Reinforcement Syntactic Dependency Tree Reasoning for Target-Oriented Opinion Word Extraction

Yaqing Dai, Pengfei Wang(✉), and Lei Zhang

Beijing University of Posts and Telecommunications, Beijing, China
{daiyaqing97,wangpengfei,zlei}@bupt.edu.cn

Abstract. Target-oriented Opinion Word Extraction (TOWE) is a subtask of Aspect Based Sentiment Analysis (ABSA), which aims to extract fine-grained opinion terms for a given aspect term from a sentence. In TOWE task, syntactic dependency tree is useful as it provides explanation to identify opinion terms to the given aspect term. It is necessary to mine relationships between aspect and opinion terms for a better performance. Previous works introduced syntactic dependency tree into TOWE task but lacked of explicit explanation. In this paper, we propose a novel model named **MM-TOWE**, which leverages **M**onte-Carlo tree search to enhance **M**arkov decision process (MDP) model for **T**arget-oriented **O**pinion **W**ord **E**xtraction task. We formulate TOWE task as an MDP of reasoning over the syntactic dependency tree. By learning the dependency relationships between aspect terms and opinion terms, our model can reason a path for an explicit explanation. Extensive experimental results illustrate that our proposed model outperforms the state-of-the-art methods.

Keywords: Target-oriented opinion word extraction · Reinforcement learning · Syntactic dependency tree

1 Introduction

Target-oriented Opinion Word Extraction (TOWE) is a subtask of Aspect Based Sentiment Analysis (ABSA). The main goal of TOWE is to extract the corresponding opinion terms from a sentence for a given aspect term (also called target). For an example sentence "*The sashimi is always fresh and the rolls are innovative and delicious.*", if the given aspect term is "rolls", the goal is to extract "innovative" and "delicious". Therefore, TOWE can be used in many applications such as sentiment analysis [7] and review summarizing [4,23].

Recently, some deep learning methods [2,21] formulate TOWE task as a sequence labeling problem. Although they have good performance, they mainly learn the relationships between aspect terms and opinion terms in sequential structure. Pouran Ben Veyseh et al. [14] show that syntactic structure is useful in TOWE task, and learn the relationships implicitly. In fact, a dependency

I. Farkaš et al. (Eds.): ICANN 2021, LNCS 12894, pp. 531–543, 2021.
https://doi.org/10.1007/978-3-030-86380-7_43

path can provide an explicit explanation to identify opinion terms to the given aspect term. As shown in Fig. 1, we can find a dependency path "rolls $\xrightarrow{nusbj^{-1}}$ innovative $\xrightarrow{conj^{-1}}$ delicious" to explain the relationship between aspect term and opinion terms of the example.

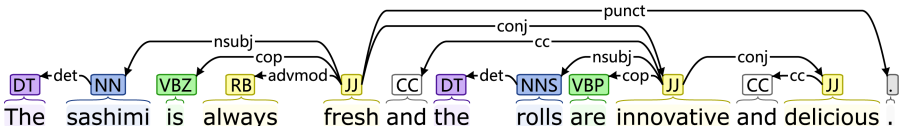

Fig. 1. The syntactic dependency tree of the example sentence - "*The sashimi is always fresh and the rolls are innovative and delicious*".

These years, reinforcement learning (RL) has been used widely in reasoning over large knowledge graph [17,22] for learning the relationships between two nodes. Therefore, RL seems to be a promising approach due to its great reasoning ability. However, many RL methods suffer from the challenges of large state space and sparse reward. AlphaGo Zero [19] uses Monte-Carlo Tree Search (MCTS) to enhance RL model for solving these two challenges during playing the game of Go. Inspired by these, we formulate TOWE task as a Markov decision process (MDP) of reasoning over the syntactic dependency tree. By learning the dependency relationships between aspect terms and opinion terms, our model can reason a path for an explicit explanation. Meanwhile, we also leverage MCTS to enhance our model.

In this paper, we propose a novel model named **MM-TOWE** to address TOWE task. We treat the syntactic dependency tree of a sentence as a graph and formulate TOWE task as an MDP. The agent starts moving from the first word of the given aspect term and reasons over this graph for opinion extraction. The reasoning path should be short and cover all opinion terms. To this end, we design a reward function to consider both of them in order to better guide the agent for reasoning path. We also leverage MCTS to enhance the model for solving the challenges of large state space and sparse reward. To evaluate the proposed model, we conduct extensive experiments on three datasets by comparing it with several competitive baselines. Experimental results show that our model can significantly outperform all the baselines in the TOWE task.

