Counting What Deserves to be Counted for Graph Parsing

Anonymous ACL submission

Abstract

Graph parsers rely on scoring every subgraphs for building a complete graph. In real syntactic parsing or semantic parsing, every types of subgraphs in terms of syntactic or semantic roles may generate quite unbalanced distribution, which seems not well captured by the current graph paring models. Thus we propose an enhanced model design to let the parser explicitly capture such kind of unbalanced distribution. In detail, we introduce Accumulative Operation-based Induction (AOI) attention mechanism to assign accumulative scores for words. AOI scorer successfully approximates word-level unbalanced distribution. With conceptually simple but general-purpose design, our proposed AOI attention enhancement indeed leads to better parsing performance on a wide range of datasets of different parsing tasks, which verifies the scalability and robustness of capturing diverse subgraph distribution.

1 Introduction

Graph parsing models have been successfully applied onto syntactic and semantic parsing tasks. Generally, graph parser relies on training some kind of subgraph scorers, and the parser itself just simply searches for a complete graph in terms of maximizing the score summing all subgraphs. In practical applications, the computational complexity of graph parsers depends on the order of the model, namely, the number of edges in a subgraph. For the sake of parsing efficiency, order-1 graph parsing is mostly applied.

Thanks for the well-developed deep learning techniques, it offers powerful representation learning ability to enable the subgraph scorer in graph parsers can accurately capture really salient features, and thus yields new high parsing performance for years. However, we argue that the current graph parsers still miss an important part in subgraph scoring when they perform syntactic or semantic parsing tasks. We take order-1 graph parsing as example in Figure 1, in which every subgraph consists of two words and one edge representing their relationship. The corresponding subgraph scorer may be as simple as just determining if such relationship exists for the two words. When we take syntactic or semantic roles of the words into consideration, we will find there comes an unbalanced distribution from every subgraphs for complete graph building.

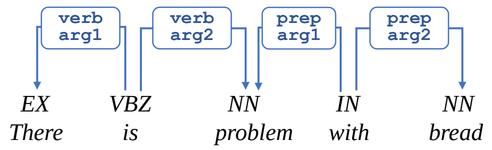

Figure 1: An example of semantic predicate-argument dependency parsing graph.

For instance, a noun (*NN* as part-of-speech tag) has nearly $3\times$ higher probability than an adjective (*JJ* as part-of-speech tag) to be an augment (79.3% v.s. 20.1%) in a semantic dependency parsing dataset (Oepen et al., 2015) for predicate-argument structures. There are also trivial labels in the parsing graph, like edges with *DT* heads will be of high probability (98.8%) to have a det_{A1} label. Unbalanced distribution issue not only occurs in word-level, but edge-level as well since 66.5% edges point to augments right to predicates in this dataset. Moreover, the appearance of second-order structures in parsing graph (Wang et al., 2019) also indicates that there should be even higher level correlation between edges. Solving higher-level unbalance requires complex inducting techniques like the second-order parser (Wang et al., 2019). But word-level unbalance, many of which are constrained by trivial rules, can be solved with rather simple techniques like the attention mechanism.

In this paper, we evaluate and mitigate the word-level unbalance in graph parsing. Our direct intuition is to use the attention mechanism to approximate the unbalanced distribution. The attention layer is positioned before the pairwise scorer to select candidates in advance for certain relationships, including edge existence and label type. After the attention layer filters candidates that are unlikely to be a head or dependent, the pairwise score can concentrate on discerning more complex patterns of remained candidates.

To select or filter candidates in practice, we propose an **A**ccumulative **O**peration-based **I**nduction (AOI) attention scorer for parsing. AOI uses a one-dimensional attention to select candidates for heads and dependents. The attention scores are pooled from global attention scores on multiple attention heads. Compared to conventional global attention mechanisms, accumulated attention enjoys a higher capacity of capturing attention distribution in multiple dependency spans.

Results from our experiments on a wide variety of graph parsing datasets have shown AOI to successfully approximate the word-level unbalanced distribution. Thus, AOI leads to prominent improvement on performance for these parsing tasks compared to the BiAF scorer.

Our contributions are listed as follows:

- We analyze the unbalanced distributions of heads and dependents in parsing graphs and leverage it for improving performance.
- We propose a novel attention scorer, AOI, to better approximate the distribution of candidates for parsing graphs than previous scorers.
- Results from our experiments show that AOI outperforms previous parsers significantly on a wide range of tasks and datasets.

2 Unbalanced Distribution Issue

We show the existence of the unbalanced distribution issue in a wide range of datasets in this section. Specifically, we study the correlation between heads and dependents and their part-of-speech. Due to the length limitation, we only present results on semantic predicate-argument and syntactic dependency graphs here.

