Aspect-based Sentiment Classification via Reinforcement Learning

Lichen Wang[1], Bo Zong[2], Yunyu Liu[1], Can Qin[1], Wei Cheng[2],
Wenchao Yu[2], Xuchao Zhang[2], Haifeng Chen[2], Yun Fu[1]
[1]*Northeastern University, Boston, USA*, [2]*NEC Laboratories America, Princeton, USA*
{wang.lich, liu.yuny, qin.ca}@northeastern.edu, lerry.z@gmail.com,
{weicheng,wyu,xuczhang, heifeng}@nec-labels.com, yunfu@ece.neu.edu

***Abstract*—Aspect-based sentiment classification aims to predict sentimental polarities of one or multiple aspects in texts. As texts always contain a large proportion of task-irrelevant words, accurate alignment between aspects and their sentimental descriptions is the most crucial and challenging step. State-of-the-art approaches are mainly based on word-level attention learned from recurrent neural network variants (e.g., LSTM) or graph neural networks. From another view, these methods essentially weight and aggregate all possible alignments. However, this mechanism heavily relies on large-scale supervision training: without enough labels, it could easily overfit with difficulty in generalization. To address this challenge, we propose SentRL, a reinforcement learning-based framework for aspect-based sentiment classification. In this framework, input texts are transformed into their dependency graphs. Then, an agent is deployed to walk on the graphs, explores paths from target aspect nodes to their potential sentimental regions, and differentiates the effectiveness of different paths. By limiting the agent's exploration budget, our method encourages the agent to skip task-irrelevant information and focus on the most effective paths for alignment purpose. Our method considerably reduces the impact of task-irrelevant words and improves generalization performance. Compared with competitive baseline methods, our approach achieves the highest performance on public benchmark datasets with up to 3.7% improvement.**

***Index Terms*—Natural Language Processing, Sentiment Classification, Reinforcement Learning**

I. Introduction

The goal of aspect-based (or aspect-level) sentiment classification is to predict the sentiment polarities of individual aspects. As shown in Figure 1, given a sentence *I like this computer but do not like the screen*, the sentiment of the aspect *computer* is positive because of *like*. Meanwhile, the sentiment of the aspect *screen* is negative for *do not like*. Compared with conventional sentiment classification task, aspect-level scenario is a more fine-grained and challenging task, where the core problem is to correctly align aspects with their sentiment descriptions. State-of-the-art methods rely on supervision signals to automatically learn such alignment. By leveraging textual context and word-level attention learned from deep models [1], [2], existing methods have made great progress on discovering aspect-specific sentimental statements.

Meanwhile, the existing methods could suffer serious overfitting problems, as natural language inevitably includes a large proportion of task-irrelevant texts, or noise from the perspective of machine learning. Ideally, with a sufficient amount of training labels, the existing methods could effectively contain the negative impact of such task-irrelevant information. In practice, because of the high variance in language expression, it is costly to collect a large number of task-specific labels, and it is difficult to guarantee the expected label sufficiency. With limited labels, the existing approaches could easily include task-irrelevant information into decision processes, overfit training data, and end up with inferior generalization performance to unseen data.

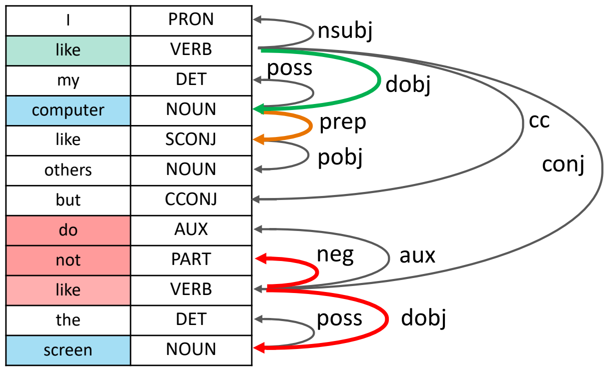

Fig. 1. Dependency graph of a given sentence. **Blue** words are the aspects (i.e., *computer* and *screen*). **Green** and **red** are *positive* and *negative* sentiment respectively. The dependency graph effectively reduces the distances between aspects and sentimental descriptions and avoids polysemy words (e.g., *like*). In our approach, an agent is deployed to walk from the aspect word to the sentimental regions, which avoids task-irrelevant information and achieves more effective and efficient performance.