In summary, the contributions of our work are as follows:

- We formulate the TOWE task into an MDP of reasoning over syntactic dependency tree with MCTS for enhancing it.
- We design a reward function to consider both accuracy and efficiency.
- Experimental results on three real-world datasets show that our model can consistently outperform state-of-the-art baselines under different evaluation metrics and can find the dependency relationships between aspect terms and opinion terms.

2 Related Work

2.1 Target-Oriented Opinion Words Extraction

Target-oriented Opinion Words Extraction (TOWE) is a subtask of Aspect Based Sentiment Analysis (ABSA). The main goal of TOWE is to extract the opinion terms for a given aspect term from a sentence.

Early works which are related to TOWE mainly focus on doing sentiment analysis and review summarizing. There are rule-based methods which use distance rules [4] or mine high frequency rules on dependency tree [23] to extract opinion words for a given aspect word. However, these simple and effective rule-based methods can only deal with high frequency and common cases. As deep learning becomes more popular, TOWE task is formulated as a sequence labeling problem [2]. Wu et al. [21] transfer opinion knowledge from resource-rich sentiment classification task into TOWE task via an auxiliary learning signal. And Pouran Ben Veyseh et al. [14] use Graph Convolutional Networks (GCN) [6] to learn distance information from syntactic dependency tree for TOWE task.

2.2 Reinforcement Learning

Reinforcement learning (RL) is a field of machine learning and has been paid more attention by researchers. Meanwhile, reasoning over the knowledge graph with RL method also has been used widely in many areas such as recommendation [22], question and answer [1], and conversation system [9]. RL has great reasoning ability and can learn the relation rules from the start node to the target node. However, lots of works suffer from the problem of sparse reward or large state space when using RL method. Inspired by Alpha Go [18], some works [3,17] utilize Monte-Carlo Tree Search (MCTS) to enhance their models and solve these problems.

3 Methodology

In this section, we introduce our proposed MM-TOWE model. Let's denote the parameters of the model as θ.

3.1 Graph Definition

We formulate TOWE task as an MDP of walking on the syntactic dependency tree. As we show in Fig. 1, the syntactic dependency tree is a directed graph because the dependency relation edge has direction. Due to this, we add an inverse edge for each dependency relation edge, and then the agent can walk from a given word to any word in the sentence.

Specifically, for a given sentence $\mathcal{W} = \{w_1, w_2, \cdots, w_n\}$ where w_i represents the i-th word in the sentence and n is the number of words of the sentence. We use Stanza [15] to generate the syntactic dependency tree for sentence

$\mathcal{W}$ and convert the tree into a dependency tree graph $\mathcal{G}$. Let's denote graph $\mathcal{G} = \{(w_i, e_{ij}, w_j) | w_i, w_j \in \mathcal{W}, e_{ij} \in \mathcal{E}\}$, where w_i and w_j are two words in the given sentence $\mathcal{W}$ while $\mathcal{E}$ is an edge set of dependency relations. The words w_i and w_j are connected by a dependency relation edge e_{ij} according to the syntactic dependency tree of the sentence $\mathcal{W}$. Meanwhile if $(w_i, e_{ij}, w_j) \in \mathcal{G}$ and the dependency relation between word w_i and w_j is $e_{ij} = dep$, there is an inverse edge $e_{ji} = dep^{-1}$ and $(w_j, e_{ji}, w_i) \in \mathcal{G}$ (Fig. 2).

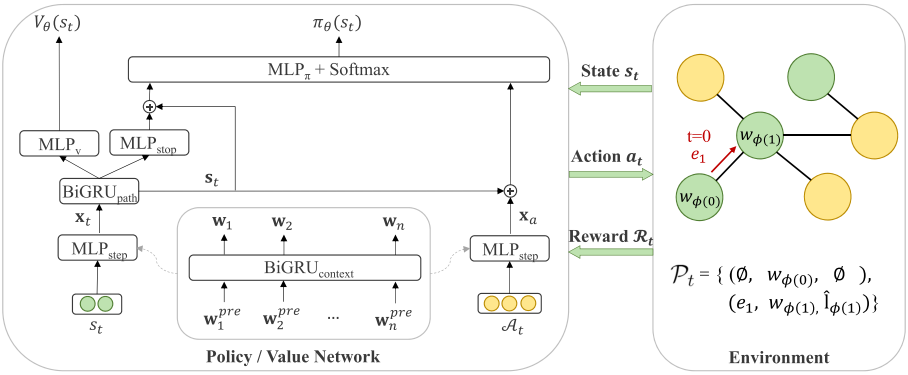

Fig. 2. The goal is to reason an explainable path from aspect term to opinion terms and predict correct label for every word to distinguish if it is an opinion word. The right part of the figure shows that the agent has chosen an action when $t = 0$ and move to word $w_{\phi(1)}$. After that, the agent will perform the next action by moving to a neighbor word through an edge and predicting the opinion label for the neighbor word.