Under ideal circumstances, edge distributions are independent of heads or dependents' part-of-speech.

$$q_E = p(E_{ij} = 1|POS_i^h) = p(E_{ij} = 1|POS_j^d)$$
$$\frac{1}{c} = p(C_{ij}|POS_i^h, E_{ij} = 1) = p(C_{ij}|POS_j^d, E_{ij} = 1)$$

	POS	q_E	$H(C)$	C_{most}	Prop.
Head	IN	1.00	1.74	prep$_{A2}$ (44.1%)	20.1%
	DT	0.97	0.11	det$_{A1}$ (98.8%)	8.73%
	JJ	0.98	0.22	adj$_{A1}$ (97.3%)	6.54%
	VBD	1.00	1.76	verb$_{A1}$ (44.3%)	6.54%
	,	1.00	1.67	punct$_{A1}$ (63.6%)	6.22%
	VB	0.98	1.68	verb$_{A1}$ (43.2%)	4.99%
	Uni.	0.70	5.39	- (2.4%)	-
Dep	NN	0.79	3.43	det$_{A1}$ (22.3%)	32.3%
	NNS	0.97	3.42	adj$_{A1}$ (20.7%)	16.7%
	NNP	0.53	3.29	noun$_{A1}$ (33.1%)	11.7%
	VB	0.90	3.42	comp$_{A1}$ (21.0%)	6.84%
	VBD	0.69	3.00	punct$_{A1}$ (29.6%)	5.19%
	VBN	0.80	3.06	aux$_{A2}$ (32.1%)	4.83%
	Uni.	0.41	5.39	- (2.4%)	-

Table 1: Word-level unbalance on semantic predicate-argument dependency dataset SemEval2015 (Oepen et al., 2015).

	POS	q_E	$H(C)$	C_{most}	Prop.
Head	NN	0.72	3.48	det (27.7%)	22.8%
	VBD	0.77	3.10	punct (28.6%)	10.3%
	NNS	0.82	3.51	amod (23.9%)	10.1%
	IN	0.86	0.84	pobj (88.6%)	9.46%
	NNP	0.38	2.94	nn (40.4%)	8.84%
	VB	0.83	3.34	aux (25.9%)	7.98%
	Uni.	0.42	5.45	- (2.3%)	-
Dep	NN	1.00	3.07	pobj(29.9%)	14.0%
	IN	1.00	1.05	prep(82.8%)	10.4%
	NNP	1.00	2.43	nn(47.5%)	9.77%
	DT	1.00	0.41	det(95.5%)	8.61%
	JJ	1.00	1.48	amod(80.0%)	6.50%
	NNS	1.00	2.55	pobj(39.3%)	6.33%
	Uni.	1.00	5.45	- (2.3%)	-

Table 2: Word-level unbalance on syntactic dependency dataset Penn Treebank (Marcus et al., 1993).

where $E_{ij} \in \{0, 1\}$ refers to the existence of an edge from i-th word to j-th word and $C_{ij} \in \{1, \cdots, c\}$ refers to the label of the edge. q_E represents a fixed probability for an edge to exist. As the existence probabilities of edges are uniform, the information entropy of classes $H(C) = \sum_i(-p_i log_2(p_i))$ will always be its maximum, $\log_2 c$. $H(C)$ will drop when part-of-speech contains information about the edge label.

Obviously, this is not the case for edge and label distributions in parsing graphs. What makes things even worse, the issue occurs in types of edges that frequently appear in the parsing graph. We list the top-6 most frequent part-of-speech at the head or dependent in Tables 1 and 2 for syntactic and semantic dependency treebanks. *Uni.* refers to every part-of-speech under the ideal circumstance that edges and labels appear with the same probability in every position. q_E of *Uni.* is estimated based on the statistical property of existing graphs and $H(C)$ is maximized under *Uni.* circumstance. For a direct understanding of the label unbalance, we propose C_{most} which represents the most common label on edges grouped by heads and dependents. Proportion of C_{most} is $\frac{1}{c}$ under $Uni.$ circumstance.

Semantic Dependency Graph In semantic dependency graphs, the most prominent property is the high density of edges with heads in a certain part-of-speech. All 6 part-of-speeches are correlated to at least one edge with extremely high probability ($> 98\%$). *IN*, *DT* and *COMMA* (,) will be the head of an edge with full confidence. 29 in 44 part-of-speeches have $q_E > 0.95$, showing a large group of part-of-speeches to be decisive for the existence of edges. For labels, their distributions are also uniform as $H(C)$ for heads are less than $\frac{1}{3}$ of the maximum, indicating heads' part-of-speeches to carry much information about the edge labels. Part-of-speeches like *DT* and *JJ* have extremely low $H(C)$, 0.11 and 0.22 respectively. They may directly point to certain edge labels, which makes the predictions trivial on these edges.

Syntactic Dependency Graph Distributions in syntactic dependency graphs are similar to semantic ones except that each word in the sentence acts as a dependent due to the property of the dependency tree. Trivial labels also exist on edges with *IN* head part-of-speech and *DT*, *JJ* dependent part-of-speech.