To effectively contain the impact from task-irrelevant information, we propose SentRL, a reinforcement learning based framework for aspect-level sentiment classification. In our approach, input texts are firstly transformed into graph objects (e.g., dependency graphs [3]), where nodes are words and edges indicate syntactic dependencies/relations between them. Next, we deploy a policy-based agent to discover aspect-related sentiment descriptions in the graphs. This agent is geared with a language understanding module so that it is able to update exploration states and make sentiment decisions for individual aspects. Unlike existing methods that aggregate potential sentiment information from all possible textual contexts or words, our agent strives to leverage the most relevant exploration paths under a limited budget. This strategy not only requires the agent to focus on the most effective paths

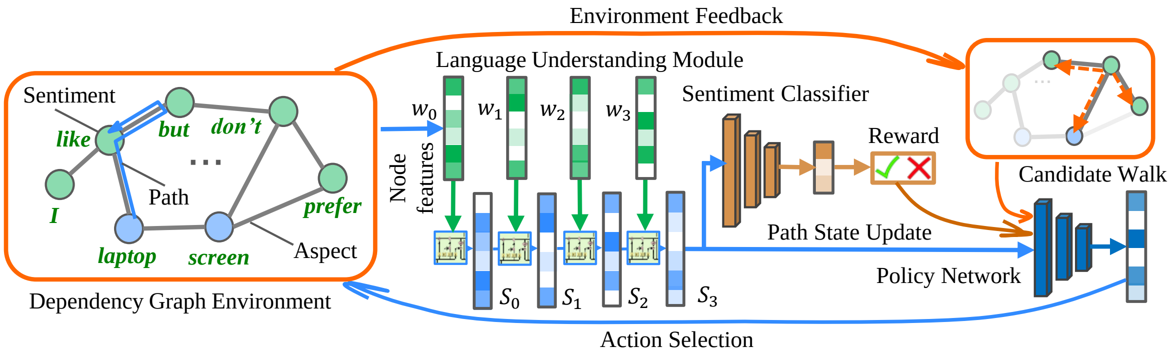

Fig. 2. Framework of our SentRL method. A dependency graph is first obtained. The related sentiment clues could be more close in dependency graph compared with the original sentence space. An agent starts from an aspect node and walk though the graph edges. The walk direction is determined by a policy network which utilizes the current walk state and the direction candidates for making the decision. A semantic understanding module is used to update path states and a sentiment classifier is designed to have the final sentiment prediction based on the walk state.

but also encourages the agent to skip task-irrelevant regions. Using standard back-propagation methods, the policy network and the language understanding module are jointly trained. From public benchmark datasets, we observe our method could achieve up to 3.6% improvement compared with competitive state-of-the-art methods. The main contributions of our work are listed below.

- A novel reinforcement learning framework for aspect-based sentiment classification is proposed. It accurately pinpoints the most effective path between sentiment descriptions and the target aspects, and effectively avoids the impact of the task-irrelevant regions.
- A policy network is developed to provide an agent with exploration guidance. This network iteratively provides suggestions on the next-hop selection. In particular, the framework is permutation invariant and guarantees the consistency and reliability of the model.
- A language understanding module is developed to help an agent "remember" its exploration history and make the final sentiment prediction.

To the best of our knowledge, this is the first work to explore the effectiveness of reinforcement learning in sentimental dependency graph and achieves state-of-the-art performance. We consider our approach is a more human-like mechanism to explore knowledge and eliminate irrelevant information.

II. Related Work

A. Aspect-based Sentiment Classifications

Aspect-based sentiment classification is to identify sentiment polarities of one or more aspects in given texts [4]. Aspects could be either substantial objects (e.g., *computer*) or conceptional objects (e.g., *service*). There are usually three sentiment categories including *positive*, *neutral*, and *negative*. Conventional approaches [5] treat input texts as word sequences, and deploy separate feature extraction modules as well as classification modules. Deep learning-based methods [6] take contextual information regarding the word order into consideration by using LSTM. [7] proposed a deep memory network which explicitly captures the importance of each context word. An attention-over-attention module [8] is proposed to improve the conventional attention strategy. [9] introduced another attention-over-attention module which jointly models aspects and sentences explicitly. [10] proposed a new model which employs a CNN layer to extract salient features from the transformed word representations. While such approaches require a large amount of training labels with non-trivial variance [11] (e.g., long sentences with the majority of irrelevant contextual words).