3.2 An MDP Formulation of TOWE

We formulate the TOWE task as an MDP and let the agent reason over the dependency tree graph. The goal of TOWE task is to extract correct opinion terms from a sentence for a given aspect term. In the real cases, an aspect or opinion term may contain several words, while there may be more than one opinion term corresponding to one aspect term. Thus, the agent will learn relationships between these words on the graph, reason an explainable path to cover all opinion words and label them correctly. The model should know the position of the given aspect term in the sentence during reasoning. To this end, let's denote the information via an aspect label sequence $\mathcal{L}^a = \{l_1^a, l_2^a, \cdots, l_n^a\}$ by using BIO schema [16] ($\mathcal{L}$=$\{B$:beginning, I:inside, O:other$\}$) to label every word w_i with an aspect label l_i^a and point out the positions of the aspect words in the sentence. An opinion label sequence $\mathcal{L}^o$ which also contains opinion label l_i^o of every word is obtained in a similar way as the ground-truth.

With the information of the given sentence $\mathcal{W}$, its graph $\mathcal{G}$ and given aspect label sequence $\mathcal{L}^a$, the agent starts moving from the first word of aspect term.

For each time step, the agent chooses a neighbor word and move to it through an edge, then predicts its opinion label. When the agent chooses **STOP** action or the maximum time steps $\mathcal{T}_{max}$ has been reached, the agent stops moving over the graph and we can get the extraction result from the path with the opinion labels predicted by the agent. The states, actions, transition function, value function, and policy function of the MDP are defined as:

State $\mathcal{S}$: Let $s_t \in \mathcal{S}$ denote the state at time step t, where $\mathcal{S}$ is the state space. For each time step, the agent makes a decision based on all the information it can know, including the sentence $\mathcal{W}$, the graph $\mathcal{G}$, the given aspect label sequence $\mathcal{L}^a$ and history path $\mathcal{P}_t$. Therefore, we design the state as a tuple $s_t = (\mathcal{W}, \mathcal{G}, \mathcal{L}^a, \mathcal{P}_t)$, where $\mathcal{P}_t = \{(e_i, w_{\phi(i)}, \hat{l}^o_{\phi(i)}) | i \in [0, t]\}$ records all action information in the past (chosen edge e_i, chosen neighbor word $w_{\phi(i)}$, the predicted opinion label $\hat{l}^o_{\phi(i)}$ of $w_{\phi(i)}$) for each time step i. At time step t, the agent has reached word $w_{\phi(t)}$, where the $\phi(t)$ is the index of word $w_{\phi(t)}$ in the sentence $\mathcal{W}$. The agent starts from the first word of aspect term, thus the aspect label of $w_{\phi(0)}$ according to $\mathcal{L}^a$ must be B. And $s_0 = (\mathcal{W}, \mathcal{G}, \mathcal{L}^a, \mathcal{P}_0)$ where $\mathcal{P}_0 = \{(\varnothing, w_{\phi(0)}, \varnothing)\}$, because the agent hasn't taken any action. When the agent stops at time step $\mathcal{T}$, we get the terminal state $s_\mathcal{T}$ with the whole path $\mathcal{P}_\mathcal{T}$.

Action $\mathcal{A}$: At each time step t, $\mathcal{A}_t = \{(e, w, \hat{l}^o) | e \in \mathcal{E}, w \notin \{w_{\phi(0)}, w_{\phi(1)}, \cdots, w_{\phi(t)}\}, \hat{l}^o \in \mathcal{L}, (w_{\phi(t)}, e, w) \in \mathcal{G}\} \cup \{\textbf{STOP}\}$ is a set of possible actions according to s_t, while the whole action space is $\mathcal{A} = \cup \mathcal{A}_t$. The agent isn't allowed to go back to the word which already in the history path. If the agent performs action $a_t = (e_{t+1}, w_{\phi(t+1)}, \hat{l}^o_{\phi(t+1)}) \in \mathcal{A}_t$, it will move through an edge e_{t+1} from current word $w_{\phi(t)}$ to its neighbor word $w_{\phi(t+1)}$, and predict the opinion label $\hat{l}^o_{\phi(t+1)}$ for $w_{\phi(t+1)}$. When the agent thinks all opinion words are in the path, it will choose the **STOP** action to stop reasoning.