3 Model and Method

3.1 Background

We first give a general description about the BiAF model as the basis for further discussion. For a sentence $W = [w_1, w_2, \ldots, w_n]$ with n words, BiAF embeds those words and their features (lemma, part-of-speech, character) to representations X_{word} and X_{feat} with d_{word} and d_{feat} dimensions respectively and concatenate them to $X \in \mathbb{R}^{n\times(d_{word}+d_{feat})}$.

$$X = \text{Embed}(W) = [X_{word}||X_{feat}].$$

The embedding is contextualized through bidirectional long short term memory (BiLSTM) (Hochreiter and Schmidhuber, 1997) network. Two Multi-layer Perceptrons (MLPs) then project output from BiLSTM to two different latent spaces X^h, X^d for head and dependent representations in a pair.

$$X = \text{BiLSTM}(X),$$
$$X^h, X^d = \text{MLP}^h(X), \text{MLP}^d(X).$$

Above is the procedure of the BiAF encoder to encode sentence W. We then describe how the BiAF scorer uses the representations to produce the final score. The BiAF edge scorer contains a weight tensor $U^{edge} \in \mathbb{R}^{d\times 2\times d}$ and the BiAF label scorer contains a weight tensor of shape $U^{label} \in \mathbb{R}^{d\times c\times d}$, where d refers to the encoding dimension in the encoder and c refers to the number of classes for classification. The BiAF scorer uses those weight tensors and biases b^{edge} and b^{label} to score as follows:

$$\text{BiAF}(x, y) = x^T U y + b,$$
$$S_{ij}^{edge} = \text{BiAF}^{edge}({X_i}^{d;edge}, {X_j}^{d;edge}),$$
$$S_{ij}^{label} = \text{BiAF}^{label}({X_i}^{h;label}, {X_j}^{d;label}).$$

3.2 AOI Scorer

AOI shares the same encoder as in BiAF, and thus we only describe the AOI scorer in this section. For predicting scores for edges and different labels, we first use different MLPs to project them to separate latent scores. In MLP^t, $t \in \{edge, label_1, label_2, \ldots, label_k\}$ where k refers to the number of labels. These MLPs are *specific* MLPs as they project representations for a specific type of scoring. Correspondingly, MLPs in the encoder are *general* MLPs.

$$X^h, X^d = \text{MLP}^t(X^h), \text{MLP}^t(X^d).$$

Here, superfix t is omitted for output as we provide a unified procedure for inference on representations of different types.

Our AOI scorer consists of two attentional subscorers, SelfAttn scorer and Multi-head Gathering Attention (MHGAttn) scorer. In SelfAttn Scorer, we use a single-headed self-attention mechanism, where we obtain dot product scores $S_{i,j}^{SA}$ for edge or label.

$$S_{i,j}^{SA} = X_i^h \cdot X_j^d.$$

MHGAttn is responsible for assigning candidate attention scores. For $X_i^h, X_j^d \in \mathbb{R}^t$ where $t = p \times q$, we split them into p attention heads with dimension q: $X_{i,1}^h, X_{i,2}^h, \ldots, X_{i,p}^h$ and $X_{j,1}^d, X_{j,2}^d, \ldots, X_{j,p}^d$. For t-th attention head of each representations, we get the timestep-averaged representations as global representations.

$$G_t^h, G_t^d = \frac{1}{n}\sum_{m=1}^{n} X_{m,t}^h, \frac{1}{n}\sum_{m=1}^{n} X_{m,t}^d.$$

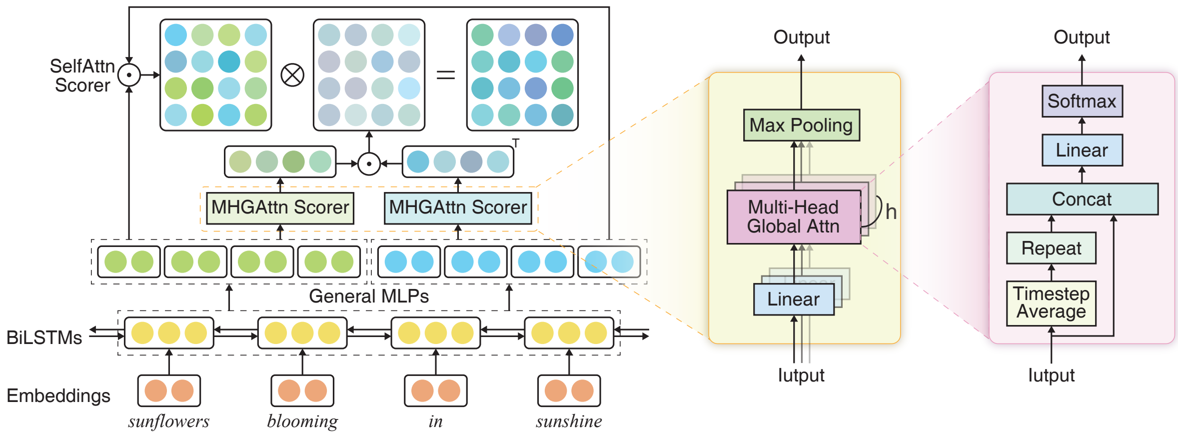

Figure 2: Overall architecture of our proposed graph parser and illustration of subscorers.