B. Sentiment Analysis on Graphs

Syntactic dependency tree [3] is a widely adopted data structure that encodes syntactic dependencies between words in input texts. Aspect-based sentiment classification problems can be cast into node classification problems. Graph structured data is widely explored for various learning applications [12], [13]. For instance, [2] deploys graph neural network to aggregate the sentiment information. Transformer based approach is proposed in [14] to further improve the performance. [15] explores the grammatical aspect of the sentence and employs the self-attention mechanism for syntactical learning. [16] presents a neat and effective multiple CRFs based structured attention model. However, these models usually perform under transductive settings, which means the complete sentence or paragraph should be given.

C. Reinforcement Learning in NLP

Reinforcement Learning (RL) [17] is a promising approach which automatically and actively explores the environment and achieves the optimized strategy for final tasks [18]–[22]. RL has been explored in NLP related interactive tasks such as text-based games [18] and question answering [19]. DeepPath [23] aims to find reasoning paths and the action space is the relation space in the knowledge graph. [24] proposed a novel Hierarchical RL approach which simulating the steps of analyzing aspect sentiment in a document. [25] proposed a visual reasoning framework which consists of a program generator that constructs an explicit representation of the reasoning process. Unlike previous works, we are the first to study RL methods that intelligently collect information from syntactic dependency graphs for aspect-based sentiment analysis. It is a more challenging task since only limited supervision information is provided in a large amount of action space for agent.

III. Our Approach

A. Preliminary

Given the target texts (e.g., a review) $c = \{w_1^c, w_2^c, w_3^c, ..., w_n^c\}$, where w_i^c represents the i-th word in c. n is the total number of the words. The target aspect of c is denoted as $a_c = \{w_{r+1}^c, ..., w_{r+m}^c\}$, where $r+1$ indicates the start location of the aspect and m is the length of the aspect. The aspect is either a single-word format (e.g., *computer*, *service*, and *screen*) or multiple-word format (e.g., *HDMI port* and *sport mode*). There could be one or more aspects in c, and different aspects could have different or opposite sentimental categories. The goal of aspect-based sentiment classification is to recover the sentiment polarities (i.e., *positive*, *neutral*, and *negative*) for each given aspect.

B. Dependency Graph Extraction

Dependency graph is a widely adopted data structure that encodes syntactic dependencies between words in input texts. Given a description c, a dependency graph $G_c = (V(G_c), E(G_c))$ is extracted from c. $V(G_c)$ indicates all vertices/nodes in G_c, and each vertice corresponds to a word in c. $E(G_c)$ denotes the edges of G_c. Each edge in G_c represents the syntactical relation categories. By deploying the dependency graph, we transfer the sequential text data into graph-structured format. In G_c, the long-distance pair of words in texts could be close in the graph. This provides extra syntactical knowledge and makes down-stream algorithms easier to pinpoint the sentimental words. We deploy the existing graph parsing algorithm to obtain the graph.

C. Path Searching via Reinforcement Learning

Compared with other graph learning algorithms (e.g., GCN and attention), we proposed a RL framework to explore the most effective aspect-sentiment path on the dependency graph. First, an agent starts walking from the target aspect node. Then, a policy network selects the most effective walk based on the previous walk history and all feasible walk candidates. Next, a semantic understanding module is deployed to aggregate the path state and a sentiment classifier is used to obtain the final sentiment prediction. In summary, there are three components in our framework, 1) A **Semantic Understanding Module** which aggregates the comprehensive walk state, 2) A **Policy Network** which makes the walk decision, and 3) A **Sentiment Classifier** which obtains the final prediction of the sentiment categories.

D. Semantic Understanding Module

The semantic understanding module aims to provide a comprehensive state of the walk for 1) allowing the policy network to make the effective walk action and 2) letting the sentiment classifier obtain a final prediction. There is a requirement that the state updating mechanism should make the walking procedure as a Markov Decision Process (MDP). The expression is shown below:

$$P(S_{t+1}|S_0, A_0, ..., S_t, A_t) = P(S_{t+1}|S_t, A_t), \qquad (1)$$

where S_i, A_i, and R_i are the state, action, and reward of the i-th move respectively. Eq. (1) indicates that the path state of $(t+1)$-th move, S_{t+1}, should be only relevant to the current state S_t and action A_t, and irrelevant to earlier states. To this end, S_t is required to preserve both the current and all previous walk information. To achieve this goal, we deploy the general LSTM structure in our framework, as LSTM and its varieties have been well validated as an effective way to capture both the feature and the sequential knowledge in a given sample. In our model, the function is shown below:

$$S_t = \text{LSTM}(A_t, S_{t-1}), \qquad (2)$$

where S_t is the current state, which could also be considered as the hidden state updated in each loop. A_t is the node feature of the corresponding t-th action (i.e., word embedding) which is walked though by agent in the t-th move. Semantic understanding module keeps updating S_t for each walk.

