Syntax-Aware Opinion Role Labeling with Dependency Graph Convolutional Networks

Bo Zhang[1], **Yue Zhang**[2], **Rui Wang**[2], **Zhenghua Li**[1]*, **Min Zhang**[1]
1. Institute of Artificial Intelligence, School of Computer Science and Technology, Soochow University, Suzhou, China
2. Alibaba Inc., Hangzhou, China
{bzhang17}@stu.suda.edu.cn, {zhli13,minzhang}@suda.edu.cn
{shiyu.zy,masi.wr}@alibaba-inc.com

Abstract

Opinion role labeling (ORL) is a fine-grained opinion analysis task and aims to answer "*who expressed what kind of sentiment towards what?*". Due to the scarcity of labeled data, ORL remains challenging for data-driven methods. In this work, we try to enhance neural ORL models with syntactic knowledge by comparing and integrating different representations. We also propose dependency graph convolutional networks (DEPGCN) to encode parser information at different processing levels. In order to compensate for parser inaccuracy and reduce error propagation, we introduce multi-task learning (MTL) to train the parser and the ORL model simultaneously. We verify our methods on the benchmark MPQA corpus. The experimental results show that syntactic information is highly valuable for ORL, and our final MTL model effectively boosts the F1 score by 9.29 over the syntax-agnostic baseline. In addition, we find that the contributions from syntactic knowledge do not fully overlap with contextualized word representations (BERT). Our best model achieves 4.34 higher F1 score than the current state-of-the-art.

1 Introduction

Opinion and sentiment analysis has a wide range of real-world applications like social media monitoring (Bollen et al., 2011), stock market prediction (Nguyen et al., 2015), box office prediction (Yu et al., 2010), and general e-commerce applications (Kim et al., 2013; Hu et al., 2017; Cui et al., 2017). In particular, fine-grained opinion analysis aims to identify users' opinions in a text, including opinion expressions, holders of the opinions, targets of the opinions, target-dependent attitude, and intensity of opinions (Marasović and Frank, 2018), which is very important for understanding political stance, customers' reviews, marketing trends, and other subjective information (Ravi and Ravi, 2015). As a typical fine-grained opinion mining task, opinion role labeling (ORL) aims to identify different roles relevant to each opinion, i.e., *who expressed what kind of sentiment towards what* (Liu, 2012).

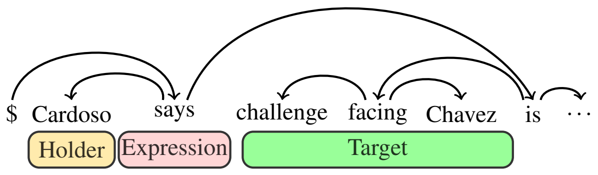


Figure 1: An Example of ORL (bottom) and syntactic dependency tree (top) for "Cardoso says challenge facing Chavezis is reestablishing normalcy."

Due to the lack of large-scale labeled data, ORL remains a challenging task to tackle. As a reference point, semantic role labeling (SRL) is very similar to ORL in the problem definition, but has 10 times more labeled data and thus achieves much higher performance than ORL (80∼90 vs. 60∼70 in F1 score). Motivated by the correlations between the two tasks, SRL has been utilized to help the ORL task by many previous studies (Ruppenhofer et al., 2008; Marasović and Frank, 2018; Zhang et al., 2019b). However, when opinion expressions and arguments compose complicated syntactic structures, it is difficult to correctly recognize the opinion arguments even with shallow semantic representation like SRL (Marasović and Frank, 2018).

To compensate for the limited scale of labeled data for data-driven approaches, linguistic knowledge like syntax provides structural information representing human understanding of the text. Naturally, dependency relations between words ease the discovering of opinion roles. Taking the example in Figure 1, the *Target* span is often incompletely recognized without syntactic dependency

*Corresponding author

relations, missing either "facing Chavez" or "challenge". For the similar SRL task, many previous works have proposed to incorporate syntax into the neural models (Marcheggiani and Titov, 2017; He et al., 2018; Xia et al., 2019a). In contrast, few studies in the recent years explore this line of research for ORL.

There are two barriers to apply syntactic dependency parsing to NLP tasks, i.e., 1) inaccuracy of the parsing results, and 2) error propagation of the processing pipeline. To overcome the first barrier, instead of employing the final discrete outputs (i.e., single 1-best dependency trees), we make use of the probability matrix of all dependency arcs (also can be viewed as an edge-weighted directed graph) before searching for the 1-best tree. Such probabilistic representation of syntax provides more information while alleviating parsing errors. For the second barrier, considering that the pipeline methods are notorious for the error propagation problem, we introduce multi-task learning (MTL) frameworks, which have been widely used in many NLP models when predictions at various processing levels are needed (Collobert and Weston, 2008; Ruder, 2017).

Apart from the syntactic information, contextualized word representations like BERT (Devlin et al., 2019) are widely used to compensate for the sparsity of task-specific training data. They compress distributional semantics of words from large corpora, making the local context fluent and natural. However, the long-distance dependencies between words are often ignored, which is ideally able to be captured by syntactic analysis.

In summary, based on previous studies in using syntax to improve various tasks, this work investigates whether syntax can enhance the neural ORL model. Particularly, we try to answer the following three questions.

