An Empirical Study on Leveraging Position Embeddings for Target-oriented Opinion Words Extraction

Samuel Mensah
Computer Science Department
University of Sheffield, UK
s.mensah@sheffield.ac.uk

Kai Sun
BDBC and SKLSDE
Beihang University, China
sunkai@buaa.edu.cn

Nikolaos Aletras
Computer Science Department
University of Sheffield, UK
n.aletras@sheffield.ac.uk

Abstract

Target-oriented opinion words extraction (TOWE) (Fan et al., 2019b) is a new subtask of target-oriented sentiment analysis that aims to extract opinion words for a given aspect in text. Current state-of-the-art methods leverage position embeddings to capture the relative position of a word to the target. However, the performance of these methods depends on the ability to incorporate this information into word representations. In this paper, we explore a variety of text encoders based on pretrained word embeddings or language models that leverage part-of-speech and position embeddings, aiming to examine the actual contribution of each component in TOWE. We also adapt a graph convolutional network (GCN) to enhance word representations by incorporating syntactic information. Our experimental results demonstrate that BiLSTM-based models can effectively encode position information into word representations while using a GCN only achieves marginal gains. Interestingly, our simple methods outperform several state-of-the-art complex neural structures.

1 Introduction

Target-oriented opinion words extraction (TOWE) (Fan et al., 2019b) is a fine-grained task of target-oriented sentiment analysis (Liu, 2012) aiming to extract opinion words with respect to an opinion target (or aspect) in text. Given the sentence *"The food is good but the service is extremely slow"*, TOWE attempts to identify the opinion words *"good"* and *"extremely slow"* corresponding respectively to the targets *"food"* and *"service"*. TOWE is usually treated as a sequence labeling problem using the BIO tagging scheme (Ramshaw and Marcus, 1999) to distinguish the **B**eginning, **I**nside and **O**utside of a span of opinion words. Table 1 shows an example of applying the BIO tagging scheme for TOWE.

Sentence:
The food is good but the service is extremely slow.
True Labels for target 'food':
The/**O** food/**O** is/**O** good/**B** but/**O** the/**O** service/**O** is/**O** extremely/**O** slow/**O**.
True Labels for target 'service':
The/**O** food/**O** is/**O** good/**O** but/**O** the/**O** service/**O** is/**O** extremely/**B** slow/**I**.
TOWE Extraction Results:
{(food, good), (service, extremely slow)}

Table 1: Identifying target-oriented opinion words in a sentence. Underlined words are opinion targets. Spans tagged **B** and **I** are considered as opinion words.

Learning effective word representations is a critical step towards tackling TOWE. Traditional work (Zhuang et al., 2006a; Hu and Liu, 2004a; Qiu et al., 2011) has used hand-crafted features to represent words which do not often generalize easily. More recent work (Liu et al., 2015; Fan et al., 2019b; Wu et al., 2020a; Veyseh et al., 2020) has explored neural networks to learn word representations automatically.

Previous neural-based methods (Liu et al., 2015; Fan et al., 2019b) has used word embeddings (Collobert and Weston, 2008; Mikolov et al., 2013; Pennington et al., 2014) to represent the input. However, TOWE is a complex task that requires a model to know the relative position of each word to the aspect in text. Words that are relatively closer to the target usually express the sentiment towards that aspect (Zhou et al., 2020).

Fan et al. (2019b) employ Long Short-Term Memory (LSTM) networks (Hochreiter and Schmidhuber, 1997) to encode the target position information in word embeddings. Wu et al. (2020a) transfer latent opinion knowledge into a Bidirectional LSTM (BiLSTM) network that leverages word and position embeddings (Zeng et al., 2014). Recently, Veyseh et al. (2020) have proposed ONG, a method that combines BERT (Bidirectional Encoder Representations from Transformers) (De-

Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pages 9174–9179
November 7–11, 2021.

vlin et al., 2018), position embeddings, Ordered Neurons LSTM (ON-LSTM) (Shen et al., 2018),[1] and a graph convolutional network (GCN) (Kipf and Welling, 2016) to introduce syntactic information into word representations. While this model achieves state-of-the-art results, previous studies have shown that the ON-LSTM does not actually perform much better than LSTMs in recovering latent tree structures (Dyer et al., 2019). Besides, ON-LSTMs perform worse than LSTMs in capturing short-term dependencies (Shen et al., 2018). Since opinion words are usually close to targets in text, ON-LSTM risks missing the relationship between the aspect and any information (e.g. position) relating to the opinion words.