E. Policy Network

The goal of the policy network is to guide the agent to find the most effective paths in the obtained dependency graph. Specifically, it is designed to select the action based on the previous walk path and the next move candidates. There are two challenges. First, the policy network should perceive all possible candidates for making the most effective action. General deep network structure requires consistent data format as input. While, in G_c, each node could have a different number of neighbor (1-hop) nodes. Second, the policy network should be permutation irrelevant to the candidate input.

To this end, we proposed a structural input mechanism. When the agent has arrived at a node, it obtains all the connected nodes (including the previously walked nodes and itself). We assume the set of the nodes is $V = \{v_1, v_2, ..., v_m\}$, where m is the number of all candidate nodes around i-th nodes. We concatenate current state S_t with each node $s_i, i = \{1, 2, ..., m\}$ and obtain a vector. We put the vector into the policy network to get a candidate score of the node v_i. The expression is formulated as shown below:

$$s_i = \text{policy}(\text{cat}(S_t, v_i)), \quad i = \{1, 2, ..., m\}, \qquad (3)$$

where cat$(\cdot)$ indicates the concatenation operation. This strategy considers the probability of each candidate separately and it relaxes the input inconsistency challenge while still preserves the local structural information of the dependency graph. When the scores of all candidates are obtained, the agent would go to the node which has the highest score. The function is shown below:

$$a = \arg\max_{i=\{1,2,...,m\}}(s_i). \qquad (4)$$

A consistent walk length is set to stop the walk process. In our experiments, we set walk length of 3 which is effective

TABLE I
STATISTICS OF THE EVALUATION DATASETS

Dataset	Split	Positive	neutral	Negative
Lap14	Training	994	464	870
	Testing	341	169	128
Twitter	Training	1561	3127	1560
	Testing	173	346	173
Rest14	Training	2164	637	807
	Testing	728	196	196
Rest15	Training	912	36	256
	Testing	326	34	182
Rest16	Training	1240	69	439
	Testing	469	30	117

enough for achieving high performance. Details are discussed in Section IV-C. Other potential solutions such as setting up a stop action to terminate the walk process, while we found the current strategy is the most efficient one with less complicity.

F. Sentiment Classifier

After the agent finishes a path, a sentiment classifier is deployed to obtain the final sentiment prediction:

$$p = \delta(W_p S_t + b_p). \tag{5}$$

Eq. (5) is a single layer network, where $\delta(\cdot)$ is the non-linear activation. In our implementation, we deploy Softmax activation to predict one polarity from the candidate pool. W_p is the weight and b_p is the bias.

G. Reward

The reward provides the information which directly guides the training procedure. Conventional RL frameworks have a clear reward design for each step. However, in our framework, the ultimate goal is high classification performance which relays on multiple modules. To this end, we directly consider the correct prediction as the reward for optimizing all the modules. We deploy Mean Squared Error (MSE) as the accuracy reward and the function is shown below:

$$r_{acc} = -\|y - p\|_2^{\mathrm{F}}, \tag{6}$$

where r_{acc} is the accuracy reward, y is the ground truth label. r_{acc} would only be calculated once after the walk is finished. It is a straightforward yet effective reward. In the training procedure, all the three modules are simultaneously optimized to achieve the best performance. Other evaluation metrics (e.g., Cross Entropy) could also be deployed. We empirically evaluate different metrics, and we found most metrics are effective while MSE achieves the best performance.