- How to effectively integrate various syntactic information into the neural ORL model?
- How to alleviate the propagation of errors brought by syntactic parsing?
- Is syntactic knowledge already covered by the contextualized word representations like BERT?

Based on our experiments, we observe that 1) compared with single 1-best parse trees, encoding the edge-weighted graphs achieves better results, as the model is less sensitive to parsing errors while keeping richer structural information; 2) integrating various syntactic information, both explicit and implicit, boosts performance, and MTL framework can effectively alleviate the error propagation problem; and 3) contributions from syntactic information, especially from long-distance dependency relations, do not fully overlap with those from the contextualized word representations like BERT. Our overall model delivers a new state-of-the-art result on the benchmark MPQA corpus, with 4.34 absolute improvement over the previous best result.

2 Related work

An opinion consists of several components, e.g., expressions, holders, and targets. Some previous works focus on recognizing some components, whereas others try to recognize all components at the same time. Yang and Cardie (2014) and Breck et al. (2007) work entirely on labeling of the opinion expressions. Kim and Hovy (2006) and Johansson and Moschitti (2013) apply pipeline models to firstly predicting opinion expressions and then labeling holders and targets for each expression. Joint models simultaneously identify all opinion components, predicting which role is related to which opinion (Choi et al., 2006; Yang and Cardie, 2013; Katiyar and Cardie, 2016). In this work, we follow the opinion role labeling (ORL) task setting of Marasović and Frank (2018) and Zhang et al. (2019b), and try to predict holders and targets for the given opinion expressions.

Previous works make use of SRL resources to address the issue of data scarcity for ORL, considering SRL is highly related to ORL and has a considerable amount of training data. Inspired by the similarity between ORL and SRL in task definition, Kim and Hovy (2006) and Ruppenhofer et al. (2008) address ORL with a well-trained SRL model by treating opinion expressions as semantic predicates, and opinion roles as semantic roles. Marasović and Frank (2018) take SRL as an auxiliary task, and employ different MTL frameworks to learn the common grounds between ORL and SRL and distinguish task-specific knowledge. Zhang et al. (2019b) extract neural features from a well-trained SRL model as SRL-aware word representations, and then feed them into the input layer of ORL, aiming to alleviate the error propagation problem.

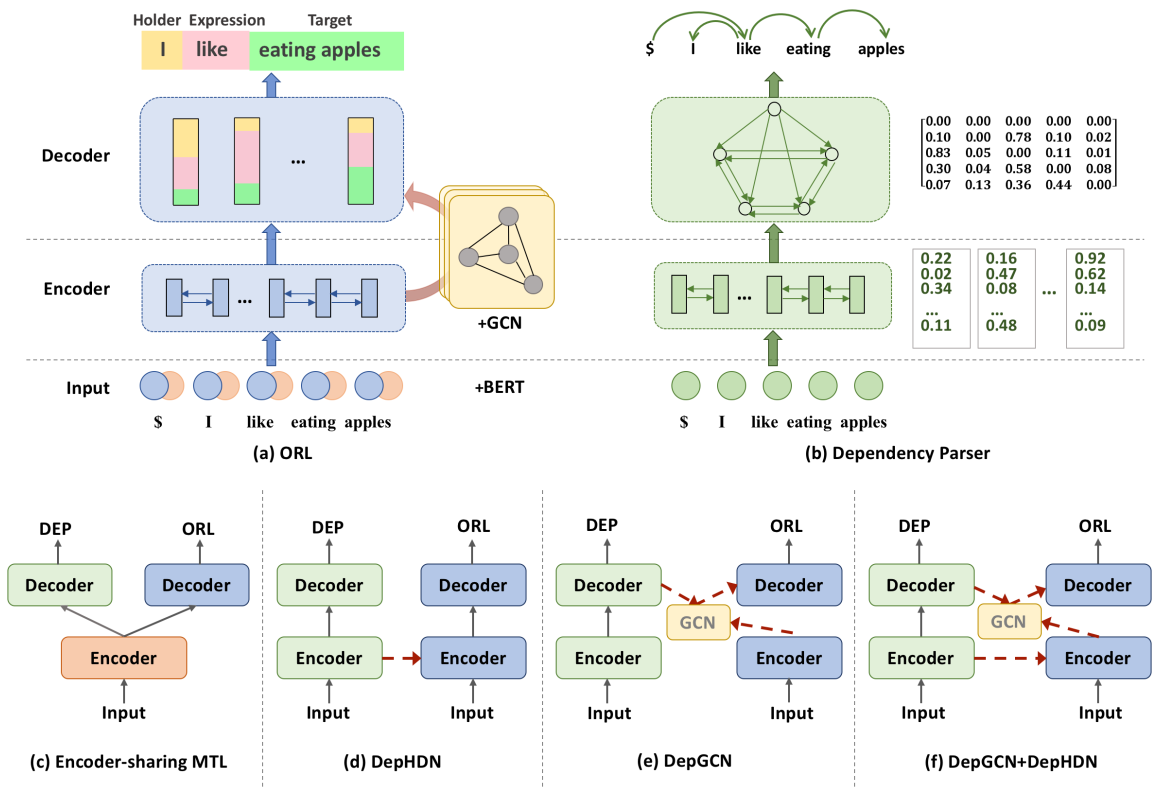


Figure 2: The overall architecture of our models.