Those global representations are then concatenated with each attention head are projected to one-dimension energy scores E and passed through softmax function for attention distribution on this head.

$$E^h_{i,t}, E^d_{j,t} = \text{MLP}([X^h_{i,t}||G^h_t]), \text{MLP}([X^d_{j,t}||G^d_t]),$$
$$E^h_{i,t}, E^d_{j,t} = \frac{exp(E^h_{i,t})}{\sum_{m=1}^{n} exp(E^h_{m,t})}, \frac{exp(E^d_{j,t})}{\sum_{m=1}^{n} exp(E^d_{m,t})}.$$

The attention scores for head and dependent are max pooled attention scores on different attention heads. Mutual product between those scores produces the final MHG attention scores for candidates in the sentence. For the balance of attention on sentences with different lengths, candidate attention is multiple by the sentence length n which acts as a modifier for attention density.

$$\begin{aligned} E^h_i &= max(E^h_{i,1}, E^h_{i,2}, \ldots, E^h_{i,p}), \\ E^d_j &= max(E^d_{j,1}, E^d_{j,2}, \ldots, E^d_{j,q}), \\ S^{MHG}_{i,j} &= E^h_i \times E^d_j \times n \end{aligned}$$

The SelfAttn subscorer focuses on the general assessment of the relation of head-dependent pairs, while the MHGAttn subscorer considers this from a more global view. In order to integrate the advantages of the two subscorers, we adopt a direct product operation on the attention scores from SelfAttn and MHGAttn subscorers to obtain the final attention scores for AOI scorer.

$$S_{i,j} = S^{SA}_{i,j} \times S^{MHG}_{i,j}.$$

Difference between candidate attention and bias in BiAF BiAF contains two bias scorers in word-level. However, scores from these scorers are used to directly modify the logits for prediction. Thus, it still attends to each word equally since adding extra bias will not modify the scale of backward gradients for parameter updating. In contrast, candidate attention in AOI does not change the predicting results from pairwise scorers but instead scales the prediction. The gradient flow of backward propagation will be weakened from predictions that are considered to be trivial by the attention. Thus, AOI attends on non-trivial parts of training, which improves the resulting performance by scaling the weight of training data.

4 Experiment

4.1 Dataset

Our main experiments are conducted on multiple graph parsing dataset.

- **SemDP** We choose SemEval-2015 dataset (Oepen et al., 2015) with three subtasks DM, PAS, PSD, each contains in-domain (ID) and out-of-domain (OOD) test data.

- **Multilingual SemDP** We also conduct experiments on multilingual semantic dependency parsing datasets including Chinese (CZ) and Czech (CS) to verify the cross-language generalization of our method.

- **SynDP** Traditional Penn Treebank (PTB) and Chinese Peen Treebank (CTB) (Marcus et al., 1993) benchmarks are used for model evaluation and performance comparison.

- **SynCP** Like in SynDP, PTB and CTB benchmarks are used for evaluation and comparison.

	POS	BiAF		AOI	
		KL^E	KL^C	KL^E	KL^C
	IN	0.011	0.004	**0.001**	**0.001**
	DT	0.000	0.006	0.000	**0.000**
	JJ	0.004	0.011	**0.000**	**0.002**
Head	VBD	0.001	0.032	**0.000**	**0.030**
	,	0.078	0.001	**0.028**	**0.000**
	VB	0.003	0.014	**0.000**	**0.012**
	NN	0.000	0.011	0.000	**0.008**
	NNS	0.000	0.014	0.000	**0.012**
	NNP	0.000	0.001	0.000	0.001
Dep	VB	0.000	0.040	0.000	**0.025**
	VBD	0.001	0.025	**0.000**	**0.015**
	VBN	0.001	0.073	**0.000**	**0.059**

Table 3: Distance (relative entropy) between predicted and real distributions on semantic predicate-argument parsing .

	POS	BiAF		AOI	
		KL^E	KL^C	KL^E	KL^C
	NN	0.000	0.052	0.000	**0.046**
	VBD	0.001	0.040	**0.000**	**0.034**
	NNS	0.000	0.053	0.000	**0.049**
Head	IN	0.000	0.050	0.000	**0.040**
	NNP	0.000	0.026	0.000	**0.021**
	VB	0.001	0.044	**0.000**	**0.040**
	NN	0.000	0.022	0.000	**0.021**
	IN	0.000	0.014	0.000	**0.010**
	NNP	0.000	0.042	0.000	**0.039**
Dep	DT	0.000	0.030	0.000	**0.027**
	JJ	0.000	0.086	0.000	**0.056**
	NNS	0.000	0.095	0.000	**0.087**

Table 4: Distance (relative entropy) between predicted and real distributions on syntactic dependency parsing.

4.2 Training Configuration

The full configuration is omitted here and can be found in Appendix A. For embedding, we use pre-trained GloVe embedding (Pennington et al., 2014) for fine-tuning. Features, including char, lemma, and POS, are incorporated through concatenation. BERT embedding is projected to lower dimensions and concatenated as a feature. Representation dimensions of edges and labels in the AOI scorer are the same as the output of the encoder. As DM and PAS dependency edges are more concentrated to several words than PSD edges, we use 2 attention heads in AOI for DM/PAS and 4 attention heads for PSD. For constituent parsing, we set attention heads in AOI scorer to 2. Dropout (Srivastava et al., 2014) is added to Embedding Layers, MLPs and LSTMs to prevent overfitting.

To be more detailed in training process, we use Adam optimizer (Kingma and Ba, 2015) for parameter updating. Cross entropy loss is calculated for optimization, and only labels on exist edges involve in loss calculation for the label scorer. For BERT, we apply *bert-large-cased* for English datasets, *bert-base-chinese* for Chinese datasets, and *bert-base-multilingual-cased* for multilingual datasets.

4.3 Unbalanced Distribution Approximation

The results for unbalanced distribution approximation are presented in Table 3 and 4. Relative entropy is applied to evaluate the distance between distributions of predictions and real data. The edge distribution is 2-dimension and the label distribution is c-dimension. AOI approximates the real distribution prominently better as the distance is reduced for all part-of-speeches on both syntactic and semantic dependency graphs, except for some cases that relative entropy is lower than 0.001.

