Syntax-Aware Graph-to-Graph Transformer for Semantic Role Labelling

Alireza Mohammadshahi
Idiap Research Institute & EPFL

James Henderson
Idiap Research Institute
{alireza.mohammadshahi, james.henderson}@idiap.ch

Abstract

Recent models have shown that incorporating syntactic knowledge into the semantic role labelling (SRL) task leads to a significant improvement. In this paper, we propose Syntax-aware Graph-to-Graph Transformer (SynG2G-Tr) model, which encodes the syntactic structure using a novel way to input graph relations as embeddings, directly into the self-attention mechanism of Transformer. This approach adds a soft bias towards attention patterns that follow the syntactic structure but also allows the model to use this information to learn alternative patterns. We evaluate our model on both span-based and dependency-based SRL datasets, and outperform previous alternative methods in both in-domain and out-of-domain settings, on CoNLL 2005 and CoNLL 2009 datasets.[1]

1 Introduction

The task of semantic role labelling (SRL) provides a shallow representation of the semantics in a sentence, and constructs event properties and relations among relevant words. Traditionally, a syntactic structure was considered a prerequisite for SRL models (Punyakanok et al., 2008; Gildea and Palmer, 2002), but, newer models that leverage deep neural network architectures (Cai et al., 2018; Tan et al., 2017; He et al., 2017; Marcheggiani et al., 2017) have outperformed syntax-aware architectures, without the need for explicit encoding of syntactic structure. However, recent studies (Zhou et al., 2020a; Strubell et al., 2018; He et al., 2017; Marcheggiani and Titov, 2017) have proposed that deep neural network models could benefit from using syntactic information, rather than disregarding it. These studies suggest that incorporating syntax into the model can improve SRL prediction by jointly learning both syntactic and semantic structures (Zhou et al., 2020a), training a self-attention head in Transformer (Vaswani et al., 2017a) to attend to each token's syntactic parent (Strubell et al., 2018), or encoding the syntactic structure using graph convolutional networks (Fei et al., 2021; Marcheggiani and Titov, 2017).

In this paper, we propose a novel method for encoding syntactic knowledge by introducing Syntax-aware Graph-to-Graph Transformer (SynG2G-Tr) architecture. The model conditions on the sentence's dependency structure and jointly predicts both span-based and dependency-based SRL structures. Inspired by Mohammadshahi and Henderson (2021, 2020), our model inputs graph relations as embeddings incorporated into the self-attention mechanism of Transformer (Vaswani et al., 2017b). Different from the original Graph-to-Graph Transformer, our self-attention function models the interaction of the graph relations with both the query and key vectors of self-attention mechanism, instead of just the query. We also find that excluding the interaction of graph structure with the value vectors of self-attention does not harm the performance. Furthermore, compared to the previous work on Graph-to-Graph Transformers (Mohammadshahi and Henderson, 2021, 2020), our architecture uses different types of graphs as the input and output. We show empirically that our model outperforms previous comparable models. In the in-domain setting, SynG2G-Tr model achieves 88.93 (87.57) F1 score on the CoNLL 2005 dataset, given the predicate (end-to-end), and 91.23 (88.05) F1 on the CoNLL 2009 dataset, given the predicate (end-to-end). In the out-of-domain setting, our model reaches 83.21 (80.53) F1 score on the CoNLL 2005 dataset, given predicate (end-to-end), and 86.43 (81.93) F1 scores on the CoNLL 2009 dataset, given predicate (end-to-end).

Our contributions are:

- We propose SynG2G-Tr model for encoding the dependency parsing graph in the SRL task.

[1]The implementation is publicly available at https://github.com/alirezamshi/SynG2GTr-SRL.

Proceedings of the 8th Workshop on Representation Learning for NLP (RepL4NLP 2023), pages 174–186
July 13, 2023

Section	UAS	LAS	PoS
Development	96.72	94.83	96.81
Test	96.85	95.24	97.41

Table 1: Labelled and unlabelled attachment scores (LAS/UAS) and PoS accuracy. Sections 22&23 of WSJ Penn Treebanks (Marcus et al., 1993) are used as evaluation and test sets.

- We evaluate our model on CoNLL 2005 and CoNLL 2009 datasets and outperform previous comparable models in most cases of both in-domain and out-of-domain sets.

2 Syntax-aware Graph-to-Graph Transformer

The architecture of the SynG2G-Tr model is illustrated in Figure 1. The input to the model is the tokenised text ($W = (w_1, w_2, ..., w_N)$), which are the nodes of the input and output graphs, and N is the length of tokenised input. The outputs are the dependency-based (G_{dep}) and span-based (G_{span}) SRL graphs. The SynG2G-Tr model can be formalised in terms of an encoder E^{sg2g} and decoder D^{sg2g}:

$$\begin{cases} Z = \mathrm{E}^{\mathrm{sg2g}}(W, P, G_{syn}) \\ G_{span}, G_{dep} = \mathrm{D}^{\mathrm{sg2g}}(Z) \end{cases} \tag{1}$$

Initially, a syntactic parser predicts the dependency graph (G_{syn}), and Part-of-Speech (PoS) tags ($P = (p_1, p_2, ..., p_N)$). Then the encoder of SynG2G-Tr (E^{sg2g}) encodes both sequences (W, P) and the dependency graph (G_{syn}) into contextualised representations of graph nodes (Z). This representation (Z) is then used by the decoder (D^{sg2g}) to jointly predict SRL graphs. For the decoder, we follow the same unified scorer and decoder as defined in Zhou et al. (2020a). Further explanation of SRL scorer and decoding mechanism is provided in Appendix A.

The encoder employs an enhanced way of inputting graph relations into the self-attention mechanism of Transformer (Vaswani et al., 2017b). Unlike the previously proposed version of Graph-to-Graph Transformer (Mohammadshahi and Henderson, 2021), we modify the self-attention mechanism to have a more comprehensive interaction between graph relations, queries and keys. We also find that excluding the interaction of graph relations with value vectors retains good performance.