Many previous works have shown that syntactic information is of great value for SRL and other NLP tasks (He et al., 2018; Zhang et al., 2019c; Strubell et al., 2018; Xia et al., 2019a; Miwa and Bansal, 2016; Zhang et al., 2019a). Xia et al. (2019b) use the relative position between predicate words and other words in a dependency tree to represent syntactic information, while Roth and Lapata (2016) employ LSTM to obtain the embedding of a dependency path. Tai et al. (2015) and Kipf and Welling (2016) propose TreeLSTM and graph convolution network (GCN) to encode the tree/graph-structural data respectively. Both TreeLSTM and GCN are commonly used techniques to encode parse trees (Miwa and Bansal, 2016; Marcheggiani and Titov, 2017; Bastings et al., 2017). Zhang et al. (2019a) and Xia et al. (2019a) extract the hidden states from the LSTM encoder of the parser model as syntax-aware word representations, and feed them to downstream tasks as extra inputs.

In contrast, few works have proved that syntactic knowledge is useful in the neural ORL models. Yang and Cardie (2013) integrate the shortest path features from dependency trees into a traditional CRF-based ORL model. To our best knowledge, this work is the first to investigate how to incorporate syntax into neural ORL models.

3 Basic Models

The ORL model aims to extract *opinion-holder-target* structures from text by identifying the segments of these opinion arguments. The task can be modeled as a sequence labeling problem. We adopt the $\{BMESO\}$ encoding schema to assign a tag for each word (Zhang et al., 2019b). Following Marasović and Frank (2018) and Zhang et al. (2019b), we focus on recognizing the holders and the targets for the given opinion expression and exploit a deep BiLSTM-CRF-based model as our baseline.

The Figure 2-(a) shows the architecture of our ORL baseline model, which is composed of three key components, i.e., the input layer, the BiLSTM-based encoder, and the CRF-based decoder. Given the input sentence $S = w_1, w_2, ..., w_n$ and the opinion expression segment $E = w_s, w_{s+1}, ..., w_e (1 \leq s \leq e \leq n)$, the input vector consists of the word embeddings and the expression-indicator embeddings as the following equation shows:

$$\mathbf{x}_i = \mathbf{e}_{w_i}^{word} \oplus \mathbf{e}_{0/1}^{exp} \quad (1)$$

where $\mathbf{e}_{w_i}^{word}$ is the embedding of word w_i, and the expression-indicator embedding is $\mathbf{e}_0^{exp}$ for non-expression words and $\mathbf{e}_1^{exp}$ for words inside the opinion expression (i.e., $s \leq i \leq e$). At the encoder layer, we apply three stacking layers of BiLSTM to fully encode the sentence and obtain the expression-specific representations at word level. The CRF-based decoder at the output layer delivers the globally optimal sequence tags.

The Biaffine parser is the state-of-the-art dependency parser proposed by Dozat and Manning (2017), as shown in Figure 2-(b). The parser contains a multi-layer BiLSTM layer for encoding the input sentence, followed by a biaffine transformation layer for computing the probabilities of all word pairs. Then it searches for the highest-scoring and well-formed tree via the maximum spanning tree (MST) algorithm.

The three cascaded layers, i.e., the BiLSTM-based encoder, the biaffine scorer, and the MST decoder, represent syntactic information at different levels. The encoder extracts the neural features from the input sentence and outputs hidden states (HDN), which can be regarded as *implicit* information. The 1-best output parse tree, on the other hand, conveys *explicit* syntactic structures. The biaffine scorer gives a probability matrix for all possible dependency arcs (also can be viewed as an edge-weighted directed graph), which represents richer *explicit* syntactic information than the 1-best parse tree.

4 The Syntax-Aware Approach

Despite of recent advances in dependency parsing (Dozat and Manning, 2017), parsers still cannot output parse trees with high accuracy on out-of-domain or irregular data. In this work, we exploit rich syntactic information contained in the edge-weighted graphs to mitigate the effects of parsing errors. Specifically, we firstly employ graph convolutional networks (GCN) to encode the edge-weighted graphs, and then integrate them into different processing levels of ORL with implicit parser hidden states. Finally, we employ novel MTL frameworks to alleviate the error propagation problem further.

4.1 Dependency Graph Convolutional Networks (DEPGCN)

In this subsection, we propose dependency graph convolutional networks (DEPGCN) to better encode the syntactic information from the edge-weighted graphs. On the one hand, compared with explicit 1-best parse trees, edge-weighted graphs convey richer structural information by providing all latent syntactic structures, and avoid error propagation as well. On the other hand, compared with the implicit hidden states of the parser encoder (Zhang et al., 2019a; Xia et al., 2019a), an edge-weighted graph, denoted as an attention matrix, explicitly captures the modification strength of word pairs.

The original GCN is designed for directly modeling graph-structured data (Kipf and Welling, 2016). Although each node only receives information from its immediate neighbors through edges in one GCN layer, multi-layer GCN can propagate information more globally if there exist connected paths. Formally, the output of node i at the l-th layer of GCN is computed by the following equation:

$$\mathbf{h}_i^{(l)} = F\left(\sum_{j=1}^{n} \mathbf{A}_{ij}\mathbf{W}^{(l)}\mathbf{h}_j^{(l-1)} + \mathbf{b}^{(l)}\right) \tag{2}$$

where $\mathbf{A}$ is the adjacency matrix of a graph with n nodes, $\mathbf{W}^{(l)}$ and $\mathbf{b}^{(l)}$ are the model parameters, F is an activation function. $\mathbf{h}_i^0$ is the initial input vector.