As semantic dependency graph is in rather an irregular pattern compared to the syntactic dependency graph, AOI reduces more distribution distance of edge existence on semantic dependency graphs. For label distribution, the distance reduction is significant and can be attributed to MH-GAttn's label-wise candidate attention assigning, which modifies the label distributions by attention scores.

4.4 Semantic Parsing Results

English SemDP Results from our experiments on English SemDP datasets are shown in Table 5. We re-implement the BiAF parser and find its performance close to previously reported results. We then run our AOI parser on these datasets and find a salient performance improvement, especially on the PSD dataset, where the AOI parser results in nearly 1.0 F1 score improvement. On average, the AOI parser leads to about 0.6 F1 score improvement on both ID and OOD datasets from the previous baseline BiAF parser. Remarkably, AOI has reached a new SOTA with no extra auxiliary mechanism by defeating the BiAF model with second-order method incorporated as auxiliary mechanism (Wang et al., 2019). We also compare the performance of BiAF and AOI with the incorporation of second-order refining and BERT. Experiment results have shown AOI still results in more significant improvement, which is strong proof of the efficiency of our AOI parser.

Model	DM		PAS		PSD		Avg	
	ID	OOD	ID	OOD	ID	OOD	ID	OOD
(Du et al., 2015)	89.09	81.84	91.26	87.23	75.66	73.28	85.34	80.78
(Almeida and Martins, 2015)	88.21	81.75	90.88	86.88	76.36	74.82	85.15	81.15
(Peng et al., 2017)	90.40	85.30	92.70	89.00	78.50	76.40	87.20	83.60
(Wang et al., 2018)	90.30	84.90	91.70	87.60	78.60	75.90	86.90	82.80
BiAF (Dozat and Manning, 2018)	93.70	88.90	93.90	90.60	81.00	79.40	89.50	86.30
BiAF	93.52	88.92	93.87	90.78	81.30	79.27	89.56	86.32
AOI	**93.92**	**89.32**	**94.18**	**91.15**	**82.27**	**79.78**	**90.12**	**86.75**
BiAF2o (Wang et al., 2019)	93.90	89.50	94.20	91.30	81.40	79.50	89.80	86.80
AOI2o	**94.21**	**89.78**	**94.33**	**91.50**	**82.61**	**80.12**	**90.38**	**87.14**
BiAF (w/ BERT)	94.61	91.59	95.04	93.04	82.98	80.10	90.87	88.24
AOI (w/ BERT)	**95.08**	**91.80**	**95.31**	**93.64**	**83.96**	**81.05**	**91.45**	**88.83**

Table 5: Comparison of results on SemEval-2015 SemDP datasets. Underline: significant improvement ($p < 0.05$).

Model	CS-PSD		CZ-PAS	Avg
	ID	OOD	ID	
BiAF	86.12	71.05	86.70	81.29
AOI	**86.67**	**71.61**	**87.60**	**81.96**
BiAF (w/ BERT)	87.04	72.98	88.90	82.97
AOI (w/ BERT)	**87.68**	**73.44**	**89.29**	**83.47**

Table 6: Comparison of results on multilingual SemDP datasets.

Model	PTB		CTB	
	UAS	LAS	UAS	LAS
BiAF	95.88	94.25	85.43	82.79
AOI	**96.07**	**94.42**	**85.76**	**83.08**
BiAF (w/ BERT)	96.62	94.97	90.62	88.62
AOI (w/ BERT)	**96.79**	**95.15**	**90.75**	**88.81**

Table 7: Comparison of results on syntactic dependency parsing datasets.

Multilingual SemDP As Table 6, AOI still shows salient performance improvement on multilingual SemDP as it outperforms the baseline BiAF model by 0.9 F1 score on the Chinese PAS-ID dataset. On average, AOI remarkably leads to 0.67 F1 score improvement from the baseline. With the incorporation of multilingual BERT, the performance of parsers gets improved, and AOI still outperforms the baseline by keeping a gap of 0.50 F1 score on average.

4.5 Syntactic Parsing Result

To illustrate the cross-task effectiveness of our proposed AOI scorer, we also conducted experiments on syntactic parsing. Due to the difference in task between syntactic parsing and semantic dependency parsing, the advantages of AOI over BiAF will no longer be obvious. Therefore, the comparison of other tasks mainly illustrates the lower limit of the performance of our scorer under the situation without special data features.

Syntactic Dependency Parsing SynDP is a task that is similar to SemDP, but it is relatively simpler. Since in the task definition, a dependent has only one head, therefore does not require as much reasoning as in SemDP. In the evaluation of SynDP, the results of each model are shown in Table 7. The comparison shows that our AOI scorer still outperforms the BiAF baseline on the SynDP task, while the improvement is not as significant as on the SemDP task. Because the task is relatively simple and BiAF is strong enough for it, the baseline performs exceptionally well. As a result, compared to BiAF, our AOI method is not only comparable but also outperforms it in PTB and CTB, demonstrating that our AOI is a general parsing scorer.