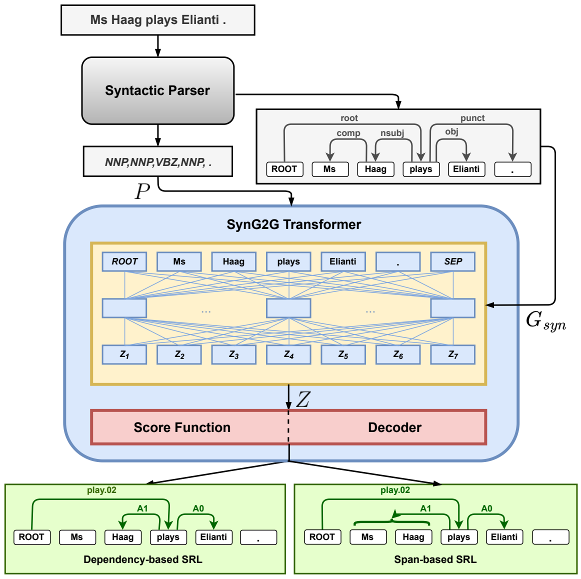


Figure 1: The architecture of SynG2G-Tr.

Specifically, given the output of an intermediate embedding layer $X = (x_1, ..., x_N)$, we define the attention mechanism of each head in each layer to take the dependency graph as input. These attention scores (α_{ij}) are calculated as a Softmax function over e_{ij} values:

$$e_{ij} = \frac{1}{\sqrt{d}} \Big[x_i \boldsymbol{W^Q} (x_j \boldsymbol{W^K})^T + x_i \boldsymbol{W^Q} (r_{ij} \boldsymbol{W^R})^T + r_{ij} \boldsymbol{W^R} (x_j \boldsymbol{W^K})^T \Big] \tag{2}$$

where $\boldsymbol{W^Q}, \boldsymbol{W^K} \in \mathbb{R}^{d_x \times d}$ are learned query and key matrices. $r_{ij} \in R$ is a one-hot vector specifying both the label and direction of the dependency relation between token i and token j. R is the matrix of graph relations, derived from the syntactic graph (G_{syn}). Figure 2 illustrates a sample computation of R matrix, where $r_{ij} = id_{label}$ if $i \rightarrow j$, $id_{label} + |L_{syn}|$ if $j \leftarrow i$, or NONE ($|L_{syn}|$ is the size of syntactic label set). $\boldsymbol{W^R} \in \mathbb{R}^{(2|L_{syn}|+1) \times d}$ is a matrix of learned relation embeddings. d is the attention head size, and d_x is the hidden size.

The second and third terms in Equation 2 incorporate the graph information into the self-attention mechanism of Transformer with a soft bias, while the model can still learn other structures, using this encoded graph information. For better efficiency, we share the relation embeddings across multiple attention heads in each layer. Additionally, the computation complexity of both the second and third terms is $O(N)$, as we ignore the NONE graph relation, and the syntactic dependency graph is a tree. The output of the attention function is the value embedding (v_i), which is calculated as:

$$v_i = \sum_j \alpha_{ij} (x_j \boldsymbol{W^V}) \tag{3}$$

which, in our model, does not use the graph, and $\boldsymbol{W^V} \in \mathbb{R}^{d_x \times d}$ is the learned value matrix.

Syntactic Parser. The parser jointly predicts PoS tags and the dependency graph. We apply the parser defined in Zhou et al. (2020a), which uses a joint scorer and decoder for dependency and constituency graphs based on Head-driven Phrase Structure Grammar (Zhou and Zhao, 2019). This method has achieved state-of-the-art results in the dependency parsing task.

3 Related Work

Several approaches have been proposed to use syntax for the SRL task. Roth and Lapata (2016) embed dependency paths, while some researchers (Fei et al., 2021; Munir et al., 2021; Marcheggiani and Titov, 2017) use graph convolutional networks to encode the syntactic structure. Strubell et al. (2018) incorporates a dependency graph by training one attention head of Transformer to attend to syntactic parents for each token, in a multi-task setting. He et al. (2019, 2018b) use syntactic information to guide the argument pruning. Xia et al. (2019) exploit different alternatives e.g. tree-structured GRU and graph features of dependency tree to encode syntactic knowledge. Kasai et al. (2019) apply BiLSTM to tag the text with supertags extracted from dependency parses and feed them into SRL models. Xia et al. (2020) showed that incorporating heterogeneous syntactic knowledge results in significant improvement. Some other work focus on joint learning of both SRL and syntax (Zhou et al., 2020a,b; Cai and Lapata, 2019a,b). Additionally, some approaches discarded the syntax, but achieve impressive results (Shi and Lin, 2019; Peters et al., 2018; He et al., 2018a; Marcheggiani et al., 2017; He et al., 2017; Tan et al., 2017; Zhou and Xu, 2015). Our work is different from previous work since we encode the syntactic graph by directly inputting it as embeddings into the attention mechanism of Transformer, which provides a soft bias. Moreover, both sequences and syntactic graph can be encoded in one general model.

4 Results and Discussion

Experimental Setup. Our models are evaluated on CoNLL 2005 (Carreras and Màrquez, 2005) and CoNLL 2009 (Hajič et al., 2009).[2] For predicate disambiguation, we follow previous work (Marcheggiani and Titov, 2017), and use an off-the-shelf disambiguator from Roth and Lapata (2016). As in previous work, we evaluate in both *end-to-end*, and *given predicate* settings. For a more accurate comparison, we train SynG2G-Tr both with and without BERT initialisation (SynG2G-Tr w/o BERT). The discrepancy between BERT tokenisation and the tokenisation used in the SRL corpora is handled as in Mohammadshahi and Henderson (2020). [3] For the syntactic parser, we use the same hyper-parameters as defined in Zhou et al. (2020a). The performance of the syntactic parser is shown in Table 1.