In this paper, we empirically evaluate a battery of popular text encoders which apart from words, take positional and part-of-speech information into account. Surprisingly, we show that methods based on BiLSTMs can effectively leverage position embeddings to achieve competitive if not better results than more complex methods such as ONG on standard TOWE datasets. Interestingly, combining a BiLSTM encoder with a GCN to explicitly capture syntactic information achieves only minor gains. This empirically highlights that BiLSTM-based methods have an inductive bias appropriate for the TOWE task, making a GCN less important.

2 Methodology

Given sentence $s = \{w_1, \ldots, w_n\}$ with aspect $w_t \in s$, our approach consists of a text encoder that takes as input a combination of words, part-of-speech and position information for TOWE. We further explore enhancing text encoding by incorporating information from a syntactic parse of the sentence through a GCN encoder.

2.1 Input Representation

Word Embeddings: We experiment with Glove word vectors (Pennington et al., 2014) as well as BERT-based representations, extracted from the last layer of a BERT base model (Devlin et al., 2018) fine-tuned on TOWE.

Position Embeddings (POSN): We compute the relative distance d_i from w_i to w_t (i.e., $d_i = i - t$), and lookup their embedding in a randomly initialized position embedding table.

Par-of-Speech Tag Embeddings (POST): We assign part-of-speech tags to each word token using the Stanford parser,[2] and lookup their embedding in a randomly initialized POST embedding table.

Combined Input: We consider two types of input representations:

1. **Glove Input (G)**: Constructed from concatenating Glove word embeddings, POST and POSN embeddings for each token.

2. **BERT Input (B)**: Constructed from concatenating BERT vectors with POSN embeddings for each word token following a similar approach as (Veyseh et al., 2020).[3] We ignore POST embeddings since BERT is efficient in modeling such information (Tenney et al., 2019).

2.2 Text Encoders

We experiment with the following neural encoders that take word vector representations as input:

CNN: A single layer convolutional neural network (LeCun et al., 1990). Given a word $w_i \in s$, the CNN takes a fixed window of words around it and applies a filter on their representation to extract a feature vector for w_i. We concatenate the feature vectors corresponding to different filters for w_i to compute word representations.

Transformer: A Transformer encoder (Vaswani et al., 2017) that takes a linear transformation of the input words to learn contextualized representations.

BiLSTM: A bi-directional LSTM that takes the input representation and models the context in a forward and backward direction.

ON-LSTM: A variant of the LSTM neural network proposed by (Shen et al., 2018) which has an inductive bias toward learning latent tree structures.

2.3 GCN Encoder

First, we interpret the syntactic parse tree as an adjacency binary matrix $A^{n \times n}$ (n is the sentence length) with entries $A_{ij} = 1$ if there is a connection between nodes i and j, and $A_{ij} = 0$ otherwise. To apply a GCN on A, we consider the tree with self-loops at each node (i.e., $A_{ii} = 1$), ensuring

[1] An LSTM variant with an inductive bias toward learning latent tree structures in sequences.

[2] https://stanfordnlp.github.io/CoreNLP/

[3] Early experimentation with RoBERTa (Liu et al., 2019) yielded lower performance.

nodes are informed by their corresponding representations at previous layers. Formally, let $H^{(k)}$ be the output at the k-th GCN layer, $H^{(k)}$ is given by:

$$H^{(k)} = \text{ReLU}(AH^{(k-1)}W^{(k)}) + H^{(k-1)} \quad (1)$$

where $k = 1, \ldots, K$, $W^{(k)}$ is a parameter matrix at layer k. $RELU$ is used as the activation function. $H^{(0)}$ corresponds to the set of word representations extracted by the text encoder. The second term in (1) induces a residual connection that retains the contextual information of $H^{(0)}$ during the propagation process (Sun et al., 2020).

2.4 Classification and Optimization

Our model uses the representation $H^{(l)}$ (where $l \geq 0$), applies a linear layer and then normalize it with a softmax function to output a probability distribution over the set {B,I,O} for each word in the input. During training, we minimize the cross-entropy function for each word in text for the entire training set.

