sentiment analysis. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI*, pages 4096–4103.
- Hao Fei, Qian Liu, Meishan Zhang, Min Zhang, and Tat-Seng Chua. 2023b. Scene graph as pivoting: Inference-time image-free unsupervised multimodal machine translation with visual scene hallucination. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5980–5994.
- Hao Fei, Shengqiong Wu, Jingye Li, Bobo Li, Fei Li, Libo Qin, Meishan Zhang, Min Zhang, and Tat-Seng Chua. 2022b. Lasuie: Unifying information extraction with latent adaptive structure-aware generative language model. In *Proceedings of the Advances in Neural Information Processing Systems, NeurIPS 2022*, pages 15460–15475.
- Hao Fei, Shengqiong Wu, Yafeng Ren, and Meishan Zhang. 2022c. Matching structure for dual learning. In *Proceedings of the International Conference on Machine Learning, ICML*, pages 6373–6391.
- Erick Fonseca and André F. T. Martins. 2020. **Revisiting Higher-Order Dependency Parsers**. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8795–8800, Online. Association for Computational Linguistics.
- Yao Fu, Chuanqi Tan, Mosha Chen, Songfang Huang, and Fei Huang. 2021. **Nested Named Entity Recognition with Partially-Observed TreeCRFs**. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 12839–12847. AAAI Press.
- Yarin Gal and Zoubin Ghahramani. 2016. **Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning**. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1050–1059, New York, New York, USA. PMLR.
- Zilong Huang, Xinggang Wang, Lichao Huang, Chang Huang, Yunchao Wei, and Wenyu Liu. 2019. **CCNet: Criss-Cross Attention for Semantic Segmentation**. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 603–612.
- Arzoo Katiyar and Claire Cardie. 2016. **Investigating LSTMs for Joint Extraction of Opinion Entities and Relations**. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*. The Association for Computer Linguistics.
- Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. **Neural Architectures for Named Entity Recognition**. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270, San Diego, California. Association for Computational Linguistics.

- Bobo Li, Hao Fei, Fei Li, Yuhan Wu, Jinsong Zhang, Shengqiong Wu, Jingye Li, Yijiang Liu, Lizi Liao, Tat-Seng Chua, and Donghong Ji. 2023. DiaASQ: A benchmark of conversational aspect-based sentiment quadruple analysis. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 13449–13467.
- Bobo Li, Hao Fei, Lizi Liao, Yu Zhao, Fangfang Su, Fei Li, and Donghong Ji. 2024. Harnessing holistic discourse features and triadic interaction for sentiment quadruple extraction in dialogues. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 16, pages 18462–18470.
- Jingye Li, Hao Fei, Jiang Liu, Shengqiong Wu, Meishan Zhang, Chong Teng, Donghong Ji, and Fei Li. 2022. Unified named entity recognition as word-word relation classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 10965–10973.
- Zhenghua Li, Min Zhang, Yue Zhang, Zhanyi Liu, Wenliang Chen, Hua Wu, and Haifeng Wang. 2016. [Active Learning for Dependency Parsing with Partial Annotation](#). In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 344–354, Berlin, Germany. Association for Computational Linguistics.
- Chao Lou, Songlin Yang, and Kewei Tu. 2022. [Nested Named Entity Recognition as Latent Lexicalized Constituency Parsing](#). In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, ACL 2022, Dublin, Ireland, May 22-27, 2022, pages 6183–6198. Association for Computational Linguistics.
- Yue Mao, Yi Shen, Chao Yu, and Longjun Cai. 2021. [A Joint Training Dual-MRC Framework for Aspect Based Sentiment Analysis](#). In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 13543–13551. AAAI Press.
- Ryan McDonald and Fernando Pereira. 2006. [Online Learning of Approximate Dependency Parsing Algorithms](#). In *11th Conference of the European Chapter of the Association for Computational Linguistics*, pages 81–88, Trento, Italy. Association for Computational Linguistics.
- 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.
- Fernando C. N. Pereira and David H. D. Warren. 1983. [Parsing as deduction](#). In *Proceedings of the 21st Annual Meeting on Association for Computational Linguistics, ACL '83*, pages 137–144, USA. Association for Computational Linguistics. Event-place: Cambridge, Massachusetts.
- 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 Loukachevitch, Evgeniy Kotelnikov, Nuria Bel, Salud María Jiménez-Zafra, and Gülşen Eryiğit. 2016. [SemEval-2016 Task 5: Aspect Based Sentiment Analysis](#). In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 19–30, San Diego, California. Association for Computational Linguistics.
- Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Haris 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.
- David Samuel, Jeremy Barnes, Robin Kurtz, Stephan Oepen, Lilja Øvrelid, and Erik Velldal. 2022. [Direct parsing to sentiment graphs](#). In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 470–478, Dublin, Ireland. Association for Computational Linguistics.
- Wenxuan Shi, Fei Li, Jingye Li, Hao Fei, and Donghong Ji. 2022. [Effective Token Graph Modeling using a Novel Labeling Strategy for Structured Sentiment Analysis](#). In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, ACL 2022, Dublin, Ireland, May 22-27, 2022, pages 4232–4241. Association for Computational Linguistics.
- Cigdem Toprak, Niklas Jakob, and Iryna Gurevych. 2010. [Sentence and Expression Level Annotation of Opinions in User-Generated Discourse](#). In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 575–584, Uppsala, Sweden. Association for Computational Linguistics.
- Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. [Graph Attention Networks](#). In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.
- Huiyu Wang, Yukun Zhu, Bradley Green, Hartwig Adam, Alan Yuille, and Liang-Chieh Chen. 2020. [Axial-DeepLab: Stand-Alone Axial-Attention for Panoptic Segmentation](#). In *Computer Vision – ECCV*