Reward $\mathcal{R}$: We employ the delayed reward strategy and design a reward function to consider both accuracy and efficiency to evaluate the whole path. Thus, the reward $\mathcal{R}_t$ is 0 during the reasoning process and there is a nonzero terminal reward $\mathcal{R}_\mathcal{T}$ to evaluate the whole path when the agent stops at time step $\mathcal{T}$. A high-quality path $\mathcal{P}_\mathcal{T}$ should cover all opinion words with all correct predicted labels (i.e. accuracy) based on a few actions (i.e. efficiency). To this end, we define the reward function as follows:

$$\mathcal{R}_{acc} = \frac{\hat{n}_o}{n_o} \cdot \frac{n_c}{\mathcal{T}}, \tag{1}$$

$$\mathcal{R}_{eff} = -\frac{\mathcal{T}}{\mathcal{T}_{max}}, \tag{2}$$

$$\mathcal{R}_\mathcal{T} = \mathcal{R}_{acc} + \mathcal{R}_{eff}, \tag{3}$$

where n_o is the number of opinion words according to the ground-truth $\mathcal{L}^o$, $\hat{n}_o$ is the number of hit opinion words in path $\mathcal{P}_\mathcal{T}$, n_c is the count of correct predicted labels in $\mathcal{P}_\mathcal{T}$. Please note that $\mathcal{T} = 0$ means the agent refuses to explore on the graph, therefore we define $\mathcal{R}_\mathcal{T} = -1$ for this special case.

Transition function T**:** The transition function $T : \mathcal{S} \times \mathcal{A} \rightarrow \mathcal{S}$ is defined that: $s_{t+1} = T(s_t, a_t) = (\mathcal{W}, \mathcal{G}, \mathcal{L}^a, \mathcal{P}_t \cup \{(e_{t+1}, w_{\phi(t+1)}, \hat{l}^o_{\phi(t+1)})\})$, where means state records the information of action $a_t = (e_{t+1}, w_{\phi(t+1)}, \hat{l}^o_{\phi(t+1)})$.

Value Function V_θ**:** The value function $V_\theta : \mathcal{S} \rightarrow \mathbb{R}$ is a scalar evaluation. It learns to estimate the terminal reward $\mathcal{R}_\mathcal{T}$ for evaluating the quality of the whole path (an episode) based on the input state s_t. The same word in different sentences may represent different meanings. Thus, we input the sentence into a Bi-directional Gated Recurrent Unit (BiGRU) for encoding:

$$\mathbf{w}_i = \text{BiGRU}(\mathbf{w}_{i-1}, \mathbf{w}^{pre}_i; \theta_{context}), \tag{4}$$

where $\mathbf{w}^{pre}_i$ is the pretrained word embedding of word w_i, and $\mathbf{w}_i$ is the new word embedding of word w_i which contains the context information, $\theta_{context}$ denotes all the related parameters of the BiGRU network. We leverage a Multi-Layer Perceptrons (MLPs) to compress the information of every time step i in history path $\mathcal{P}_t = \{(e_i, w_{\phi(i)}, \hat{l}^o_{\phi(i)}) | i \in [0, t]\}$ and the aspect label $l^a_{\phi(i)}$ of word $w_{\phi(i)}$. Then, put them into a BiGRU to calculate the current state representation $\mathbf{s}_t$ and the value $V_\theta(s_t)$ is calculated by $\mathbf{s}_t$.

$$\mathbf{x}_t = \text{MLP}(\mathbf{e}_t \oplus \mathbf{w}_{\phi(t)} \oplus \mathbf{l}^o_{\phi(t)} \oplus \mathbf{l}^a_{\phi(t)}; \theta_{step}), \tag{5}$$

$$\mathbf{s}_t = \text{BiGRU}(\mathbf{s}_{t-1}, \mathbf{x}_t; \theta_{path}), \tag{6}$$

$$V_\theta(s_t) = \text{MLP}(\mathbf{s}_t; \theta_v), \tag{7}$$

where $\mathbf{e}_t$ is the embedding of edge, $\mathbf{w}_{\phi(t)}$ is the new embedding of word $w_{\phi(t)}$, $\mathbf{l}^o_{\phi(t)}$ is the embedding of predicted opinion label of $w_{\phi(t)}$, $\mathbf{l}^a_{\phi(t)}$ is the embedding of aspect label of $w_{\phi(t)}$, $\oplus$ denotes the concatenation operator, θ_{path} denotes all the related parameters of the BiGRU network, and θ_{step} and θ_v are parameters of two MLPs.