Model	SA	WSJ (in-domain)			Brown (out-of-domain)		
		P	R	F1	P	R	F1
end-to-end							
He et al. (2017)	✗	85.0	84.3	84.6	74.9	72.4	73.6
He et al. (2018a)	✗	81.2	83.9	82.5	69.7	71.9	70.8
Strubell et al. (2018)	✓	85.53	84.45	<u>84.99</u>	75.8	73.54	<u>74.66</u>
Li et al. (2019)	✗	-	-	83.0	-	-	-
Xia et al. (2019)	✓	84.3	83.8	84.1	73.7	72.0	72.9
Xia et al. (2020)	✓	83.05	84.49	84.49	73.47	74.92	74.19
SynG2G-Tr (w/o BERT)	✓	84.48	86.46	**85.45**	73.92	76.65	**75.26**
+*pre-training*							
He et al. (2018a)	✗	84.8	87.2	86.0	73.9	78.4	76.1
Strubell et al. (2018)†	✓	87.13	86.67	<u>86.9</u>	79.02	77.49	<u>78.25</u>
Li et al. (2019)	✗	85.2	87.5	86.3	74.7	78.1	76.4
SynG2G-Tr	✓	86.86	88.3	**87.57**	80.01	81.07	**80.53**
given predicate							
Tan et al. (2017)	✗	84.5	85.2	84.8	73.5	74.6	74.1
He et al. (2018a)	✗	-	-	83.9	-	-	73.7
Strubell et al. (2018)†	✓	86.02	86.05	<u>86.04</u>	76.65	76.44	<u>76.54</u>
Ouchi et al. (2018)	✗	84.7	82.3	83.5	76.0	70.4	73.1
Xia et al. (2020)	✓	85.12	85.0	85.06	76.3	75.42	75.86
SynG2G-Tr (w/o BERT)	✓	86.46	86.56	**86.50**	77.73	77.18	**77.45**
+*pre-training*							
He et al. (2018a)	✗	-	-	87.4	-	-	80.4
Ouchi et al. (2018)	✗	88.2	87.0	87.6	79.9	77.5	78.7
Li et al. (2019)	✗	87.9	87.5	87.7	80.6	80.4	80.5
Jindal et al. (2020)	✗	87.70	88.15	87.93	81.52	81.36	81.44
Zhang et al. (2021)	✗	88.70	88.00	87.90	80.30	80.10	80.20
Jia et al. (2022)	✗	-	-	<u>88.25</u>	-	-	<u>81.90</u>
SynG2G-Tr	✓	89.11	88.74	**88.93**	83.93	82.50	**83.21**

Table 2: Comparing our SynG2G-Tr with previous comparable models on CoNLL 2005 test sets. 'SA' means a syntax-aware model. Scores being boldfaced means that they are significantly better than the second best model, specified by the underline marker.

CoNLL 2005 Results.[4] The results for span-based SRL are shown in Table 2.[5] Without BERT

[2]Implementation details of datasets and SynG2G-Tr model are provided in Appendices B and C.

[3]For inputting the dependency graph, the relation between two sub-words of different words is defined as the same dependency relation between their corresponding words in the sentence. This means that the relation (i,j,l_{in}) is repeated for each sub-word of word x_i, and word x_j. The same strategy is applied to predicted PoS tags.

[4]Results are calculated with official evaluation scripts of CoNLL 2005 (`https://www.cs.upc.edu/~srlconll`).

[5]For a fair comparison, we excluded Li et al. (2021); Zhou et al. (2020a), as they use information from the constituency

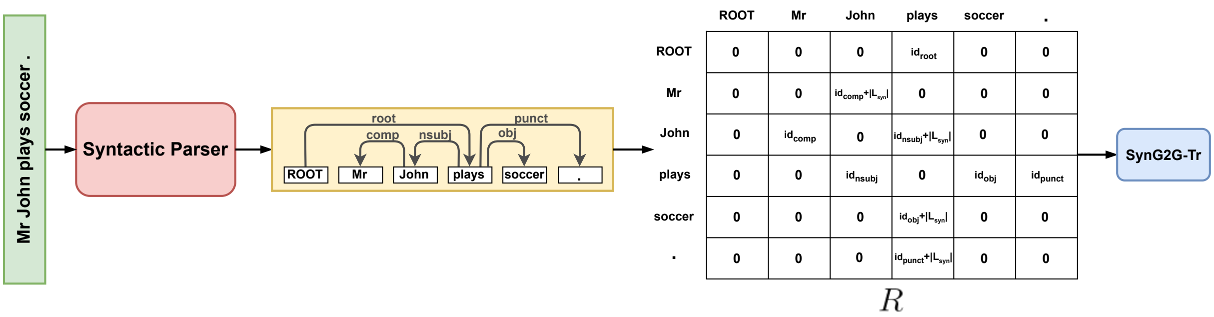


Figure 2: A sample computation of R matrix for the sentence "Mr John plays soccer.".

initialisation, our SynG2G-Tr model outperforms Strubell et al. (2018) (the second best model) in both *end-to-end* and *given-predicate* settings. This highlights the benefit of injecting the graph information into the self-attention mechanism using a soft bias, instead of hard-coding one attention head to attend to the syntactic parent of each token, as used in Strubell et al. (2018). The main reason for this improvement is that the model can still learn other attention patterns in combination with the graph information, which will be described later in this section. When adding BERT initialisation, our SynG2G-Tr model outperforms best previous work by 5.4%/8.8% F1 relative error reduction (RER) on average in both in-domain and out-of-domain evaluation sets, which demonstrates the compatibility of the modified self-attention mechanism of SynG2G-Tr with BERT (Devlin et al., 2019) initialisation.

CoNLL 2009 Results.[6] Table 3 illustrates the results of dependency-based SRL on the test set of CoNLL 2009 dataset. Without BERT initialisation, SynG2G-Tr significantly outperforms previous work in in-domain and out-of-domain settings. With BERT initialisation, our model significantly outperforms previous work in *end-to-end* setting with 3.2%/10.4% F1 RER in both in-domain and out-of-domain evaluation sets, while having competitive performance in *given-predicate* setting. For a better comparison with Fei et al. (2021) (last setting of Table 3), we also employ the gold dependency tree for training and use the predicted dependency graph at inference time. Our model significantly outperforms Fei et al. (2021), especially on the out-of-domain dataset. This shows the benefit of encoding the dependency graph by modifying the self-attention mechanism of Transformer (Vaswani et al., 2017b) compared to using graph convolutional network, as in Fei et al. (2021).