- 2020, pages 108–126, Cham. Springer International Publishing.
- Wenya Wang, Sinno Jialin Pan, Daniel Dahlmeier, and Xiaokui Xiao. 2016. [Recursive Neural Conditional Random Fields for Aspect-based Sentiment Analysis](#). In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 616–626, Austin, Texas. Association for Computational Linguistics.
- Xinyu Wang and Kewei Tu. 2020. [Second-Order Neural Dependency Parsing with Message Passing and End-to-End Training](#). In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 93–99, Suzhou, China. Association for Computational Linguistics.
- Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. [Annotating Expressions of Opinions and Emotions in Language](#). *Lang. Resour. Evaluation*, 39(2-3):165–210.
- Shengqiong Wu, Hao Fei, Wei Ji, and Tat-Seng Chua. 2023a. [Cross2StrA: Unpaired cross-lingual image captioning with cross-lingual cross-modal structure-pivoted alignment](#). In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2593–2608.
- Shengqiong Wu, Hao Fei, Fei Li, Meishan Zhang, Yijiang Liu, Chong Teng, and Donghong Ji. 2022. [Mastering the explicit opinion-role interaction: Syntax-aided neural transition system for unified opinion role labeling](#). In *Proceedings of the Thirty-Sixth AAAI Conference on Artificial Intelligence*, pages 11513–11521.
- Shengqiong Wu, Hao Fei, Leigang Qu, Wei Ji, and Tat-Seng Chua. 2023b. [Next-gpt: Any-to-any multi-modal llm](#). *arXiv preprint arXiv:2309.05519*.
- Shengqiong Wu, Hao Fei, Yafeng Ren, Donghong Ji, and Jingye Li. 2021. [Learn from syntax: Improving pair-wise aspect and opinion terms extraction with rich syntactic knowledge](#). In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*, pages 3957–3963.
- Qingrong Xia, Bo Zhang, Rui Wang, Zhenghua Li, Yue Zhang, Fei Huang, Luo Si, and Min Zhang. 2021. [A Unified Span-Based Approach for Opinion Mining with Syntactic Constituents](#). In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1795–1804, Online. Association for Computational Linguistics.
- Songlin Yang and Kewei Tu. 2022a. [Combining \(Second-Order\) Graph-Based and Headed-Span-Based Projective Dependency Parsing](#). In *Findings of the Association for Computational Linguistics: ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 1428–1434. Association for Computational Linguistics.
- Songlin Yang and Kewei Tu. 2022b. [Headed-Span-Based Projective Dependency Parsing](#). In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 2188–2200. Association for Computational Linguistics.
- Juntao Yu, Bernd Bohnet, and Massimo Poesio. 2020. [Named Entity Recognition as Dependency Parsing](#). In *Annual Meeting of the Association for Computational Linguistics*.
- Zepeng Zhai, Hao Chen, Fangxiang Feng, Ruifan Li, and Xiaojie Wang. 2022. [COM-MRC: A CONTEXT-MASKED MACHINE READING COMPREHENSION FRAMEWORK FOR ASPECT SENTIMENT TRIPLET EXTRACTION](#). In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 3230–3241, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.
- Zepeng Zhai, Hao Chen, Ruifan Li, and Xiaojie Wang. 2023. [USSA: A Unified Table Filling Scheme for Structured Sentiment Analysis](#). In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 14340–14353. Association for Computational Linguistics.
- Meishan Zhang, Qiansheng Wang, and Guohong Fu. 2019. [End-to-end neural opinion extraction with a transition-based model](#). *Inf. Syst.*, 80:56–63.
- Yu Zhang, Zhenghua Li, and Min Zhang. 2020. [Efficient Second-Order TreeCRF for Neural Dependency Parsing](#). In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 3295–3305. Association for Computational Linguistics.
- Yu Zhang, Qingrong Xia, Shilin Zhou, Yong Jiang, Guohong Fu, and Min Zhang. 2022. [Semantic Role Labeling as Dependency Parsing: Exploring Latent Tree Structures inside Arguments](#). In *Proceedings of the 29th International Conference on Computational Linguistics, COLING 2022, Gyeongju, Republic of Korea, October 12-17, 2022*, pages 4212–4227. International Committee on Computational Linguistics.
- Lilja Øvrelid, Petter Mæhlum, Jeremy Barnes, and Erik Velldal. 2020. [A Fine-grained Sentiment Dataset for Norwegian](#). In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 5025–5033, Marseille, France. European Language Resources Association.