As shown by Figure 2-(e), we apply DEPGCN to connect the parser model and the ORL model. We first obtain the edge-weighted graph from the decoder of a well-trained biaffine parser as a data preprocessing step, and then feed the graph into our DEPGCN in the form of an adjacency matrix A [1]. Then we feed the outputs of the ORL BiLSTM-based encoder as the initial inputs $\mathbf{h}^0$ to the DEPGCN. Finally, we feed the output of the DEPGCN to the CRF-based decoder, and update the ORL results under the guidance of the syntactic information.

Moreover, we introduce dense connections to the multi-layer DEPGCN for extracting more structural information (Huang et al., 2017; Guo et al., 2019). Instead of only adding connections between adjacent layers, we use dense connections from each layer to all the subsequent layers. Formally, the input of node i at the l-th layer is:

$$\mathbf{x}_i^{(l)} = \mathbf{h}_i^{(0)} \oplus \mathbf{h}_i^{(1)} \oplus \cdots \oplus \mathbf{h}_i^{(l-1)} \tag{3}$$

where $\mathbf{h}_i^{(l)}$ is the output of node i at the l-th layer. We also make residual connections over DEPGCN to mitigate the vanishing gradient problem, which

[1]Moreover, following Marcheggiani and Titov (2017), we also add a self-loop for each node in the graph, which means all diagonal elements of A are set to 1.

means that the output dimension of each DepGCN layer is decided by the layer number and the input dimension of the bottom DepGCN.

4.2 Combining Explicit and Implicit Syntax (DepGCN+DepHDN)

Different from explicit 1-best parse trees or edge-weighted graphs, hidden states of the BiLSTM encoder of a dependency parser provide useful syntactic knowledge and are less sensitive to parsing errors. Using such implicit syntactic representations has been demonstrated to be highly effective for downstream tasks (Zhang et al., 2019a; Xia et al., 2019a). In this section, we describe how to integrate implicit syntactic information from parser hidden states and explicit syntactic information from the edge-weighted graph into the ORL model for better performance.

We first briefly describe the use of the dependency parser's hidden states, named as DepHDN. As shown by Figure 2-(d), we extract the outputs from the parser encoder and feed them into the BiLSTM-based encoder of ORL as extra inputs. The hidden states of each parser BiLSTM layer are obtained as the syntactic representations, i.e., $\mathbf{h}_1^{(l)}, \cdots, \mathbf{h}_n^{(l)}$, where $\mathbf{h}_n^{(l)}$ is output of the l-th layer of the parser BiLSTM encoder at w_n. Then, we use the weighted-sum operation to get a single vector $\mathbf{h}_i^{syn}$ as the final syntactic representation of word w_i.

$$\mathbf{h}_i^{syn} = \mathbf{W}\lambda \sum_{j=1}^{L} \alpha_j \mathbf{h}_i^j \tag{4}$$

where L is the layer number of parser BiLSTM-based encoder; $\mathbf{W}$, α_j and λ are model parameters; α_j is softmax-normalized weights for h^j; λ is used to scale the syntactic representations. The syntactic representations $\mathbf{h}_i^{syn}$ are concatenated with the original ORL input vectors, so that richer word representations are obtained.

Furthermore, in order to simultaneously benefit from the implicit and explicit syntactic information, as shown in Figure 2-(f), we simply extract the edge-weighted graph from the parser decoder and apply the DepGCN approach over the ORL encoder to obtain syntax-enhanced representations.

4.3 Pipeline vs. Multi-Task Learning

The three approaches, depicted in Figure 2-(d-f) respectively, can work either in the pipeline way or in the MTL way. Specifically, the pipeline way first trains the dependency parser and then fixes the parser components during training the ORL model. In contrast, the MTL way trains both the parser and the ORL model at the same time. In this subsection, we explore the MTL way to alleviate the error propagation problem further besides the DepGCN approach.

As a baseline, Figure 2-(c) shows the most common MTL method, which shares a common encoder and uses multiple task-specific output layers, known as the hard-parameter-sharing MTL (Ruder, 2017; Marasović and Frank, 2018). However, this approach is not suitable for our scenario where the auxiliary parsing task has much more labeled data than the main ORL task, since the shared encoder is very likely to bias toward to parsing performance (Xia et al., 2019a).

Inspired by Xia et al. (2019a), we adopt the architectures of Figure 2-(d-f) to keep task model parameters separately, and train ORL and the parser simultaneously. We update model parameters according to the combined loss of the ORL and the dependency parser during training:

$$\zeta = \zeta_{ORL} + \alpha \zeta_{Dep} \tag{5}$$

where ζ_{ORL} and ζ_{Dep} is the loss of the ORL model and the parser respectively, and α is a corpus weighting factor to control the loss contribution of the dependency data in each batch as discussed in Section 5.

Compared with the previous pipeline training process, the parameters of the parser are not pretrained and fixed, but updated by training objectives of both ORL and the parser. This results in a ORL-preferred dependency parsing model.

5 Experiment Setup

Dataset. We conduct experiments on MPQA version 2.0 corpus (Wiebe et al., 2005), which has been widely adopted as a benchmark dataset for opinion mining (Katiyar and Cardie, 2016; Marasović and Frank, 2018; Zhang et al., 2019b). In this work, we adopt the same data split (132/350 documents as dev/test data) and the same five-fold cross-validation (CV) data split on the test data as Zhang et al. (2019b) for a fair comparison.