Policy π_θ**:** The policy π_θ calculates the probability distribution of all actions $a \in \mathcal{A}_t$ based on the state s_t. To this end, we calculate the representation $\mathbf{x}_a$ for every action. As we mentioned above the agent will select **STOP** action when it thinks the history path $\mathcal{P}_t$ covers all opinion words based on the state s_t. Thus, we leverage $\mathbf{s}_t$ to calculate $\mathbf{x}_a$ of the **STOP** action:

$$\mathbf{x}_a = \text{MLP}(\mathbf{s}_t; \theta_{stop}), \tag{8}$$

where θ_{stop} represents the parameters of MLP. The representation $\mathbf{x}_a$ of action $a = (e, w, \hat{l}^o)$ with the new word embedding can be written as:

$$\mathbf{x}_a = \text{MLP}(\mathbf{e} \oplus \mathbf{w} \oplus \mathbf{l}^o \oplus \mathbf{l}^a; \theta_{step}), \tag{9}$$

where $\mathbf{e}, \mathbf{w}, \mathbf{l}^o$ are the corresponding embeddings of the each element e, w, $\hat{l}^o$ of action a, $\mathbf{l}^a$ is the embedding of aspect label of word w and we also share the same parameters θ_{step} which is used in it formula (5). The probability of an

action should consider both the state representation $\mathbf{s}_t$ and action representation $\mathbf{x}_a$, for any action $a \in \mathcal{A}_t$, we calculate its probability by a softmax function:

$$\pi_\theta(a|s_t) = \frac{\exp\{\text{MLP}(\mathbf{s}_t \oplus \mathbf{x}_a; \theta_\pi)\}}{\sum_{a' \in \mathcal{A}_t} \exp\{\text{MLP}(\mathbf{s}_t \oplus \mathbf{x}_{a'}; \theta_\pi)\}}, \tag{10}$$

where θ_π denotes as the parameters in the MLP.

3.3 Enhancing Policy by MCTS

Due to the sparse reward and large space, it is hard to sample good path at the beginning of training. Following AlpahGo Zero [19], we leverage MCTS to make a heuristic search in the whole space by using our value function V_θ and policy π_θ to get a better policy π_e. There are four steps in MCTS:

Selection: Starting from the root node s_R each time, MCTS recursively selects the child nodes by using the following function until reaches a leaf node:

$$a_t = \text{argmax}_a(Q(s_t, a) + c_{puct} P(a|s_t) \frac{\sqrt{\sum_{a' \in \mathcal{A}_t} N(s_t, a')}}{1 + N(s_t, a)}), \tag{11}$$

where $Q(s_t, a)$ is an action value, c_{puct} is a hyper parameter to control the level of exploration of MCTS, $P(a|s_t)$ is a prior probability, and $N(s_t, a)$ is the visit count of the node. MCTS prefers choosing the node with small $N(s_t, a)$ at first. After simulating several times, it prefers choosing the node with higher $Q(s_t, a)$.

Evaluation: When reaching a leaf node s_t, we will estimate the value of this node. If s_t is a terminal node (i.e., the agent stops reasoning), we use the terminal reward $\mathcal{R}_\mathcal{T}$ calculated by formula (1)-(3), else we use our value function $V_\theta(s_t)$ for estimating.

Expansion: If s_t is not a terminal leaf node, we will expand the tree by adding all child nodes (corresponding to every action $a \in \mathcal{A}_t$) for node s_t. Initializing the elements of each new child node as $P(a|s_t) = \pi_\theta(a|s_t)$, $Q(s_t, a) = 0$, $N(s_t, a) = 0$.

Backup: We recursively backup the elements $Q(s, a)$, $N(s, a)$ of all nodes from the leaf node s_t to the root node s_R according to the path $\mathcal{P}_t \in s_t$ by:

$$Q(s, a) \leftarrow \frac{V(s) + Q(s, a) \times N(s, a)}{N(s, a) + 1}, \tag{12}$$

$$N(s, a) \leftarrow N(s, a) + 1. \tag{13}$$

After reaching the maximum simulation time, we randomly choose the action according to the probabilities evaluated by searching policy π_e. A softmax function with temperature τ is used to get the probability of every action in policy π_e by using the visit count $N(s, a)$.

$$\pi_e(a_t|s_t) = \frac{\exp\{N(s_t, a_t)^{1/\tau}\}}{\sum_{a' \in \mathcal{A}_t} \exp\{N(s_t, a')^{1/\tau}\}}. \tag{14}$$

3.4 Training and Test

We use a mean square error for value function V_θ to mimic the terminal reward $\mathcal{R}_\mathcal{T}$ and a cross entropy loss for policy π_θ to mimic the searching policy π_e. The loss function $\mathcal{L}_\theta$ can be written as:

$$\mathcal{L}_\theta = (\mathcal{R}_\mathcal{T} - V_\theta(s_t))^2 - \pi_e(s_t)^\top log\pi_\theta(s_t) + \rho \left\|\theta\right\|^2, \tag{15}$$

where ρ is a parameter controlling the level of l_2 weight regularization.