H. Implementation & Optimization

We use a 2-layer fully-connected (FC) neural network to parameterize the policy network that maps the state vector S_t to a probability distribution over all possible actions. ReLU activation is deployed in the first layer while no activation in the second layer. However, to obtain a smooth score across different candidates, the sigmoid function is deployed across all actions. In the optimization procedure, some strategies are used to improve its efficiency and effectiveness. For the policy network, simply choosing the candidate with the highest score is a non-differentiable sampling strategy making it infeasible to calculate the weight gradients. Gumbel Softmax [26] employs a continuous distribution to approximate a non-differentiable sample. In this way, we can have a one-hot vector as the output of the agent while still preserving the parameter gradients. The core functions of Gumbel Softmax is shown as follow:

$$z_i = \frac{exp((log(\pi_i) + g_i)/\tau)}{\Sigma_{j=1}^{k} exp((log(\pi_j) + g_j)/\tau)}, \tag{7}$$

where g_i are $i.i.d$ samples drawn from $Gumbel(0, 1)$ distribution, π_i are the class probabilities, z_i are the generated samples, τ is the temperature parameter. The smaller the τ, the final result is more close to a one-hot vector. The output would be a one-hot vector, while the gradients are calculated from the Gumbel Softmax. For the testing process, since we do not need to calculate the gradients and add diversity, we simply use $argmax(\cdot)$ function to get the one-hot vector.

IV. EXPERIMENT

A. Datasets

We utilized 5 public datasets. Table I illustrates their statistical information. Specifically, **LAP14** [27] includes reviews of laptops from users. Aspects are the characteristics of computers (e.g., screen, speed). **TWITTER** [1] contains mentions from public available tweets. The aspects are diverse including celebrities (e.g., *Bill Gates*)), and companies (e.g., *Google*). **REST14** [27] focuses on restaurants' reviews. The aspects include the flavor, style, service, and the environment. The annotations for aspect terms occurring in the sentences. **REST15** [28] has similar aspects compared with [27], while it includes reviews for laptops, restaurants, and hotels. **REST16** [29] contains more comprehensive reviews on restaurants, which are used to help companies to measure satisfaction and improve their products.

B. Baselines

We consider the following state-of-the-art approaches as baselines. **SVM** [5] is a conventional classification method, and traditional feature extraction pipeline and an in-house sequence tagger is used to detect aspect terms. **LSTM** [6] proposes an extension on general LSTM framework. It incorporates target information to fully explore the relevant context words. **MemNet** [7] uses external memories and a multi-hop architecture to learn the importance of context words, where input texts are modeled as naturally-ordered word sequences. **AOA** [9] introduces an attention-over-attention (AOA) strategy to the sentiment classification. It takes both text-to-aspect and aspect-to-text projections into consideration. **IAN** [8] applies an attention network to both aspects and context to learn the sentiment representations. **TNet-LF** [10] introduces a Target-Specific Transformation (TST) component to get the context information of the aspect text. Then it proposes Context-Preserving Transformation (CPT) layers to learn the abstract features. **ASGCN** [2] considers the contextual information by

TABLE II
SENTIMENT CLASSIFICATION PERFORMANCE OVER PUBLIC BENCHMARK DATASETS

Methods	Lap14		Rest14		Rest15		Rest16		Twitter	
	ACC	F1	ACC	F1	ACC	F1	ACC	F1	ACC	F1
SVM [5]	70.49	N/A	80.16	N/A	N/A	N/A	N/A	N/A	63.40	63.30
LSTM [6]	69.28	63.09	78.13	67.47	77.31	55.17	86.80	63.88	69.56	67.70
MemNet [7]	70.64	65.17	79.61	69.64	77.31	58.28	85.44	65.99	71.48	69.90
AOA [9]	72.62	67.52	79.97	70.42	78.17	57.02	87.50	66.21	72.30	70.20
IAN [8]	72.05	67.38	79.26	70.09	78.54	52.65	84.74	55.21	72.50	70.81
TNet-LF [10]	74.61	70.14	80.42	71.03	78.47	59.47	**89.07**	70.43	72.98	71.43
ASCNN [2]	72.62	66.72	81.73	73.10	78.47	58.90	87.39	64.56	71.05	69.45
ASGCN-DT [2]	74.14	69.24	80.86	72.19	79.34	60.78	88.69	66.64	71.53	69.68
ASGCN-DG [2]	75.55	71.05	80.77	72.02	79.89	61.89	88.99	67.48	72.15	70.40
SentRL	**75.55**	**71.48**	**82.05**	**73.40**	**80.07**	**65.54**	87.82	**71.17**	**73.12**	**71.64**

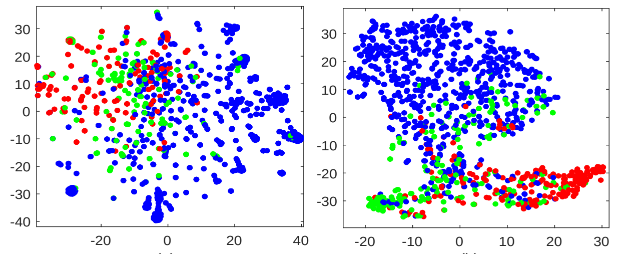

Fig. 3. t-SNE visualization of the learned sentimental representations, where (a) is ASGCN method and (b) is our approach. Different color denotes different sentimental categories including *positive*, *neutral*, and *negative*.