Evaluation Metrics. Unless specified, we use recall (R), precision (P) and their F1 measure value of exact match to evaluate the ORL performance, and the results are the average of the five-fold CV experiments. Following Marasović and Frank

(2018) and Zhang et al. (2019b), we also include the *binary* and *proportional* overlap as additional evaluation metrics.

Dependency Parser. Following the standard practice in the dependency parsing community, the original phrase-structure Penn Treebank data are converted into the Stanford dependencies using the Stanford Parser v3.3.0. We use the converted dependency data to train our biaffine parser for obtaining the 1-best trees, the edge-weighted graphs, and the parser hidden states. In addition, we use the Stanford POS tagger to obtain POS tags for the biaffine parser. For other settings, we follow the work of Dozat and Manning (2017).

BERT. We use BERT (Bidirectional Encoder Representations from Transformers) (Devlin et al., 2019) to obtain deep contextualized word representations as our extra inputs. In particular, we use BERT-base (uncased) model and extract representations from the top-1 hidden layer. Our experiments show that using the top-1 layer representations performs better than the more common use of aggregating top-4 hidden layers.[2]

Parameters. We follow the previous works of Zhang et al. (2019b) and Marasović and Frank (2018) without much parameter tuning. Specifically, we use the pretrained 100-dimensional glove embeddings (Pennington et al., 2014). The BiLSTM layer number is set to 3, and the hidden output size is 200. We apply 0.33 dropout to word representation and the hidden states of the BiLSTM. We choose Adam (Kingma and Ba, 2014) to optimize model parameters with a learning rate 10^{-3}. The entire training instances are trained for 30 epochs with the batch size of 50, and the best-epoch model at the peak performance on the dev corpus is chosen. For the MTL, we train the batches of ORL and parsing in turn since this interleaving training can obtain better performance in our experiments. Besides, we use the corpus weighting trick to balance the gap in data sizes between the two tasks.

6 Results and Analysis

In this section, we first conduct experiments on the dev data to verify the effectiveness of our proposed approaches from several aspects: 1) how to effectively use explicit syntactic information; 2) usefulness of explicit vs. implicit syntax and their combination; 3) which MTL framework is most effective. Then we present overall results on the test dataset, with and without BERT. Finally, we conduct detailed analysis to gain more insights.

[2]In fact, we also investigate another typical use of BERT, i.e., the fine-tuning method. However, the ORL performance is much lower than the feature extraction method described above. Besides, considering the training speed and flexibility in our proposed syntax-aware model, it is more flexible to adopt the feature extraction method, i.e., extracting BERT outputs as extra word representations (frozen during training).

	P	R	F1
w/o Syntax			
BASELINE	59.08	55.15	57.02
w/ Explicit Info.			
DEPHEAD	60.82	55.30	57.91
TREELSTM	60.85	55.25	57.90
DEPGCN-HARD	61.10	56.16	58.50
DEPGCN	61.53	57.26	59.28
w/ Implicit Info.			
DEPHDN	63.42	59.61	61.45
Explicit & Implicit			
DEPGCN+DEPHDN	**63.80**	**61.43**	**62.58**

Table 1: Experiments with explicit and implicit syntactic information on the dev dataset.

6.1 Using Explicit Syntax

In order to know the best way to use explicit information from the dependency parser, we conduct comparative experiments by integrating the information of the explicit 1-best trees or the explicit edge-weighted graphs. The second major row of Table 1 shows the results of integrating such explicit syntactic information on the dev data.

In particular, BASELINE uses no syntactic information, known as the syntax-agnostic method; DEPHEAD concatenates an extra embedding of the head word in the 1-best parse tree with the original input; TREELSTM applies the TreeLSTM to encode the 1-best tree structures; DEPGCN applies GCN to encode the edge-weighted graphs. For DEPGCN-HARD, the 1-best tree is converted to a binary adjacency and is encoded by DEPGCN.

It is obvious that using explicit syntactic information is helpful for ORL. All the syntax-aware models improve the performance by 0.88$\sim$ 2.26 F1 score. The DEPHEAD approach is the most intuitive way to represent syntactic information by using head word embeddings, which serves as a simple syntax-aware baseline method. The TREELSTM approach encodes 1-best tree recursively in a much more complex way, but achieves nearly the same performance with the DEPHEAD method. We suspect the reason may be that the TREELSTM method is prone to parsing errors.

The DEPGCN-HARD approach also encodes

Multi-task learning	P	R	F1
M-BASELINE	62.23	56.84	59.39
M-DEPGCN	65.59	61.61	63.52
M-DEPHDN	65.74	63.67	64.68
M-DEPGCN+DEPHDN	**65.94**	**64.15**	**65.03**

Table 2: Experimental results under the MTL framework on the dev dataset.

the 1-best tree, and achieves higher performance. Compared with the TREELSTM approach, the DEPGCN-HARD approach is less sensitive to parsing errors, since a GCN layer only considers local adjacent structures and performs one-hop information propagation, whereas a TreeLSTM propagates information in either bottom-up or top-down order where earlier errors affect later computations a lot.

The best result of exploiting explicit information is obtained by the DEPGCN method, which is able to integrate richer structural information from edge-weighted graphs.