After training, we also use MCTS to search a policy π by using V_θ and π_θ for testing and there are several differences. We use our value function $V_\theta(s_t)$ for estimating the value of leaf node s_t all the time because we don't know the ground-truth to calculate terminal reward $\mathcal{R}_\mathcal{T}$ during test time. And we select action $a_t = \text{argmax}_{a \in \mathcal{A}_t} \pi(a|s_t)$ for each step.

When reaching a terminal state $s_\mathcal{T}$, we generate a predicted opinion label sequence $\hat{\mathcal{L}}^o = \{\hat{l}^o_1, \hat{l}^o_2, \cdots, \hat{l}^o_n\}$ based on $\mathcal{P}_\mathcal{T} = \{(e_i, w_{\phi(i)}, \hat{l}^o_{\phi(i)}), i \in [0, T]\}$. For the words in sentence $\mathcal{W}$ which not exists in the path $\mathcal{P}_\mathcal{T}$, we assign the label O to these words. Then we use $\hat{\mathcal{L}}^o$ to extract opinion terms from sentence $\mathcal{W}$.

4 Experiment

4.1 Datasets and Metrics

We use three widely used datasets generated by Fan et al. [2] to evaluate our model. The 14lap is derived from the SemEval challenge 2014 Task4 [11], 15res is from SemEval challenge 2015 Task12 [13] and 16res is from SemEval challenge 2016 Task5 [12]. The suffixes "res" and "lap" mean that the reviews are from restaurant domain and laptop domain, respectively. We randomly sample 20% of the training data for validation.

We use the metrics precision, recall, and F1 score to measure the performance of baselines and our model. An opinion term is considered to be a correct prediction when the position (i.e., the beginning and the ending offset) of the term as well as the labels of the term are both predicted correctly.

4.2 Settings

We initialize word embedding vectors (i.e. $\mathbf{w}_i^{pre}$ in formula (4)) with 300 dimension Glove [10] vectors which are pretrained on 840 billion words and fix them during training. The dimension of edge embedding vectors is 50 and the dimension of label embedding vectors is 10. We randomly initialize the edge and label embedding vectors. The dimension of hidden states in context encoding BiGRU is set as 150 and the dimension of hidden states in BiGRU which calculates state

representation is set as 50. We use Adam [5] as the optimizer and set learning rate $5e^{-4}$. The l_2 weight regularization ρ is $1e^{-5}$, hyper parameter c_{puct} in MCTS is 5.

4.3 Baselines

We compare our model with the following baselines:

Distance-rule [4]: it uses POS tags and distances by choosing the nearest adjective to the aspect term as the opinion term.

Dependency-rule [23]: it learns the dependency paths with POS tags from aspect word to opinion word, then uses the high frequency dependency templates to extract from the test data.

LSTM/BiLSTM [8]: this approach employs word embeddings to represent words, put them into a LSTM or BiLSTM, and makes a 3-class classification for every hidden state. It's a sentence-level opinion words extraction.

Pipeline [2]: it combines BiLSTM and distance rule. After getting the result of BiLSTM, choose the nearest opinion term to the aspect term as the final result.

TC-BiLSTM: this method follows the design of the work for target-oriented sentiment classification [20]. It uses the average embedding of the aspect term as the aspect vector, and concatenates it to every word embedding of the sentence. Then put them into BiLSTM to do sequence labeling.

IOG [2]: the authors use six different positional and directional LSTMs to extract opinion terms of the aspect term.

LOTN [21]: it transfers sentiment classification task into TOWE task to gather more opinion knowledge via an auxiliary learning signal.

ONG [14]: this method introduces distance information of syntactic structure into extraction. It employs BERT to get word embeddings in the paper. We reproduce the model by using Glove as word embeddings and stanza to generate syntactic dependency tree for a fair comparison.

4.4 Results

Table 1 shows the performance of our model and baselines. We can observe that our model performs best among all baselines on three datasets.

Table 1. Main experiment results(%). Best results are in bold (for P, R, and F1 score, the larger is the better).