3 Experiments and Results

3.1 Baselines

We compare our methods with Distance-rule (Hu and Liu, 2004b); Dependency-rule (Zhuang et al., 2006b); $\text{LSTM}_{\text{word}}$ and $\text{BiLSTM}_{\text{word}}$ (Liu et al., 2015); Pipeline (Fan et al., 2019b); TC-BiLSTM (Fan et al., 2019b); IOG (Fan et al., 2019b); LOTN (Wu et al., 2020a); and ONG (Veyseh et al., 2020).[4]

3.2 Data

Following (Wu et al., 2020b), we use four benchmark datasets including restaurant (Res14, Res15, Res16) and laptop (Lap14) reviews from Semeval (Pontiki et al., 2014, 2015, 2016). We use the preprocessed data provided by Fan et al. (2019a). Table 2 shows the dataset statistics.

3.3 Implementation Details

Hyper-parameters are tuned on 20% of samples randomly selected from the train set since there is no development set.[5] We use the Adam optimizer to train all models. Models that use Glove word vectors are optimized with learning rate $1e^{-3}$ and trained for 100 epochs with batch size 16. Models that use BERT hidden vectors are optimized with learning rate $1e^{-5}$ and trained with batch size 6. Our source code is publicly available.[6]

Dataset	#Sent.	#ASL	#AT	#OT	#D.Dist.	#S.Dist.
Lap14 (Train)	1151	20.78	1632	1877	2.40	4.25
Lap14 (Test)	343	17.33	482	567	2.03	4.00
Res14 (Train)	1625	19.11	2636	3057	2.11	3.68
Res14 (Test)	500	19.22	862	1028	2.01	3.97
Rest15 (Train)	754	16.50	1076	1277	1.97	3.62
Rest15 (Test)	325	17.47	436	493	2.13	3.53
Rest16 (Train)	1079	16.78	1512	1770	2.01	3.59
Rest16 (Test)	328	16.54	456	524	1.93	3.43

Table 2: Dataset Statistics. No. of sentences (#Sent), Avg. sentence length (#ASL), No. of aspect terms (#AT), No. of opinion words (#OT), Avg. dependency distance (#D.Dist) and Avg. sequential distance (#S.Dist) between aspect and opinion.

3.4 Performance Comparison

Table 3 presents the results of all methods. Our models that use Glove Input (or BERT Input) are appended with "G"(or "B") to distinguish them. We report precision (Prec), recall (Rec), F1 score and average F1 score (Avg.F1) across all datasets.

Comparison of Text Encoders: We first observe that CNN(G) is adept at exploiting the information from simpler word representations (Glove), outperforming the Transformer(G) by +4.52 Avg.F1. We believe that this behavior is due to the fact that TOWE is a short-sequence task (see #ASL in Table 2). This assumption lies well with previous observations by (Yin et al., 2021), which found that CNNs often perform better than Transformers at short-sequence tasks. However, the Transformer(B) is able to improve performance and even outperform CNN(B) by +0.71 Avg.F1 by using BERT.

In addition, we find that ON-LSTM(G) and ON-LSTM(B) lag behind BiLSTM(G) and BiLSTM(B) by 4.33 and 0.54 Avg.F1 respectively. ON-LSTM performs worse than LSTMs on tasks that require tracking short-term dependencies (Shen et al., 2018). Since opinion words are usually close to the target in the sequence (see #ASL vrs. #S.Dist. in Table 2), tracking short-term dependency information is important in TOWE. This explains why

[4]Note that $\text{LSTM}_{\text{word}}$/$\text{BiLSTM}_{\text{word}}$ only use word embeddings as input.

[5]We use 300-dim Glove word vectors (Pennington et al., 2014) and apply a dropout of 0.8. Dimensions of part-of-speech and position embeddings are set to 30, but dimensions of position embeddings for pretrained models are set to 100. The CNN uses three filters with sizes 3, 4 and 5 and has a hidden dimension of 300. All other models have a hidden dimension of 200. The number of GCN layers is set over $K \in \{1, \ldots, 5\}$. Experiments are performed on NVIDIA Tesla V100.