6.2 Explicit vs. Implicit Syntax, and Combination

The bottom two major rows of Table 1 show the results on the dev data. DEPHDN exploits implicit information of parser hidden states.

We can see that the implicit DEPHDN method outperforms the best explicit DEPGCN method by 2.17 F1 score, indicating the effectiveness of the integration of parser hidden states, which is consistent with previous studies on the SRL task (Xia et al., 2019a). The advantage of using implicit hidden states is being able to greatly alleviate the error propagation from explicit parsing results.

We further simultaneously integrate explicit and implicit syntactic information into one model, which achieves the best performance of 62.58 F1 score, and outperforms the syntax-agnostic baseline and the DEPHDN method by 5.56 and 1.13 F1 scores, respectively. This demonstrates that ORL can benefit from both explicit and implicit syntactic information.

In summary, we can conclude that encoding the edge-weighted graphs is more effective than the 1-best trees, and combining both explicit and implicit syntactic information brings higher performance than either.

6.3 Effects of Multi-Task Learning

In order to alleviate the error propagation problem and explore better integration of different approaches, we apply MTL frameworks to the above-mentioned pipeline architectures.

Table 2 shows the results of the MTL settings with previously better-performing configurations on the dev dataset, together with a commonly used hard-parameter-sharing MTL for parsing and ORL. M-BASELINE serves as an MTL baseline, which shares the encoder for the two tasks (Figure 2-c). M-DEPGCN and M-DEPHDN respectively apply the DEPGCN and DEPHDN approaches under our MTL framework, and M-DEPGCN+DEPHDN combines them.

Firstly, although sharing the encoder of the parser and ORL already brings in more than 2 F1 score improvement compared with the syntax-agnostic baseline (BASELINE), it is much inferior to other MTL approaches and the pipeline DEPHDN method (comparing Table 1). This may be caused by the weakness of the encoder parameters for ORL, as discussed in Section 4 and Xia et al. (2019a).

Secondly, compared with the corresponding approaches under the pipeline architecture, all approaches under our MTL framework improve the performance by 2.45∼4.24 F1 scores, which indicates that MTL is highly effective in alleviating the error propagation problem.

Finally, the combination of the explicit edge-weighted graphs and the implicit parser hidden states is still the most effective model under the MTL framework, outperforming the BASELINE in Table 1 by 8.01 F1 score.

6.4 Final Results

In this section, we report the overall performance of our approaches compared with previous methods on the test data, as shown in Table 3.

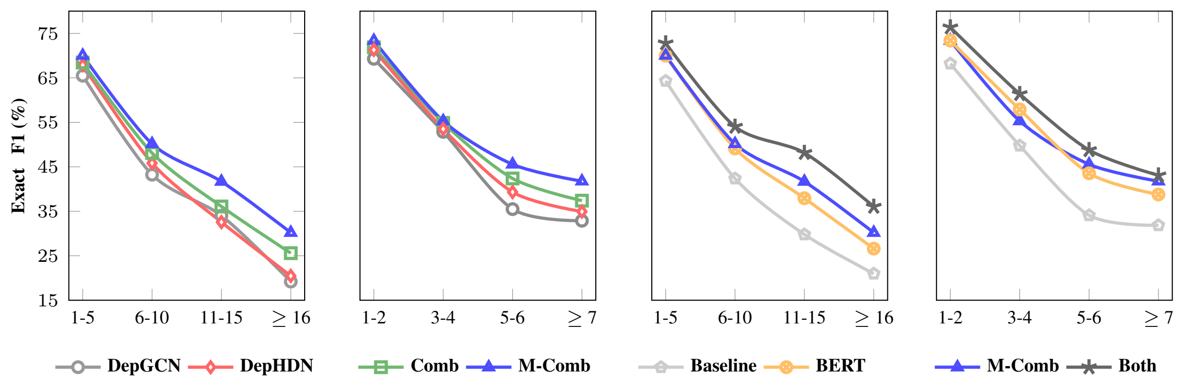
In particular, we list our syntax-agnostic baseline (BASELINE in Table 1), others' works (Zhang et al. (2019b) and Marasović and Frank (2018), using SRL for ORL), best non-MTL approaches based on our results on the dev data (DEPGCN for explicit syntactic information and DEPHDN for implicit syntactic information), and finally the MTL-based models. The results of BASELINE with BERT and our best model with BERT are also listed to demonstrate the contributions from the contextualized word representations.

We can draw the following findings.

- Combining explicit and implicit syntactic information improves the performance, indicat-

	Exact F1			Binary F1			Proportional F1		
	Holder	Target	Overall	Holder	Target	Overall	Holder	Target	Overall
Basic Model									
Baseline	73.05	44.21	58.79	81.21	69.50	75.43	79.33	62.53	71.03
Baseline+BERT	76.74	52.61	64.73	85.45	75.74	80.62	83.58	69.31	76.48
Zhang et al. (2019b)	73.07	42.70	58.30	81.57	68.34	75.15	79.35	61.22	70.55
w/ SRL									
Marasović and Frank (2018)	75.58	46.40	61.51	83.80	72.06	77.87	81.67	65.18	73.61
Zhang et al. (2019b)	76.95	50.50	63.74	84.91	73.29	79.10	82.82	67.31	75.08
w/ Syntax									
DepGCN	73.82	45.97	60.12	81.11	68.54	74.93	79.15	61.96	70.70
DepHDN	76.96	46.95	62.29	83.79	70.20	77.15	82.44	63.56	73.21
DepGCN + DepHDN	76.21	49.38	63.12	83.0	72.25	77.81	81.58	66.59	74.28
w/ Syntax + MTL									
M-DepGCN	77.50	50.78	64.28	84.17	72.91	78.60	82.77	66.77	74.85
M-DepHDN	77.36	50.81	64.31	84.35	72.45	78.50	82.95	66.51	74.87
M-DepGCN+DepHDN	78.01	51.92	65.13	84.97	73.36	79.24	83.67	67.77	75.82
M-DepGCN+DepHDN+BERT	**79.51**	**56.61**	**68.08**	**87.09**	**76.99**	**82.04**	**85.70**	**72.32**	**79.01**