Model	14lap			15res			16res		
	P	R	F1	P	R	F1	P	R	F1
Distance-rule	50.13	33.86	40.42	54.12	39.96	45.97	61.90	44.57	51.83
Dependency-rule	45.09	31.57	37.14	65.49	48.88	55.98	76.03	56.19	64.62
LSTM	55.71	57.53	56.52	57.27	60.69	58.93	62.46	68.72	65.33
BiLSTM	64.52	61.45	62.71	60.46	63.65	62.00	68.68	70.51	69.57
Pipeline	72.58	56.97	63.83	74.75	60.65	66.97	81.46	67.81	74.01
TC-BiLSTM	62.45	60.14	61.21	66.06	60.16	62.94	73.46	72.88	73.10
IOG	73.24	69.63	71.35	76.06	70.71	73.25	82.25	78.51	81.69
LOTN	77.08	67.62	72.02	76.61	70.29	73.29	86.57	80.89	83.62
ONG(Glove)	78.55	68.17	72.75	79.30	73.02	76.03	88.09	81.71	84.78
MM-TOWE(Our)	**81.24**	**69.49**	**74.90**	**81.00**	**75.25**	**78.02**	**89.02**	**83.43**	**86.14**

The unsupervised rule-based methods perform poorly on all datasets. Although the Dependency-rule is better than Distance-rule, it is still limited by its quality of the rules and worse than the deep-learning methods. LSTM and BiLSTM perform not well in the task because they will extract the same opinion terms for different aspect terms in the sentence. They can't extract for a special aspect term. The Pipeline combines BiLSTM and distance rule. It extracts the nearest opinion term of the aspect term and obtains a high performance as compared with LSTM/BiLSTM. TC-LSTM performs worse than Pipeline because it ignores the position of the aspect term. IOG is better than other baselines, but it suffers from the high model complexity and no supplementary information. LOTN transfers opinion knowledge from sentiment classification into TOWE and get better results, but it needs external information. ONG leverages the distance information of the dependency tree and it performs better than other baselines. However, ONG ignores the dependency relations in the syntactic structure, which results in a sub-optimal performance. Our MM-TOWE model gets great improvement and better than all baselines. The results verify the effectiveness of leveraging RL method to reason over the syntactic dependency tree and learn the dependency relationships.

4.5 Ablation Study

In order to learn the effects of different parts of our model. We compare MM-TOWE with the following variations: (i) **no context information**: we remove the BiGRU used in formula (4) from our model. We only use the pretrained word embeddings and not to aggregate the context information. (ii) **test without MCTS**: we use MCTS during training but don't use MCTS during test. We only use our policy π_θ and choose action with the maximal probability at each time when testing.

Table 2. Experiment results(%) of ablation study.

Model	14lap			15res			16res		
	P	R	F1	P	R	F1	P	R	F1
No context information	74.85	65.61	69.92	76.00	69.37	72.53	83.74	78.48	81.02
Test without MCTS	80.17	69.35	74.37	80.57	74.85	77.60	88.37	82.48	85.32
MM-TOWE(Our)	**81.24**	**69.49**	**74.90**	**81.00**	**75.25**	**78.02**	**89.02**	**83.43**	**86.14**

From Table 2, we can know that context information and testing with MCTS are all significant for opinion extraction. The performance will drop apparently when we remove any part of the model.

4.6 Case Study and Path Analysis

Table 3 shows the results of our method MM-TOWE and the best performing baseline ONG on some cases and we analyze the strengths and weaknesses of them. The first case shows that ONG only extracts the opinion word "easy" while neglect the opinion word "intuitive". However, in the second cases, ONG predicts a wrong opinion word "fast" which is not the opinion term of the aspect "graphics". In the third case, ONG extracts "not" and "enjoy" but the boundary of the opinion term is wrong. In contrast, our model correctly extracts all opinion terms for the three cases. In Table 4, we show the reasoning paths generated by our model for every sentence in Table 3.

Table 3. Cases of the extracted results of our method and the best performing baseline method (ONG). The aspect terms are red and the corresponding opinion words are blue.

SENTENCE	ONG	MM-TOWE
Everything is so easy and intuitive to setup or configure.	easy	easy, intuitive
It is super fast and has outstanding graphics.	fast & outstanding	outstanding
Did not enjoy the new Windows 8 and touchscreen functions.	not & enjoy	not enjoy

Table 4. The reasoning paths generated by our model for each sentence in Table 3. The relation types of chosen edge are shown above the arrows and the opinion labels predicted by the agent are shown in the parentheses behind the word.

configure $\xrightarrow{conj^{-1}}$ setup(O) $\xrightarrow{ccomp^{-1}}$ easy(B) $\xrightarrow{conj^{-1}}$ intuitive(B) $\longrightarrow$ **STOP**
graphics $\xrightarrow{amod}$ outstanding(B) $\longrightarrow$ **STOP**
Windows $\xrightarrow{obj^{-1}}$ enjoy(I) $\xrightarrow{advmod}$ not(B) $\longrightarrow$ **STOP**

5 Conclusion and Future Work

In this paper, we propose a novel deep reinforcement learning model for the TOWE task. To the end, we formulate the extraction task into a Markov Decision Process (MDP) of reasoning over the syntactic dependency tree. The model learns the relationships between aspect terms and opinion terms, and reasons a path to explain the result of the extraction. To better guide the reasoning process, we design a reward function to consider both accuracy and efficiency. Experimental results on three widely used datasets show that our model consistently outperforms all baselines. As transformer and BERT become more popular, future studies could fruitfully explore TOWE task further by using these structures to improve our model.