Syntactic Constituency Parsing Although SynCP is not a head-dependent pair classification task in a narrow sense, and its span division scoring can be modeled as a pair classification task on the left and right boundaries of the span. Therefore the BiAF and AOI pair scorers can be employed as well. In the SynCP task, our AOI produced fairly similar results as BiAF, confirming that our AOI and BiAF scorers perform similarly in general parsing tasks. When parsing tasks like SemDP require more global reasoning, AOI can provide a significant performance boost.

Generally speaking, AOI boosts performance more on SemDP tasks. This can be explained by comparison between Table 3 and 4 in which more unbalance exists in edge distributions of semantic

Model	PTB			CTB		
	LP	LR	LF1	LP	LR	LF1
BiAF	94.18	93.96	94.07	88.77	88.92	88.85
AOI	94.25	94.16	**94.20**	89.44	89.16	**89.29**
BiAF (w/ BERT)	95.67	95.29	95.48	92.13	91.94	92.03
AOI (w/ BERT)	95.75	95.47	**95.61**	92.46	92.27	**92.36**

Table 8: Comparison of results on constituency parsing datasets.

parsing graphs. Thus, there are more edges for the rectification of the MHGAttn on candidates, which results in a better parsing graph produced.

4.6 How about directly using POS for scaling?

Other than AOI, another choice is to learn part-of-speech-based weights to scale the attention on different positions of the parsing graph. We add such an attention scorer to BiAF and find the results not comparable to AOI's (81.84 v.s. 82.27 F1 on PSD-ID and 94.31 v.s. 94.42 LAS on PTB). This can be attributed to the fact that unbalance is more complex than just POS-to-label and should be learned by more carefully designed structures. Still, adding such a modifier will benefit the training of the parser as the results are higher than the initial BiAF.

5 Further Analysis

5.1 Ablation Study

We conduct the ablation study on PSD-ID dataset for the SemDP task. Removing the MHGAttn Scorer results in a drop of 0.55 F1 score (81.72) and using only one attention head leads to a drop of 0.18 F1 score (82.09). These results verify the contributions of attention on candidates and the multi-head implementation of it.

5.2 Performance v.s. Complexity

Sentence Length We explore the robustness of our model by comparing its performance with the baseline BiAF model on sentences of different lengths. Intuitively, a longer sentence implicates higher complexity and makes it harder for the parser to parse. AOI shows strong robustness when parsing sentences with ordinary length, that is, fewer than 30 words. Also, AOI outperforms BiAF on both extremely long and rather short sentences, verifying the general performance improvement from our proposed AOI scorer.

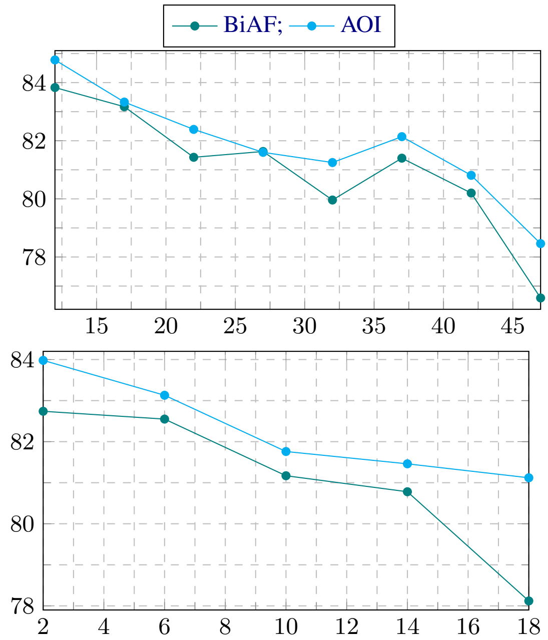

Figure 3: Model Performance vs. Sentence Length (Upper) & Dependency Head (Lower) on SemEval 2015 PSD-ID dataset.

Number of Dependency Head Our AOI model shows high robustness when dependency heads in the sentence increase. AOI keeps a gap with the baseline BiAF on performance when parsing sentences of the different number of dependency heads. Moreover, while BiAF will degrade on sentences with more than 18 heads, our AOI still keeps a strong performance on those sentences.

5.3 Case Study

Here we use a case study to show how our AOI scorer produces a better result than BiAF by taking advantage of unbalanced dependency distribution. We take edge building as an example, as shown in Table 4. In the left figure, the BiAF parses each component in the sentence equally. Thus it has missed the dependency edge from *deny* to *that*.

AOI instead assigns global attention to components. With multiple head attention, AOI chooses *Brokers*, *do* and *deny* as candidates for heads and *Brokers*, *n't* and *that* for dependents. Thereby, the AOI scorer can be more focused on assigning scores to the edges with a higher existing probability between those candidates. As a result, the AOI scorer is more capable of building edges between components and has built all dependency edges correctly as in the case above. Also, we can see the global attention for heads is concentrated on nouns (*Broker*) and verbs (*do*, *deny*), which proves