Model	SA	WSJ (in-domain)			Brown (out-of-domain)		
		P	R	F1	P	R	F1
end-to-end							
He et al. (2018b)	✓	83.9	82.7	83.3	-	-	-
Cai et al. (2018)	✗	84.7	85.2	85.0	-	-	72.5
Li et al. (2019)	✗	-	-	85.1	-	-	-
SynG2G-Tr (w/o BERT)	✓	84.10	87.07	**85.59**	73.66	72.56	**73.11**
+pre-training							
Li et al. (2019)	✗	84.5	86.1	85.3	74.6	73.8	74.2
SynG2G-Tr	✓	86.38	89.78	**88.05**	80.35	83.57	**81.93**
given predicate							
Marcheggiani et al. (2017)	✗	88.7	86.8	87.7	79.4	76.2	77.7
M&T(2017)	✓	89.1	86.8	88.0	78.5	75.9	77.2
He et al. (2018b)	✓	89.7	89.3	89.5	81.9	76.9	79.3
Cai et al. (2018)	✗	89.9	89.2	89.6	79.8	78.3	79.0
Cai and Lapata (2019c)	✓	90.5	88.6	89.6	80.5	78.2	79.4
Kasai et al. (2019)	✓	89.0	88.2	88.6	78.0	77.2	77.6
SynG2G-Tr (w/o BERT)	✓	89.78	90.28	**90.03**	81.32	82.15	**81.73**
+pre-training							
Li et al. (2019)	✗	89.6	91.2	90.4	81.7	81.4	81.5
Kasai et al. (2019)	✓	90.3	90.0	90.2	81.0	80.5	80.8
Lyu et al. (2019)	✗	-	-	90.99	-	-	82.18
Chen et al. (2019)	✗	90.74	91.38	**91.06**	82.66	82.78	82.72
He et al. (2019)	✓	90.41	91.32	90.86	86.15	86.70	**86.42**
Cai and Lapata (2019a)	✓	91.1	90.4	90.7	82.1	81.3	81.6
Munir et al. (2021)	✓	91.2	90.6	90.9	83.1	82.6	82.8
SynG2G-Tr	✓	91.31	91.16	**91.23**	86.40	86.47	**86.43**
gold syntax							
Fei et al. (2021)	✓	92.5	92.5	92.5	85.6	85.3	85.4
SynG2G-Tr+Gold	✓	92.71	93.37	**93.03**	88.27	88.31	**88.29**

Table 3: Comparing our SynG2G-Tr with previous comparable models on CoNLL 2009 test sets. 'SA' means a syntax-aware model. Scores being boldfaced means that they are significantly better than the second best model, specified by the underline marker.

Further Analysis. We also analyse the self-attention matrix of SynG2G-Tr model for different heads and layers. Figure 3 in Appendix D demonstrates that the self-attention mechanism of SynG2G-Tr ignores the dependency graph information in the first few layers, and only uses the context-dependent information. However, as it progresses to upper layers, it begins to utilise the graph relation information, as shown in the atten-

graph additional to the dependency tree. Also, to better understand the effect of syntactic information, we exclude Fernández-González (2023); Zhou et al. (2022); Conia and Navigli (2020), as they exploited different scorer and training mechanism for SRL graphs. However, the best setting of SynG2G-Tr model still shows competitive or better results when compared to aforementioned excluded works.

[6]Scores are calculated with CoNLL 2009 shared task script (`https://ufal.mff.cuni.cz/conll2009-st/`).

tion matrix. This highlights the benefit of encoding the dependency graph with a soft bias as the model can still learn different structures in different layers, given this encoded graph information. Furthermore, in Appendix E, we show that removing the interaction of graph embeddings with key vectors results in a performance drop. Additionally, ignoring the interaction of graph relations with both key and query vectors [7] results in a significant drop as well. However, integrating the graph information into Equation 3 as stated in Mohammadshahi and Henderson (2021) does not improve the performance, and we remove it for better efficiency.

5 Conclusion

In this paper, we propose the Syntax-aware Graph-to-Graph Transformer architecture, which effectively incorporates syntactic information by inputting the syntactic dependency graph into the self-attention mechanism of Transformer. Our mechanism for inputting graph relation embeddings differs from the original Graph-to-Graph Transformer in that it models the complete interaction between the dependency relation, query vector and key vector. It also excludes the graph interaction with value vectors while maintaining good performance. We have evaluated our model on CoNLL 2005 and CoNLL 2009 SRL datasets and outperformed previous comparable models. Future studies can apply our model to any NLP task which might benefit from conditioning on the syntactic structure or other graphs.

Limitations

SynG2G-Tr encodes the syntactic dependency graph because the nodes of input and output graphs should be similar. Future work could include investigating the use of constituency graphs in the self-attention mechanism of Transformer (Vaswani et al., 2017b), where the nodes of the input graph (constituency graph) are different from those of the SRL output graph. In this paper, we initialise our model with the pre-trained BERT (Devlin et al., 2019) model. As future study, larger and better pre-trained language models will be used for the initialisation of SynG2G-Tr models, to achieve better performance. Additionally, future studies can easily extend our work to multilingual SRL benchmarks.

[7] This leads to a BERT-based syntax-agnostic model, similar to Shi and Lin (2019).

Acknowledgement

We are grateful to the Swiss National Science Foundation (SNSF), grant CRSII5-180320, for funding this work. We also thank members of the IDIAP NLU group for helpful discussions and suggestions. We are grateful to anonymous reviewers for their fruitful comments and corrections.

References

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. 2016. Layer normalization.

Jiaxun Cai, Shexia He, Zuchao Li, and Hai Zhao. 2018. A full end-to-end semantic role labeler, syntactic-agnostic over syntactic-aware? In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2753–2765, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Rui Cai and Mirella Lapata. 2019a. Semi-supervised semantic role labeling with cross-view training. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1018–1027, Hong Kong, China. Association for Computational Linguistics.