Algorithm 1 The Second-order Inside Algorithm.

```
1: Define:  $I, S, C, F \in \mathbf{R}^{n \times n \times b}$ 
2:
3: Initialize:  $C_{i,i} = \mathbf{0}, 0 \leq i \leq n$ 
4: for  $w = 1$  to  $n$  do
5:   Parallelization on  $0 \leq i; j = i + w \leq n$ 
6:    $F_{i,j} \leftarrow C_{i,j} + s_{i,j}^{right}$ 
7:    $F_{j,i} \leftarrow C_{j,i} + s_{j,i}^{left}$ 
    $I_{i,j} \leftarrow \log(\exp(C_{i,i} + F_{j,i+1})$ 
8:      $+ \sum_{i < r < j} \exp(I_{i,r} + S_{r,j} + s_{i,r,j}^{sib}))$ 
      $+ s_{i,j}^{arc}$ 
    $I_{j,i} \leftarrow \log(\exp(C_{j,j} + F_{i,j-1})$ 
9:      $+ \sum_{i < r < j} \exp(I_{j,r} + S_{r,i} + s_{i,r,j}^{sib}))$ 
      $+ s_{j,i}^{arc}$ 
10:   $S_{i,j} \leftarrow \log \sum_{i \leq r < j} \exp(F_{i,r} + F_{j,r+1})$ 
11:   $C_{i,j} \leftarrow \log \sum_{i < r \leq j} \exp(I_{i,r} + F_{r,j})$ 
12:   $C_{j,i} \leftarrow \log \sum_{i \leq r < j} \exp(I_{j,r} + F_{r,i})$ 
13: end for
14: return  $F_{0,n}$ 
```

A Implement Details

For fair comparison, we obtain the contextual token representations from the pre-trained BERT-multilingual-base and word2vec skip-gram embeddings used by previous work (Barnes et al., 2021). Furthermore, we use 4-layer BiLSTMs to encode the sentence, with an output size of 768 and the dropout rate is set to 0.3. We train our model for 60 epochs and save the model parameters based on the highest SF1 score on the development set. We conduct training and testing on a NVIDIA A100 Server. The reported results are the averages from five runs with different random seeds. Our proposed model statistically outperforms the baselines at $p < 0.05$.

B Inside algorithm

We give the pseudocode of the modified Inside algorithm (Yang and Tu, 2022a) in Alg. 1 as well as the illustration of deduction rules in Figure 4 to help understanding the algorithm. We highlight the constraint rules in Figure 4, it is the only difference between our proposed one and the vanilla inside algorithm.

In Line 3, the term $C_{i,i}$ denotes the axiom items

\triangleleft_i assigned an initial score of 0. Line 6 is associated with the complete span $C_{i,j}$, assuming all child spans with the right boundary of j are absorbed. This is achieved by adding the scores of headed-split right spans to derive the finished span

$F_{i,j} (\triangleleft_i \rightarrow_j)$ as **R-FINISH** operation. Line 8 aligns with two merging processes depicted in Figure 4.