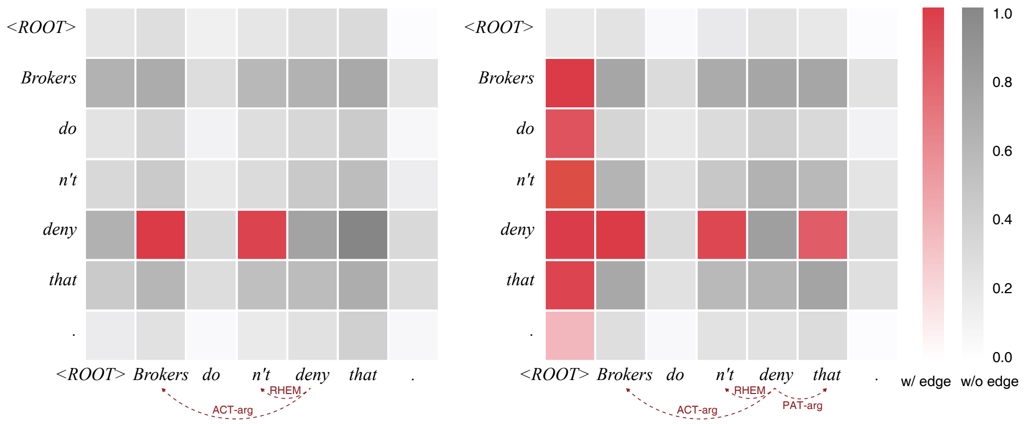

Figure 4: A case study. Left is the parsing result of BiAF and right is the parsing result of AOI. Deeper color refers to higher global attention (AMH attention) score.

the ability of our scorer to be concerned about and leverage the unbalanced dependency distribution of data.

6 Related Work

Dependency parsers aim to build directional dependency edges between components in a sentence. Transition-based parsers (Wang et al., 2016, 2018; Fernández-González and Gómez-Rodríguez, 2020) maintained a stack and relied on the stack and context to choose actions like edge building to complete parsing. Graph-based parsers do this by scoring edge and label graphs of the sentence. Early graph-based parsers (Kiperwasser and Goldberg, 2016; Hashimoto et al., 2016) simply applies feed forward and recurrent neural network to score dependencies for building and labeling edges. The introduction of BiAF (Dozat and Manning, 2017, 2018; Zhang et al., 2020) significantly boosts the efficiency and performance of graph parsers on a variety of graph parsing tasks. High efficiency and performance of graph-based parsers even make some transition-based parsers (Fernández-González and Gómez-Rodríguez, 2020) use graph scorers to improve the prediction of transition actions.

Unbalance exists in parsing graphs at word-level and edge-level. To leverage these unbalance, CRFs (Jia et al., 2020a) and second-order mechanisms (Jia et al., 2020b; Wang et al., 2019) have been proposed to improve parsing performance. These works concentrate on relationships among edges while we aim to exploit word-edge correlations. We study unbalanced distributions related to part-of-speeches and build a parser with better performance.

The attention mechanism is widely used in the deep learning field. In computer vision, attention scoring is commonly used for models like SENet (Hu et al., 2017) and CBAM (Woo et al., 2018). The attention mechanism has also been successfully applied to NLP models including sequence-to-sequence with attention (Bahdanau et al., 2015) and self-attention mechanism-based models like Transformer (Vaswani et al., 2017).

First proposed in the transformer structure (Vaswani et al., 2017) for neural machine translation, multi-head attention has drawn much attention from the whole NLP community so far. Multi-head attention can be applied for better generative models for language models (Guo et al., 2019; Sarkhel et al., 2020), and more precise understanding (Cheng et al., 2021; Jin et al., 2020; Kumar et al., 2020). Moreover, the contribution from multi-head attention has been carefully researched (Ampomah et al., 2020; Voita et al., 2019). For parsing, Li et al. (2019) used Transformer as an encoder for dependency parsing. Though multi-head attention is introduced initially as the self-attention between words, we develop this mechanism into global attention for scoring dependency edges.

7 Conclusion

In this paper, we elaborate on the unbalanced subgraph distribution issue in graph parsing. To mitigate the word-level unbalance, we propose a novel attention scorer AOI which applies accumulative attention to approximate the unbalance. Parsing on a wide variety of graph parsing tasks verifies the performance of AOI enriched parsers to be generally higher than conventional graph parsers.

References

Mariana S. C. Almeida and André F. T. Martins. 2015. Lisbon: Evaluating turbosemanticparser on multiple languages and out-of-domain data. In *Proceedings of the 9th International Workshop on Semantic Evaluation, SemEval@NAACL-HLT 2015, Denver, Colorado, USA, June 4-5, 2015*, pages 970–973. The Association for Computer Linguistics.

Isaac K. E. Ampomah, Sally I. McClean, Zhiwei Lin, and Glenn I. Hawe. 2020. Every layer counts: Multi-layer multi-head attention for neural machine translation. *Prague Bull. Math. Linguistics*, 115:51–82.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Yan Cheng, Huan Sun, Haomai Chen, Meng Li, Yingying Cai, Zhuang Cai, and Jing Huang. 2021. Sentiment analysis using multi-head attention capsules with multi-channel cnn and bidirectional gru. *IEEE Access*, 9:60383–60395.

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.

Timothy Dozat and Christopher D. Manning. 2018. Simpler but more accurate semantic dependency parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 2: Short Papers*, pages 484–490. Association for Computational Linguistics.