References

1. Das, R., et al.: Go for a walk and arrive at the answer: Reasoning over paths in knowledge bases using reinforcement learning. In: ICLR (2017)
2. Fan, Z., Wu, Z., Dai, X.Y., Huang, S., Chen, J.: Target-oriented opinion words extraction with target-fused neural sequence labeling. In: NAACL-HLT, pp. 2509–2518 (2019)
3. Feng, Y., Xu, J., Lan, Y., Guo, J., Zeng, W., Cheng, X.: From greedy selection to exploratory decision-making: Diverse ranking with policy-value networks. In: SIGIR, pp. 125–134 (2018)
4. Hu, M., Liu, B.: Mining and summarizing customer reviews. In: ACM SIGKDD, pp. 168–177 (2004)
5. Kingma, D., Ba, J.: Adam: A method for stochastic optimization. In: International Conference on Learning Representations (12 2014)
6. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24–26, 2017, Conference Track Proceedings (2017)
7. Liu, B.: Sentiment analysis and opinion mining. In: Synthesis Lectures on Human Language Technologies (2012)
8. Liu, P., Joty, S., Meng, H.: Fine-grained opinion mining with recurrent neural networks and word embeddings. In: EMNLP, pp. 1433–1443 (2015)
9. Moon, S., Shah, P., Kumar, A., Subba, R.: Opendialkg: Explainable conversational reasoning with attention-based walks over knowledge graphs. In: Proceedings of the 57th ACL, pp. 845–854 (2019)
10. Pennington, J., Socher, R., Manning, C.: Glove: Global vectors for word representation. In: EMNLP, pp. 1532–1543 (2014)
11. Pontiki, M., Galanis, D., Pavlopoulos, J., Papageorgiou, H., Androutsopoulos, I., Manandhar, S.: Semeval-2014 task 4: Aspect based sentiment analysis. In: SemEval, pp. 27–35 (2014)
12. Pontiki, M., et al.: Semeval-2016 task 5: Aspect based sentiment analysis. In: SemEval, pp. 19–30 (2016)
13. Pontiki, M., Galanis, D., Papageorgiou, H., Manandhar, S., Androutsopoulos, I.: SemEval-2015 task 12: aspect based sentiment analysis. In: SemEval, pp. 486–495 (2015)

14. Pouran Ben Veyseh, A., Nouri, N., Dernoncourt, F., Dou, D., Nguyen, T.H.: Introducing syntactic structures into target opinion word extraction with deep learning. In: EMNLP, pp. 8947–8956 (2020)
15. Qi, P., Zhang, Y., Zhang, Y., Bolton, J., Manning, C.D.: Stanza: a python natural language processing toolkit for many human languages. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations (2020). https://nlp.stanford.edu/pubs/qi2020stanza.pdf
16. Ramshaw, L., Marcus, M.: Text chunking using transformation-based learning. In: Third Workshop on Very Large Corpora (1995)
17. Shen, Y., Chen, J., Huang, P.S., Guo, Y., Gao, J.: M-walk: Learning to walk over graphs using monte carlo tree search. In: NeurIPS, pp. 6787–6798 (2018)
18. Silver, D., et al.: Mastering the game of go with deep neural networks and tree search. Nature **529**(7587), 484–489 (2016)
19. Silver, D., et al.: Mastering the game of go without human knowledge. Nature **550**(7676), 354–359 (2017)
20. Tang, D., Qin, B., Feng, X., Liu, T.: Effective LSTMs for target-dependent sentiment classification. In: COLING, pp. 3298–3307 (2016)
21. Wu, Z., Zhao, F., Dai, X.Y., Huang, S., Chen, J.: Latent opinions transfer network for target-oriented opinion words extraction. In: AAAI, pp. 9298–9305 (2020)
22. Xian, Y., Fu, Z., Muthukrishnan, S., de Melo, G., Zhang, Y.: Reinforcement knowledge graph reasoning for explainable recommendation. In: ACM SIGIR, pp. 285–294 (2019)
23. Zhuang, L., Jing, F., Zhu, X.Y.: Movie review mining and summarization. In: CIKM, pp. 43–50 (2006)