Table 3: Overall experimental results on the test dataset.



(a) Span Length (b) Distance to Expression (c) Span Length (d) Distance to Expression

Figure 3: Performance on predicting arguments with different span lengths and distances to the expressions.

ing they are complementary to each other.

- Compared with the DepGCN and DepHDN approaches (i.e., explicit or implicit only), the DepGCN+DepHDN approach achieves better performance on both *Holder* and *Target* recognition.
- All of the MTL configurations boost the performance compared with their pipeline counterparts, as ORL-oriented parsing models are learned, and the error propagation problem is less severe.
- Our best syntax-aware MTL model combined with BERT achieves the best performance, outperforming the baseline with BERT by more than 3 F1 score.
- Compared with the previous state-of-the-art methods, we obtain 4.34 and 1.39 improvement of F1 scores with and without BERT, respectively. Overall, our best model achieves 9.29 higher F1 score over the syntax-agnostic baseline.

6.5 Further Analysis

In this section, we conduct analysis to better understand the contributions from the syntactic information and BERT. In particular, we compute the exact F1 score according to different lengths of opinion arguments, as well as different distances between the arguments and their corresponding expressions.

Influence of Syntax. Figure 3-(a-b) show the effects of syntax on predicting arguments of different span lengths and distances to their expressions, respectively. We observe that 1) the performance of combining explicit and implicit syntactic information is always higher than either of them, while the DepGCN and DepHDN approaches compensate each other at different argument span lengths; and 2) MTL performs better than the best pipeline

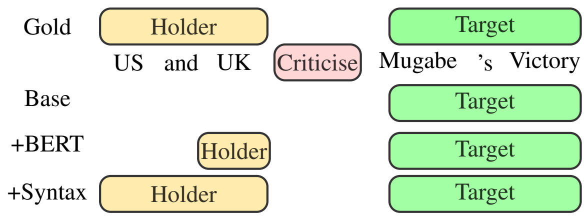


Figure 4: An example of different ORL outputs for "US and UK Criticise Mugabe 's Victory".

model consistently, which indicates that the usage of syntax is further enhanced as the error propagation is less severe.

Influence of BERT. Figure 3-(c-d) show the similar graphs of the best syntax-aware model and BERT. Firstly, both M-Comb and BERT bring substantial improvements over the syntax-agnostic baseline. Secondly, despite that the syntactic information and BERT are similar in the overall performance, the syntactic information is more effective for arguments with longer spans and farther distances to the expressions, as the syntax helps to capture long-distance dependencies between words. And lastly, the integration of syntax and BERT can further improve the performance, demonstrating that contributions from the two are complementary.

Case Study. One case study is given in Figure 4. In this example, the gold holder "US and UK" is difficult to be identified by the baseline model. Even with the help of BERT, which brings more contextual information, the model still only captures one of them, the closest holder "UK". Our syntax-aware model accurately predicts the holder due to the coordination structure being captured by the syntactic dependency information.

7 Conclusions

In this paper, we present a syntax-aware opinion role labeling approach based on dependency GCN and MTL. We compare different representations of syntactic dependency information and propose dependency GCN to encode richer structural information from different processing levels of the parser. The MTL framework further boosts the performance, and together with BERT, our best model achieves a new state-of-the-art result on the widely-used ORL benchmark MPQA corpus. Overall, our syntax-aware model brings in about 9.29 improvement of exact F1 score compared with the syntax-agnostic baseline.

Acknowledgments

The authors would like to thank the anonymous reviewers for the helpful comments. This work was supported by National Natural Science Foundation of China (Grant No. 61525205, 61876116) and a Project Funded by the Priority Academic Program Development (PAPD) of Jiangsu Higher Education Institutions, and was also partially supported by Alibaba Group through Alibaba Innovative Research Program.

References

Joost Bastings, Ivan Titov, Wilker Aziz, Diego Marcheggiani, and Khalil Sima'an. 2017. Graph convolutional encoders for syntax-aware neural machine translation. In *Proceedings of EMNLP*, pages 1957–1967.

Johan Bollen, Huina Mao, and Xiaojun Zeng. 2011. Twitter mood predicts the stock market. *Journal of computational science*, 2(1):1–8.

Eric Breck, Yejin Choi, and Claire Cardie. 2007. Identifying expressions of opinion in context. In *Proceedings of IJCAI*, volume 7, pages 2683–2688.

Yejin Choi, Eric Breck, and Claire Cardie. 2006. Joint extraction of entities and relations for opinion recognition. In *Proceedings of EMNLP*, pages 431–439.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of ICML*, pages 160–167.