The incomplete span $I_{i,j} (\square_{i,j})$ emerges from aggregating either complete span $C_{i,i}$ with Finished span $F_{j,i+1}$ (**R-LINK**) or combining incomplete span $I_{i,r}$ with sibling span $S_{j,r}$ (**R-LINK2**). In

Line 10, the sibling span $S_{i,j} (\square_{i,j})$ results from summing finished spans $F_{i,r}$ and $F_{j,r+1}$ (**COMB**). Line 11 details a merging operation for pairs of incomplete span $I_{i,r}$ and finished span $F_{r,j}$ to form a complete span $C_{i,j} (\triangleleft_i \rightarrow_j)$ as **R-COMB** rule.

Lines 9 and 12 describe symmetric L-rules, which are not included in Figure 4.

C Comparing to Large Language Models

	NoReC _{Fine}	MultiB _{EU}	MultiB _{CA}	MPQA	DS _{Unis}
Ours	41.5	62.7	62.8	32.4	35.7
GPT 3.5	27.3	29.5	32.3	20.6	22.5
GPT 4	32.3	34.2	35.2	24.5	25.6
+1 shot	35.5	37.5	38.5	<u>25.2</u>	<u>27.6</u>
+5 shot	<u>35.8</u>	<u>38.1</u>	<u>39.8</u>	24.6	23.4

Table 5: Experimental results of the SF1 score utilize the ChatGPT and GPT4 to generate the sentiment tuple of five benchmarks. We underline the best results of GPT performance, which still lags behind our method (the highlight ones) largely.

In Table 5, we present a comparative analysis of our method against ChatGPT-3.5-Turbo and GPT4. The results indicate that on the English dataset, GPT4 surpasses the baseline performance in a zero-shot setting and shows further improvement with in-context learning (1-shot or more). However, GPT4 does not reach the performance levels of more robust baselines such as TGLS and USSA, which are specifically trained for the SSA task.

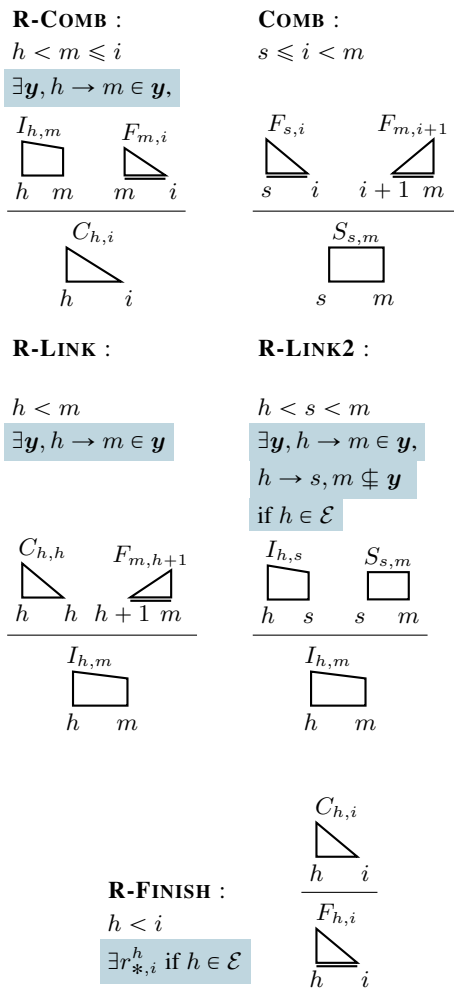


Figure 4: Deduction rules for our constrained Inside algorithm based on Eisner algorithm (Eisner, 1997), we extend it to second-order and introduce span scoring (Yang and Tu, 2022a) for unify the deduction framework to expression identification and expression-target/holder relation prediction. (**R-COMB** and **R-LINK**), combination with span scoring (**R-FINISH**) and its second-order extension (**COMB** and **R-LINK2**). Our modified rule constraints are highlighted in blue color. $r_{*,i}^h$ denotes a span that takes h as the headword and ends with i , $h \in \mathcal{E}$ denotes h is a word within expression span. We show only R-rules, omitting the symmetric L-rules as well as initial conditions for brevity.