Yantao Du, Fan Zhang, Xun Zhang, Weiwei Sun, and Xiaojun Wan. 2015. Peking: Building semantic dependency graphs with a hybrid parser. In *Proceedings of the 9th International Workshop on Semantic Evaluation, SemEval@NAACL-HLT 2015, Denver, Colorado, USA, June 4-5, 2015*, pages 927–931. The Association for Computer Linguistics.

Daniel Fernández-González and Carlos Gómez-Rodríguez. 2020. Transition-based semantic dependency parsing with pointer networks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7035–7046, Online. Association for Computational Linguistics.

Qian Guo, Jifeng Huang, Naixue Xiong, and Pan Wang. 2019. Ms-pointer network: Abstractive text summary based on multi-head self-attention. *IEEE Access*, 7:138603–138613.

Kazuma Hashimoto, Caiming Xiong, Yoshimasa Tsuruoka, and Richard Socher. 2016. A joint many-task model: Growing a neural network for multiple NLP tasks. *CoRR*, abs/1611.01587.

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

Jie Hu, Li Shen, and Gang Sun. 2017. Squeeze-and-excitation networks. *CoRR*, abs/1709.01507.

Zixia Jia, Youmi Ma, Jiong Cai, and Kewei Tu. 2020a. Semi-supervised semantic dependency parsing using CRF autoencoders. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 6795–6805. Association for Computational Linguistics.

Zixia Jia, Youmi Ma, Jiong Cai, and Kewei Tu. 2020b. Semi-supervised semantic dependency parsing using CRF autoencoders. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6795–6805, Online. Association for Computational Linguistics.

Yanliang Jin, Chenjun Tang, Qianhong Liu, and Yan Wang. 2020. Multi-head self-attention-based deep clustering for single-channel speech separation. *IEEE Access*, 8:100013–100021.

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*.

Eliyahu Kiperwasser and Yoav Goldberg. 2016. Simple and accurate dependency parsing using bidirectional LSTM feature representations. *Trans. Assoc. Comput. Linguistics*, 4:313–327.

Avinash Kumar, Vishnu Teja Narapareddy, Veerubhotla Aditya Srikanth, Aruna Malapati, and Lalita Bhanu Murthy Neti. 2020. Sarcasm detection using multi-head attention based bidirectional lstm. *IEEE Access*, 8:6388–6397.

Ying Li, Zhenghua Li, Min Zhang, Rui Wang, Sheng Li, and Luo Si. 2019. Self-attentive biaffine dependency parsing. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 5067–5073. ijcai.org.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of english: The penn treebank. *Comput. Linguistics*, 19(2):313–330.

Stephan Oepen, Marco Kuhlmann, Yusuke Miyao, Daniel Zeman, Silvie Cinková, Dan Flickinger, Jan Hajic, and Zdenka Uresová. 2015. Semeval 2015 task 18: Broad-coverage semantic dependency parsing. In *Proceedings of the 9th International Workshop on Semantic Evaluation, SemEval@NAACL-HLT 2015, Denver, Colorado, USA, June 4-5, 2015*, pages 915–926. The Association for Computer Linguistics.

Hao Peng, Sam Thomson, and Noah A. Smith. 2017. Deep multitask learning for semantic dependency parsing. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 2037–2048. Association for Computational Linguistics.

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.

Ritesh Sarkhel, Moniba Keymanesh, Arnab Nandi, and Srinivasan Parthasarathy. 2020. Interpretable multi-headed attention for abstractive summarization at controllable lengths. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6871–6882, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. 2019. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. 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 5797–5808. 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 Annual Meeting of the Association for Computational Linguistics*, pages 4609–4618, Florence, Italy. Association for Computational Linguistics.

Yuxuan Wang, Wanxiang Che, Jiang Guo, and Ting Liu. 2018. A neural transition-based approach for semantic dependency graph parsing. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 5561–5568. AAAI Press.

Yuxuan Wang, Jiang Guo, Wanxiang Che, and Ting Liu. 2016. Transition-based chinese semantic dependency graph parsing. In *Chinese Computational Linguistics and Natural Language Processing Based on Naturally Annotated Big Data - 15th China National Conference, CCL 2016, and 4th International Symposium, NLP-NABD 2016, Yantai, China, October 15-16, 2016, Proceedings*, volume 10035 of *Lecture Notes in Computer Science*, pages 12–24.

Sanghyun Woo, Jongchan Park, Joon-Young Lee, and In So Kweon. 2018. CBAM: convolutional block attention module. In *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part VII*, volume 11211 of *Lecture Notes in Computer Science*, pages 3–19. Springer.

Yu Zhang, Houquan Zhou, and Zhenghua Li. 2020. Fast and accurate neural CRF constituency parsing. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pages 4046–4053. ijcai.org.

A Configuration

Embed	**Embedding Dimension**
Word Embed	100
Char	50
POS	100
Lemma	100
BERT	100
MLPs&BiLSTMs	**Embedding Dimension**
BiLSTMs	400×2
Edge MLPs	500
Label MLPs	160
AOI	**Value**
Edge Dimension	500
Label Dimension	160
Edge Head	$2/4$
Label Head	$2/4$
Dropout	**Probability**
Embed	0.33
MLPs	0.33
LSTMs	0.33
Optimizer	**Value**
Learning Rate	0.002
Adam μ	0.9
Adam ν	0.9
Batch Size	5000
Decay Rate	0.75
Decay Step	5000

Table 9: Full configuration of the AOI model