[6]https://github.com/samensah/Encoders_TOWE_EMNLP2021

Model	Lap14			Res14			Res15			Res16			
	Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1	Avg.F1
Distance-rule	50.13	33.86	40.42	58.39	43.59	49.92	54.12	39.96	45.97	61.90	44.57	51.83	47.04
Dependency-rule	45.09	31.57	37.14	64.57	52.72	58.04	65.49	48.88	55.98	76.03	56.19	64.62	53.95
$\text{LSTM}_{\text{word}}$	55.71	57.53	56.52	52.64	65.47	58.34	57.27	60.69	58.93	62.46	68.72	65.33	59.78
$\text{BiLSTM}_{\text{word}}$	64.52	61.45	62.71	58.34	61.73	59.95	60.46	63.65	62.00	68.68	70.51	69.57	63.56
Pipeline	72.58	56.97	63.83	77.72	62.33	69.18	74.75	60.65	66.97	81.46	67.81	74.01	68.50
TC-BiLSTM	62.45	60.14	61.21	67.65	67.67	67.61	66.06	60.16	62.94	73.46	72.88	73.10	66.22
IOG	73.24	69.63	71.35	82.85	77.38	80.02	76.06	70.71	73.25	82.25	78.51	81.69	76.58
LOTN	77.08	67.62	72.02	84.00	80.52	82.21	76.61	70.29	73.29	86.57	80.89	83.62	77.79
ONG	73.87	77.78	75.77	83.23	81.46	82.33	76.63	81.14	78.81	87.72	84.38	86.01	80.73
Glove Input													
Transformer(G)	68.33	61.91	64.91	71.77	70.29	70.98	78.90	59.07	67.41	83.59	70.57	76.49	69.94
CNN(G)	64.81	73.83	69.00	75.86	78.83	77.29	68.21	73.87	70.91	76.93	84.77	80.64	74.46
ON-LSTM(G)	69.27	69.70	69.47	83.01	76.98	79.87	76.19	74.24	75.20	84.17	82.90	83.52	77.02
BiLSTM(G)	76.49	70.94	73.59	86.22	83.44	84.80	81.49	77.93	79.66	88.96	84.05	87.36	81.35
Transformer+GCN(G)	66.32	70.83	68.45	82.98	75.14	78.82	76.80	69.45	72.88	84.71	79.92	82.25	75.60
CNN+GCN(G)	66.88	74.88	70.65	82.45	80.12	81.24	75.32	73.75	74.51	82.17	84.89	83.48	77.47
ON-LSTM+GCN(G)	71.63	74.04	72.75	87.06	80.97	83.90	80.18	77.53	78.83	89.89	83.97	86.82	80.58
BiLSTM+GCN(G)	76.49	74.46	75.46	87.60	83.66	85.57	82.32	78.82	80.52	91.63	85.65	**88.52**	82.52
BERT Input													
Transformer(B)	78.88	78.03	78.13	83.97	84.40	84.18	82.37	78.21	80.22	88.22	84.05	86.06	82.14
CNN(B)	77.94	75.91	76.87	86.35	82.16	84.20	80.01	78.62	79.30	88.50	82.41	85.33	81.43
ON-LSTM(B)	77.96	77.53	77.71	85.58	83.25	84.39	82.57	78.34	80.38	87.76	83.55	86.54	82.26
BiLSTM(B)	78.38	78.27	78.25	86.38	84.82	85.60	82.17	78.78	80.41	89.94	84.16	86.94	82.80
Transformer+GCN(B)	79.38	77.04	78.19	85.43	84.18	84.79	82.21	79.55	**80.84**	89.34	84.16	86.66	82.62
CNN+GCN(B)	79.19	76.19	77.62	84.96	84.08	84.50	82.39	77.36	79.77	88.16	84.09	86.06	81.98
ON-LSTM+GCN(B)	80.33	76.01	77.96	85.68	84.03	84.83	82.14	78.18	80.07	89.35	83.93	86.54	82.35
BiLSTM+GCN(B)	79.72	78.06	**78.82**	86.45	85.06	**85.74**	83.37	77.93	80.54	88.98	85.80	87.35	**83.11**

Table 3: Results of experiments across baseline methods (across 5 runs). Results of compared models are retrieved from (Veyseh et al., 2020). The best F1 performance is bold-typed.