Lei Cui, Shaohan Huang, Furu Wei, Chuanqi Tan, Chaoqun Duan, and Ming Zhou. 2017. SuperAgent: A customer service chatbot for e-commerce websites. In *Proceedings of ACL*, pages 97–102.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186.

Timothy Dozat and Christopher D. Manning. 2017. Deep biaffine attention for neural dependecy parsing. In *Proceedings of ICLR*.

Zhijiang Guo, Yan Zhang, and Wei Lu. 2019. Attention guided graph convolutional networks for relation extraction. In *Proceedings of ACL*, pages 241–251.

Shexia He, Zuchao Li, Hai Zhao, and Hongxiao Bai. 2018. Syntax for semantic role labeling, to be, or not to be. In *Proceedings of ACL*, pages 2061–2071.

Ya-Han Hu, Yen-Liang Chen, and Hui-Ling Chou. 2017. Opinion mining from online hotel reviews–a text summarization approach. *Information Processing & Management*, 53(2):436–449.

Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. 2017. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708.

Richard Johansson and Alessandro Moschitti. 2013. Relational features in fine-grained opinion analysis. *Computational Linguistics*, 39(3):473–509.

Arzoo Katiyar and Claire Cardie. 2016. Investigating lstms for joint extraction of opinion entities and relations. In *Proceedings of ACL*, pages 919–929.

Soo-Min Kim and Eduard Hovy. 2006. Extracting opinions, opinion holders, and topics expressed in online news media text. In *Proceedings of the Workshop on Sentiment and Subjectivity in Text*, pages 1–8.

Suin Kim, Jianwen Zhang, Zheng Chen, Alice Oh, and Shixia Liu. 2013. A hierarchical aspect-sentiment model for online reviews. In *Proceedings of AAAI*, pages 526–533.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Bing Liu. 2012. *Sentiment Analysis and Opinion Mining*. Morgan & Claypool.

Ana Marasović and Anette Frank. 2018. Srl4orl: Improving opinion role labeling using multi-task learning with semantic role labeling. In *Proceedings of NAACL-HLT*, pages 583–594.

Diego Marcheggiani and Ivan Titov. 2017. Encoding sentences with graph convolutional networks for semantic role labeling. In *Proceedings of EMNLP*, pages 1506–1515.

Makoto Miwa and Mohit Bansal. 2016. End-to-end relation extraction using lstms on sequences and tree structures. In *Proceedings of ACL*, pages 1105–1116.

Thien Hai Nguyen, Kiyoaki Shirai, and Julien Velcin. 2015. Sentiment analysis on social media for stock movement prediction. *Expert Systems with Applications*, 42(24):9603–9611.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of EMNLP*, pages 1532–1543.

Kumar Ravi and Vadlamani Ravi. 2015. A survey on opinion mining and sentiment analysis: tasks, approaches and applications. *Knowledge-Based Systems*, 89:14–46.

Michael Roth and Mirella Lapata. 2016. Neural semantic role labeling with dependency path embeddings. In *Proceedings of ACL*, pages 1192–1202.

Sebastian Ruder. 2017. An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*.

Josef Ruppenhofer, Swapna Somasundaran, and Janyce Wiebe. 2008. Finding the sources and targets of subjective expressions. In *Proceedings of LREC*.

Emma Strubell, Patrick Verga, Daniel Andor, David Weiss, and Andrew McCallum. 2018. Linguistically-informed self-attention for semantic role labeling. In *Proceedings of EMNLP*, pages 5027–5038.

Kai Sheng Tai, Richard Socher, and Christopher D Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. In *Proceedings of ACL-IJCNLP*, pages 1556–1566.

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language resources and evaluation*, 39(2-3):165–210.

Qingrong Xia, Zhenghua Li, and Min Zhang. 2019a. A syntax-aware multi-task learning framework for chinese semantic role labeling. In *Proceedings of EMNLP-IJCNLP*, pages 5385–5395.

Qingrong Xia, Zhenghua Li, Min Zhang, Zhang Meishan, Guohong Fu, Rui Wang, and Luo Si. 2019b. Syntax-aware neural semantic role labeling. In *Proceedings of AAAI*, pages 7305–7313.

Bishan Yang and Claire Cardie. 2013. Joint inference for fine-grained opinion extraction. In *Proceedings of ACL*.

Bishan Yang and Claire Cardie. 2014. Joint modeling of opinion expression extraction and attribute classification. *Transactions of ACL*, 2:505–516.

Xiaohui Yu, Yang Liu, Xiangji Huang, and Aijun An. 2010. Mining online reviews for predicting sales performance: A case study in the movie domain. *IEEE Transactions on Knowledge and Data engineering*, 24(4):720–734.

Meishan Zhang, Zhenghua Li, Guohong Fu, and Min Zhang. 2019a. Syntax-enhanced neural machine translation with syntax-aware word representations. In *Proceedings of NAACL-HLT*, pages 1151–1161.

Meishan Zhang, Peili Liang, and Guohong Fu. 2019b. Enhancing opinion role labeling with semantic-aware word representations from semantic role labeling. In *Proceedings of NAACL-HLT*, pages 641–646.

Yue Zhang, Rui Wang, and Luo Si. 2019c. Syntax-enhanced self-attention-based semantic role labeling. In *Proceedings of EMNLP-IJCNLP*, pages 616–626.