Rui Cai and Mirella Lapata. 2019b. Syntax-aware Semantic Role Labeling without Parsing. *Transactions of the Association for Computational Linguistics*, 7:343–356.

Rui Cai and Mirella Lapata. 2019c. Syntax-aware semantic role labeling without parsing. *Transactions of the Association for Computational Linguistics*, 7:343–356.

Xavier Carreras and Lluís Màrquez. 2005. Introduction to the CoNLL-2005 shared task: Semantic role labeling. In *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*, pages 152–164, Ann Arbor, Michigan. Association for Computational Linguistics.

Xinchi Chen, Chunchuan Lyu, and Ivan Titov. 2019. Capturing argument interaction in semantic role labeling with capsule networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5415–5425, Hong Kong, China. Association for Computational Linguistics.

Simone Conia and Roberto Navigli. 2020. Bridging the gap in multilingual semantic role labeling: a language-agnostic approach. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1396–1410, Barcelona, Spain (Online). International Committee on 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, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Luis Espinosa Anke, Alexander Shvets, Alireza Mohammadshahi, James Henderson, and Leo Wanner. 2022. Multilingual extraction and categorization of lexical collocations with graph-aware transformers. In *Proceedings of the 11th Joint Conference on Lexical and Computational Semantics*, pages 89–100, Seattle, Washington. Association for Computational Linguistics.

Hao Fei, Fei Li, Bobo Li, and Donghong Ji. 2021. Encoder-decoder based unified semantic role labeling with label-aware syntax. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(14):12794–12802.

Daniel Fernández-González. 2023. Transition-based semantic role labeling with pointer networks. *Knowledge-Based Systems*, 260:110127.

Daniel Gildea and Martha Palmer. 2002. The necessity of parsing for predicate argument recognition. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 239–246, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antònia Martí, Lluís Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, Pavel Straňák, Mihai Surdeanu, Nianwen Xue, and Yi Zhang. 2009. The CoNLL-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL 2009): Shared Task*, pages 1–18, Boulder, Colorado. Association for Computational Linguistics.

Luheng He, Kenton Lee, Omer Levy, and Luke Zettlemoyer. 2018a. Jointly predicting predicates and arguments in neural semantic role labeling. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 364–369, Melbourne, Australia. Association for Computational Linguistics.

Luheng He, Kenton Lee, Mike Lewis, and Luke Zettlemoyer. 2017. Deep semantic role labeling: What works and what's next. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 473–483, Vancouver, Canada. Association for Computational Linguistics.

Shexia He, Zuchao Li, and Hai Zhao. 2019. Syntax-aware multilingual semantic role labeling. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5350–5359, Hong Kong, China. Association for Computational Linguistics.

Shexia He, Zuchao Li, Hai Zhao, and Hongxiao Bai. 2018b. Syntax for semantic role labeling, to be, or not to be. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2061–2071, Melbourne, Australia. Association for Computational Linguistics.

Zixia Jia, Zhaohui Yan, Haoyi Wu, and Kewei Tu. 2022. Span-based semantic role labeling with argument pruning and second-order inference. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(10):10822–10830.

Ishan Jindal, Ranit Aharonov, Siddhartha Brahma, Huaiyu Zhu, and Yunyao Li. 2020. Improved semantic role labeling using parameterized neighborhood memory adaptation. *ArXiv*, abs/2011.14459.

Jungo Kasai, Dan Friedman, Robert Frank, Dragomir Radev, and Owen Rambow. 2019. Syntax-aware neural semantic role labeling with supertags. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 701–709, Minneapolis, Minnesota. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization.

Zuchao Li, Shexia He, Hai Zhao, Yiqing Zhang, Zhuosheng Zhang, Xi Zhou, and Xiang Zhou. 2019. Dependency or span, end-to-end uniform semantic role labeling.

Zuchao Li, Hai Zhao, Shexia He, and Jiaxun Cai. 2021. Syntax role for neural semantic role labeling. *Computational Linguistics*, 47(3):529–574.

Chunchuan Lyu, Shay B. Cohen, and Ivan Titov. 2019. Semantic role labeling with iterative structure refinement. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1071–1082, Hong Kong, China. Association for Computational Linguistics.

Diego Marcheggiani, Anton Frolov, and Ivan Titov. 2017. A simple and accurate syntax-agnostic neural model for dependency-based semantic role labeling. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 411–420, Vancouver, Canada. Association for Computational Linguistics.

Diego Marcheggiani and Ivan Titov. 2017. Encoding sentences with graph convolutional networks for semantic role labeling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1506–1515, Copenhagen, Denmark. Association for Computational Linguistics.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Alireza Mohammadshahi and James Henderson. 2020. Graph-to-graph transformer for transition-based dependency parsing. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3278–3289, Online. Association for Computational Linguistics.

Alireza Mohammadshahi and James Henderson. 2021. Recursive Non-Autoregressive Graph-to-Graph Transformer for Dependency Parsing with Iterative Refinement. *Transactions of the Association for Computational Linguistics*, 9:120–138.

Alireza Mohammadshahi, Rémi Lebret, and Karl Aberer. 2019. Aligning multilingual word embeddings for cross-modal retrieval task. In *Proceedings of the Beyond Vision and LANguage: inTEgrating Real-world kNowledge (LANTERN)*, pages 11–17, Hong Kong, China. Association for Computational Linguistics.

Alireza Mohammadshahi, Vassilina Nikoulina, Alexandre Berard, Caroline Brun, James Henderson, and Laurent Besacier. 2022a. SMaLL-100: Introducing shallow multilingual machine translation model for low-resource languages. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 8348–8359, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Alireza Mohammadshahi, Thomas Scialom, Majid Yazdani, Pouya Yanki, Angela Fan, James Henderson, and Marzieh Saeidi. 2022b. Rquge: Reference-free metric for evaluating question generation by answering the question.

Kashif Munir, Hai Zhao, and Zuchao Li. 2021. Adaptive convolution for semantic role labeling. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:782–791.

Vinod Nair and Geoffrey E. Hinton. 2010. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, page 807–814, Madison, WI, USA. Omnipress.