Model	Lap14			Res14			Res15			Res16		
	Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1
BiLSTM+GCN(G)	76.49	74.46	75.46	87.60	83.66	85.57	82.32	78.82	80.52	91.63	85.65	88.52
— GCN	76.49	70.94	73.59	86.22	83.44	84.80	81.49	77.93	79.66	88.96	84.05	87.36
— GCN, POST	75.38	70.12	72.63	86.83	82.94	84.83	82.45	75.58	78.85	88.71	84.01	86.29
— GCN, POST, POSN	61.65	62.08	61.80	63.17	56.63	59.66	62.16	61.54	61.76	70.11	70.23	70.08
BiLSTM+GCN(B)	79.72	78.06	78.82	86.45	85.06	85.74	83.37	77.93	80.54	88.98	85.80	87.35
— GCN	78.38	78.27	78.25	86.38	84.82	85.60	82.17	78.78	80.41	89.94	84.16	86.94
— GCN, POSN	62.92	72.17	67.21	60.84	64.42	62.54	63.88	64.42	63.97	69.59	71.45	70.39

Table 4: Precision, Recall and F1 scores of ablated models on the benchmark datasets (across 5 runs).

BiLSTM(G)(or BiLSTM(B)) achieves a better performance over ON-LSTM(G)(or ON-LSTM(B)).

The performance of BiLSTM(G) over $\text{BiLSTM}_{\text{word}}$ suggests that the substantial boost in performance comes from either part-of-speech or position embeddings. We later perform an ablation experiment to examine which information is more useful. Interestingly, BiLSTM(G) outperforms the current state-of-the-art ONG by +0.62 Avg.F1 despite its simple architecture, demonstrating the importance to first experiment with simpler methods before designing more complex structures.

Comparison of Text+GCN Encoders: Adding a GCN over any text encoder generally improves performance. This happens because the GCN provides additional syntactic information that is helpful for representation learning. We find that BiLSTM+GCN(G) achieves few gains over BiLSTM(G) while other text encoders including Transformer+GCN(G) and CNN+GCN(G) achieve relatively higher gains than their counterparts. This suggest that BiLSTM(G) has an inductive bias appropriate for the TOWE task and the performance mostly depends on the quality of the input representation. We observe that when using BERT embeddings, there is a minimal performance difference between using GCNs or not. We attribute this to the expressiveness of BERT embeddings and its ability to capture syntactic dependencies (Jawahar et al., 2019). Overall results suggest that our proposed method outperforms SOTA consistently across datasets.

3.5 Ablation Study

We perform ablation experiments on the two best performing models, BiLSTM+GCN(G) and BiLSTM+GCN(B), to study the contribution of their different components. The results are shown in Table 4. On BiLSTM+GCN(G), as we consecutively remove the GCN and POST embeddings from the input representation, we observe a slight drop in performance. The results indicate that POST embeddings as well as the GCN are not critical compo-

nents for BiLSTM+GCN(G). Therefore, they can be ignored to reduce model complexity. However, we observe a substantial drop in performance by removing the position embedding from the input representation, obtaining an F1 score equivalent to $\text{BiLSTM}_{\text{word}}$ across datasets. Similarly, removing the position embeddings in BiLSTM+GCN(B) causes a substantial drop in performance. The results suggest that leveraging position embeddings is crucial for TOWE performance.

4 Conclusion

We presented through extensive experiments that by employing a simple BiLSTM architecture that uses input representations from pre-trained word embeddings or language models, POST embeddings and position embeddings, we can obtain competitive, if not better results than the more complex current state-of-the-art methods Veyseh et al. (2020). The BiLSTM succeeds in exploiting position embeddings to improve performance. By adapting a GCN to incorporate syntactic information from the sentence we achieve further gains. In future work, we will explore how to improve existing TOWE models by effectively leveraging position embeddings.

Acknowledgements

Samuel Mensah and Nikolaos Aletras are supported by a Leverhulme Trust Research Project Grant.

References

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina N. Toutanova. 2018. 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.

Chris Dyer, Gábor Melis, and Phil Blunsom. 2019. A critical analysis of biased parsers in unsupervised parsing. *arXiv preprint arXiv:1909.09428*.

Zhifang Fan, Zhen Wu, Xin-Yu Dai, Shujian Huang, and Jiajun Chen. 2019a. 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.

Zhifang Fan, Zhen Wu, Xinyu Dai, Shujian Huang, and Jiajun Chen. 2019b. 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, Volume 1 (Long and Short Papers)*, pages 2509–2518.

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

Minqing Hu and Bing Liu. 2004a. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 168–177.

Minqing Hu and Bing Liu. 2004b. Mining and summarizing customer reviews. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Seattle, Washington, USA, August 22-25, 2004*, pages 168–177. ACM.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does bert learn about the structure of language? In *ACL 2019-57th Annual Meeting of the Association for Computational Linguistics*.

Thomas N. Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. In *ICLR (Poster)*.