Hiroki Ouchi, Hiroyuki Shindo, and Yuji Matsumoto. 2018. A span selection model for semantic role labeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1630–1642, Brussels, Belgium. Association for Computational Linguistics.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Vasin Punyakanok, Dan Roth, and Wen-tau Yih. 2008. The importance of syntactic parsing and inference in semantic role labeling. *Computational Linguistics*, 34(2):257–287.

Michael Roth and Mirella Lapata. 2016. Neural semantic role labeling with dependency path embeddings. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1192–1202, Berlin, Germany. Association for Computational Linguistics.

Peng Shi and Jimmy Lin. 2019. Simple bert models for relation extraction and semantic role labeling.

Emma Strubell, Patrick Verga, Daniel Andor, David Weiss, and Andrew McCallum. 2018. Linguistically-informed self-attention for semantic role labeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5027–5038, Brussels, Belgium. Association for Computational Linguistics.

Zhixing Tan, Mingxuan Wang, Jun Xie, Yidong Chen, and Xiaodong Shi. 2017. Deep semantic role labeling with self-attention.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017a. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017b. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Qingrong Xia, Zhenghua Li, Min Zhang, Meishan Zhang, Guohong Fu, Rui Wang, and Luo Si. 2019. Syntax-aware neural semantic role labeling. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):7305–7313.

Qingrong Xia, Rui Wang, Zhenghua Li, Yue Zhang, and Min Zhang. 2020. Semantic role labeling with heterogeneous syntactic knowledge. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2979–2990, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Zhisong Zhang, Emma Strubell, and Eduard H. Hovy. 2021. Comparing span extraction methods for semantic role labeling. In *SPNLP*.

Jie Zhou and Wei Xu. 2015. End-to-end learning of semantic role labeling using recurrent neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1127–1137, Beijing, China. Association for Computational Linguistics.

Junru Zhou, Zuchao Li, and Hai Zhao. 2020a. Parsing all: Syntax and semantics, dependencies and spans. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4438–4449, Online. Association for Computational Linguistics.

Junru Zhou, Zhuosheng Zhang, Hai Zhao, and Shuailiang Zhang. 2020b. LIMIT-BERT : Linguistics informed multi-task BERT. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4450–4461, Online. Association for Computational Linguistics.

Junru Zhou and Hai Zhao. 2019. Head-driven phrase structure grammar parsing on Penn treebank. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2396–2408, Florence, Italy. Association for Computational Linguistics.

Shilin Zhou, Qingrong Xia, Zhenghua Li, Yu Zhang, Yu Hong, and Min Zhang. 2022. Fast and accurate end-to-end span-based semantic role labeling as word-based graph parsing. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 4160–4171, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Appendix A SRL Scorer and Decoder Details

Scorer. Inspired by Zhou et al. (2020a), we first define span representation (s_{ij}) as the difference between right and left end-points of the span:

$$s_{ij} = \vec{sr}_j - \overleftarrow{sl}_i \tag{4}$$

where $\vec{sr}_j$ is defined as $[\vec{z_{j+1}}; \vec{z_j}]$, and $\overleftarrow{sl}_i$ is calculated as $[\overleftarrow{z_i}; \overleftarrow{z_{i+1}}]$. $\overleftrightarrow{z}_i$ is computed by dividing the output representation of Transformer (z_i) in half.
Argument (a_{ij}) and predicate (v_k) representations are defined as:

$$\begin{aligned} a_{ij} &= \mathrm{ReLU}(\boldsymbol{W^1_{srl}} s_{ij} + b^1_{srl}) \\ v_k &= z_k \end{aligned} \tag{5}$$

where $\boldsymbol{W^1_{srl}}$ and b^1_{srl} are learned parameters and $\mathrm{ReLU}(.)$ is the Rectified Linear Unit (Nair and Hinton, 2010) function.
We predict semantic roles as defined in Zhou et al. (2020a):

$$\Phi_l(v,a) = \boldsymbol{W^3_{srl}}(\mathrm{LN}(\boldsymbol{W^2_{srl}}[a_{ij}; v_k] + b^2_{srl})) + b^3_{srl} \tag{6}$$

where $\mathrm{LN}(.)$ is the layer normalisation (Ba et al., 2016) function, and $\boldsymbol{W^2_{srl}}$, $\boldsymbol{W^3_{srl}}$, b^2_{srl}, and b^3_{srl} are learned parameters. The semantic role score for a specific label l_{out} is defined as:

$$\Phi_l(v,a,l_{out}) = [\Phi_l(v,a)]_{l_{out}} \tag{7}$$

Since the number of predicate-argument pairs is $O(n^3)$, we apply the pruning method proposed in Li et al. (2019); He et al. (2018a) by defining separate scorers for argument and predicate candidates (Φ_a and Φ_v), and pruning all but the top-ranked arguments and predicates based on their corresponding scores.

Training. The model is trained to optimise the probability $P(\hat{y}|W,P,G_{syn})$ of predicate-argument pairs, conditioned on input sequence (W), PoS tags (P), and predicated dependency graph (G_{syn}). This objective can be factorised as:

$$\begin{aligned} J(\theta) &= \sum_{y\in\Gamma} -log P_\theta(y|W,P,G_{syn}) \\ &= \sum_{\langle v,a,l_{out}\rangle \in \Gamma} -log \frac{\exp(\Phi(v,a,l_{out}))}{\sum_{\hat{l}\in L_{srl}} \exp(\Phi(v,a,\hat{l}))} \end{aligned} \tag{8}$$

where $\Phi(v,a,l_{out})$ is defined as $\Phi_v(v) + \Phi_a(a) + \Phi_l(v,a,l_{out})$, and θ is model parameters. Γ is the set of predicate-argument-relation tuples for all possible predicate-argument pairs and either the correct relation or NONE.

Decoders. Following Zhou et al. (2020a), we apply a single dynamic programming decoder according to the uniform score following the non-overlapping constraints (Punyakanok et al., 2008).

Appendix B Implementation Details and Pre-processing Steps

CoNLL 2005: In this shared task (Carreras and Màrquez, 2005) (under LDC license), the focus was on verbal predicates in English. The training data includes sections 2-21 of the Wall Street Journal (WSJ) dataset. Section 24 is considered as the development set, while section 23 is used as the in-domain test set. Three sections of the Brown corpus are used for the out-of-domain dataset. The dataset can be downloaded from here, and pre-processing steps are provided in here.

CoNLL 2009: This shared task (Hajič et al., 2009) (under LDC license) focused on dependency-based SRL and was created by merging PropBank and NomBank treebanks. We evaluate our models on the English dataset with the same split as the CoNLL 2005 dataset. The dataset and pre-processing steps can be found at here and here. The number of sentences in train and evaluation sets is as follows:

Train	Dev	Test-WSJ	Test-Brown
39'832	1'334	2'399	425

Table 4: The number of sentences for each split of CoNLL 2005 and CoNLL 2009 datasets.

Appendix C Hyper-parameters Setting

We use *bert-large-whole-word-masking*[8] (345M parameters) for the initialisation of encoder in SynG2G-Tr model. We use Adam optimiser (Kingma and Ba, 2014) and apply separate optimisers for pre-trained parameters and randomly initialised ones. We use bucket batching, grouping sentences by their lengths to the same batch to speed up the model. Early stopping is used to mitigate over-fitting, as in previous work (Mohammadshahi et al., 2022b,a, 2019). In a pre-defined predicate setting, we use different dynamic programming decoders to find SRL graphs, since predicates are not necessarily the same in dependency-based and span-based SRL graphs. For choosing the best hyper-parameters, we use manual tuning to find the base learning rate and BERT learning rate. For other hyper-parameters, we follow previous work (Zhou et al., 2020a). The base learning rate is selected from $\{1e-2, 1e-3, 1.5e-3\}$, and the BERT learning rate is chosen from $\{1e-5, 1.5e-5, 2e-5\}$. So, we train our models with 9 different learning rates to find the best performing model based on the summation of F1 scores of span-based and dependency-based SRL graphs. We use NVIDIA GeForce GTX 1080 Ti for training and evaluating our models. [9] For the dependency parser, we apply the same hyper-parameters as Zhou et al. (2020a). We use the base learning rate of $2e-3$, and the BERT learning rate of $1.5e-5$. Here is the list of hyper-parameters for the SynG2G-Tr model:

Component	Specification
Optimiser	Adam
Base Learning rate	1.5e-3
BERT Learning rate	1e-5
Adam Betas (b_1,b_2)	(0.9,0.999)
Adam Epsilon	1e-5
Weight Decay	0.01
Max-Grad-Norm	1
Warm-up	0.001
Self-Attention	
No. Layers	24
No. Heads	16
Embedding size	1024
Max Position Embedding	512

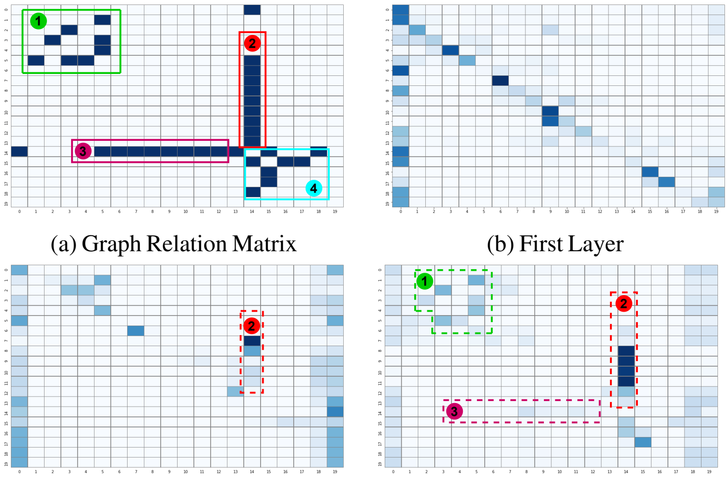
Component	Specification
Feed-Forward layers (SRL)	
Span Hidden size	512
Label Hidden size	250
Feed-Forward layers (PoS)	
Hidden size	250
Pruning (SRL)	
λ_{verb}	0.6
λ_{span}	0.6
Max No. Span	300
Max No. Verb	30
Epoch	100

Table 5: Hyper-parameters for training SynG2G-Tr.

Appendix D Attention Visualisation

Figure 3 shows the attention weights for different layers of self-attention in the SynG2G-Tr model (Figure 3b-3d), alongside the dependency relation matrix (Figure 3a). The self-attention matrix includes four patterns. The first layer of the SynG2G-Tr model (Figure 3b) ignores the graph relations and learns string-local context information. For the middle layer (Figure 3c), attention weights partially use the graph relation pattern. Then, in the last layer (Figure 3d), the dependency graph relations are evident in the attention pattern. This demonstrates the benefit of adding the graph information with a soft bias, allowing the model to learn different structures using both local context and graph information. Furthermore, it can be inferred that the last layers of the self-attention mechanism require a global view and between-edge information, while the first few layers learn local context information. More examples are provided in Figures 4-5-6.

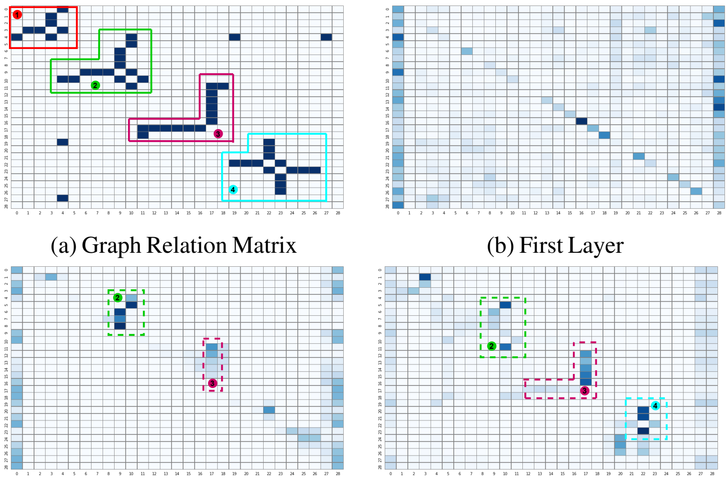
[8] `https://github.com/google-research/bert`. Apache License 2.0.
[9] The training time of SynG2G-Tr model is 0h20m40s, and the evaluation time is 0h02m24s.