Yann LeCun, Bernhard E Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne E Hubbard, and Lawrence D Jackel. 1990. Handwritten digit recognition with a back-propagation network. In *Advances in neural information processing systems*, pages 396–404.

Bing Liu. 2012. Sentiment analysis and opinion mining. *Synthesis lectures on human language technologies*, 5(1):1–167.

Pengfei Liu, Shafiq R. Joty, and Helen M. Meng. 2015. Fine-grained opinion mining with recurrent neural networks and word embeddings. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 1433–1443. The Association for Computational Linguistics.

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

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26*, volume 26, pages 3111–3119.

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.

Maria Pontiki, Dimitrios Galanis, Haris Papageorgiou, Ion Androutsopoulos, Suresh Manandhar, Mohammad Al-Smadi, Mahmoud Al-Ayyoub, Yanyan Zhao, Bing Qin, Orphée De Clercq, et al. 2016. Semeval-2016 task 5: Aspect based sentiment analysis. In *International workshop on semantic evaluation*, pages 19–30.

Maria Pontiki, Dimitrios Galanis, Harris Papageorgiou, Suresh Manandhar, and Ion Androutsopoulos. 2015. Semeval-2015 task 12: Aspect based sentiment analysis. In *Proceedings of the 9th international workshop on semantic evaluation (SemEval 2015)*, pages 486–495.

Maria Pontiki, Dimitrios Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014. Semeval-2014 task 4: Aspect based sentiment analysis. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, page 27–35.

Guang Qiu, Bing Liu, Jiajun Bu, and Chun Chen. 2011. Opinion word expansion and target extraction through double propagation. *Computational linguistics*, 37(1):9–27.

Lance A Ramshaw and Mitchell P Marcus. 1999. Text chunking using transformation-based learning. In *Natural language processing using very large corpora*, pages 157–176. Springer.

Yikang Shen, Shawn Tan, Alessandro Sordoni, and Aaron C. Courville. 2018. Ordered neurons: Integrating tree structures into recurrent neural networks. In *International Conference on Learning Representations*.

Kai Sun, Richong Zhang, Yongyi Mao, Samuel Mensah, and Xudong Liu. 2020. Relation extraction with convolutional network over learnable syntax-transport graph. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8928–8935.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R. Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R. Bowman, Dipanjan Das, and Ellie Pavlick. 2019. What do you learn from context? probing for sentence structure in contextualized word representations. In *7th International Conference on Learning Representations, ICLR 2019*.

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 *Proceedings of the 31st International Conference on Neural Information Processing Systems*, volume 30, pages 5998–6008.

Amir Pouran Ben Veyseh, Nasim Nouri, Franck Dernoncourt, Dejing Dou, and Thien Huu Nguyen. 2020. Introducing syntactic structures into target opinion word extraction with deep learning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 8947–8956. Association for Computational Linguistics.

Zhen Wu, Fei Zhao, Xin-Yu Dai, Shujian Huang, and Jiajun Chen. 2020a. Latent opinions transfer network for target-oriented opinion words extraction. 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 9298–9305. AAAI Press.

Zhen Wu, Fei Zhao, Xin-Yu Dai, Shujian Huang, and Jiajun Chen. 2020b. Latent opinions transfer network for target-oriented opinion words extraction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 9298–9305.

Xiaoyu Yin, Dagmar Gromann, and Sebastian Rudolph. 2021. Neural machine translating from natural language to sparql. *Future Generation Computer Systems*, 117:510–519.

Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, and Jun Zhao. 2014. Relation classification via convolutional deep neural network. In *Proceedings of COLING 2014, the 25th international conference on computational linguistics: technical papers*, pages 2335–2344.

Jie Zhou, Jimmy Xiangji Huang, Qinmin Vivian Hu, and Liang He. 2020. Is position important? deep multi-task learning for aspect-based sentiment analysis. *Applied Intelligence*, 50:3367–3378.

Li Zhuang, Feng Jing, and Xiao-Yan Zhu. 2006a. Movie review mining and summarization. In *Proceedings of the 15th ACM international conference on Information and knowledge management*, pages 43–50.

Li Zhuang, Feng Jing, and Xiaoyan Zhu. 2006b. Movie review mining and summarization. In *Proceedings of the 2006 ACM CIKM International Conference on Information and Knowledge Management, Arlington, Virginia, USA, November 6-11, 2006*, pages 43–50. ACM.