(a) Graph Relation Matrix (b) First Layer

(c) Middle Layer (d) Last Layer

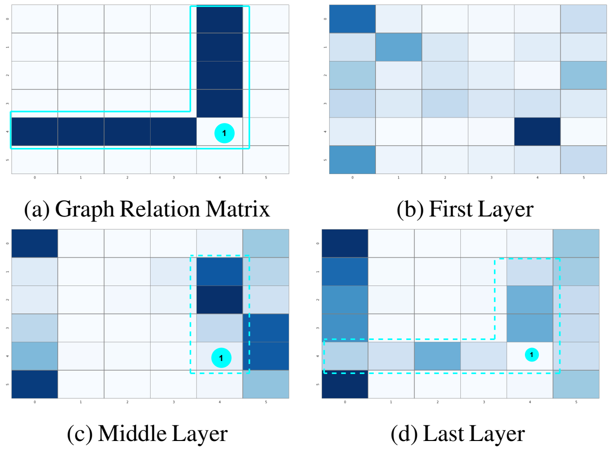
Figure 3: The attention weights for the CoNLL 2009 example "[CLS] The most troublesome report may be the August merchandise trade deficit due out tomorrow . [SEP]". The first figure shows the dependency graph matrix.



(a) Graph Relation Matrix (b) First Layer

(c) Middle Layer (d) Last Layer

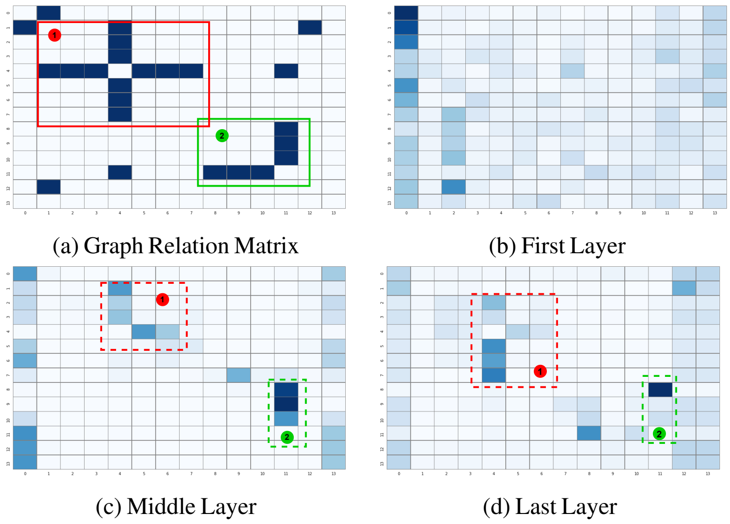
Figure 4: The attention weights for a specific example "[CLS] The consensus view expects a 0.4 % increase in the September CPI after a flat reading in August . [SEP]" on CoNLL 2009 dataset. The first figure shows the dependency graph matrix.



(a) Graph Relation Matrix (b) First Layer

(c) Middle Layer (d) Last Layer

Figure 5: The attention weights for a specific example "[CLS] Candid Comment [SEP]" on CoNLL 2009 dataset. The first figure shows the dependency graph matrix.



(a) Graph Relation Matrix (b) First Layer

(c) Middle Layer (d) Last Layer

Figure 6: The attention weights for a specific example "[CLS] Let 's make that 1929 , just to be sure . [SEP]" on CoNLL 2009 dataset. The first figure shows the dependency graph matrix.

Appendix E Ablation Study

In Table 6, we analyse the interaction of the dependency graph with key and query vectors in the attention mechanism, as defined in Equation 2. Excluding the key interaction results in a similar attention mechanism as defined in Mohammadshahi and Henderson (2021). This SynG2G-Tr-*key* model achieves similar results compared to the SynG2G-Tr model on the WSJ test dataset given the predicate, but the SynG2G-Tr model outperforms it on all other settings, including both types of out-of-domain datasets, confirming that key interaction is a critical part of the SynG2G-Tr model.

When both key and query interactions are excluded from the SynG2G-Tr model (SynG2G-Tr-*key-query*), it has significantly lower performance than the SynG2G-Tr model in all settings. This demonstrates the impact of encoding the graph relation embeddings in the self-attention mechanism of Transformer (Vaswani et al., 2017b) model.

We also evaluate adding the interaction of graph relations with value vectors to the SynG2G-Tr model, as defined in Espinosa Anke et al. (2022); Mohammadshahi and Henderson (2020). The SynG2G-Tr+*value* model achieves similar or worse results compared to the SynG2G-Tr model. So, we exclude this interaction to speed up the modified attention mechanism.

Model	CoNLL 2005			CoNLL 2009		
	Dev	WSJ	Brown	Dev	WSJ	Brown
end-to-end						
SynG2G-Tr -*key* -*query*	86.65	87.08	79.40	86.40	87.26	81.12
SynG2G-Tr -*key*	86.82	87.27	80.33	86.85	87.50	81.51
SynG2G-Tr	87.08	87.57	80.53	87.13	88.05	81.93
SynG2G-Tr +*value*	87.17	87.45	80.40	86.92	87.95	82.03
given predicate						
SynG2G-Tr -*key* -*query*	87.93	88.52	82.56	90.16	90.68	85.72
SynG2G-Tr -*key*	88.03	88.91	82.90	90.31	91.22	86.28
SynG2G-Tr	88.17	88.93	83.21	90.66	91.23	86.43
SynG2G-Tr +*value*	88.15	88.78	83.10	90.48	91.15	86.41

Table 6: Model comparison of SynG2G-Tr and other variants, by F1 score on CoNLL 2005 and CoNLL 2009 datasets. The SynG2G-Tr-*key-query* model is the same as the syntax-agnostic BERT model.