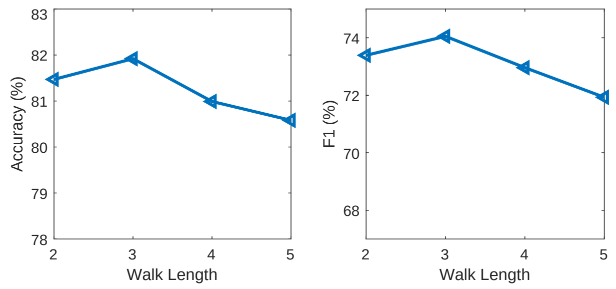

Fig. 4. Performance with different walk length in the dependent graph.

utilizing dependency graphs, and a GCN and an attention mechanism are used to learn sentimental representations.

In the experiments, we follow the commonly adopted experimental setting described in [7]. GloVe word embedding [30] is used to extract the initial representations which is further fine-tuned by a bidirectional LSTM [2]. ADAM optimizer is used to train SentRL with a learning rate of 0.0005. We employ the same evaluation metrics discussed in [2], where accuracy (ACC) and macro-F1 (F1) are adopted. Dependency graphs are obtained from spcCy[1].

C. Performance Analysis

The overview performance is reported in Table II. We observe that SentRL outperforms all the baseline methods in 4 out of 5 cases in terms of accuracy (ACC). In terms of macro-F1, SentRL outperforms in all the cases, with up to 3.7% improvement over the best baseline method.

We visualize the learned sentimental representations and compare it with the ones generated from the second best

[1]https://spacy.io/

TABLE III
SENTIMENT CLASSIFICATION PERFORMANCE BASED ON RANDOM WALK

Walk Length	Lap14		Rest14		Rest15		Rest16		Twitter	
	ACC	F1	ACC	F1	ACC	F1	ACC	F1	ACC	F1
1	65.88	60.05	74.37	61.51	74.11	51.59	84.30	61.13	67.34	63.86
2	67.47	63.28	75.44	64.82	74.76	53.15	84.44	63.29	67.34	65.17
3	70.42	65.10	76.03	65.89	74.91	53.85	84.98	64.12	67.93	66.12
4	70.51	65.43	76.15	66.28	75.61	54.04	85.15	65.00	68.14	66.82
Ours	**75.55**	**71.48**	**82.05**	**73.40**	**80.07**	**65.54**	**87.82**	**71.17**	**73.12**	**71.64**

model, ASGCN [2] by t-SNE [31]. As shown in Figure 3, red, green, and blue dots denote *negative*, *neutral*, and *positive* sentiments, respectively. We observe that our approach could better separate different sentiments.

Walk length is another crucial parameter. We evaluated the model with different walk length (i.e., from 2 to 5) and the result is shown in Figure 4. Our model achieves the best performance when walk length is 3. We assume that if walk length is less than 3, the agent cannot reach to the sentiment word and the performance decreases significantly. Meanwhile, if walk length is more than 3, the performance only has slight decline, and we conjecture this is due to the unnecessary walks which aggregate extra information.

To demonstrate the effectiveness of the RL module, we let the agent randomly walk through the dependency graph while keeping the other modules unchanged. The result is illustrated in Table III. We observed that when walk length is 1, the performance significantly drops. However, it is still higher than random guess. When the walk length increases, the performance cannot increase consistently. From the results, we could conclude that 1) our RL approach is effective and essential, and 2) the aspect node does have biases for different sentiment polarity, and walking through the dependency graph could more effectively capture the syntactical knowledge.

We present a case study and visualize traces from aspect words to their related sentimental words. Three correct predictions and one incorrect prediction are shown in Figure 5, where words in blue boxes are aspects, grey edges denote syntactic dependencies between words, and red edges are traces left by a learned agent. For example, in case 1, the agent starts from the aspect word *food* and ends at the word *fresh*, collecting necessary information within three hops as expected. If the policy network believes it is the final word and no extra moves are required, then it keeps walking to the same word until the

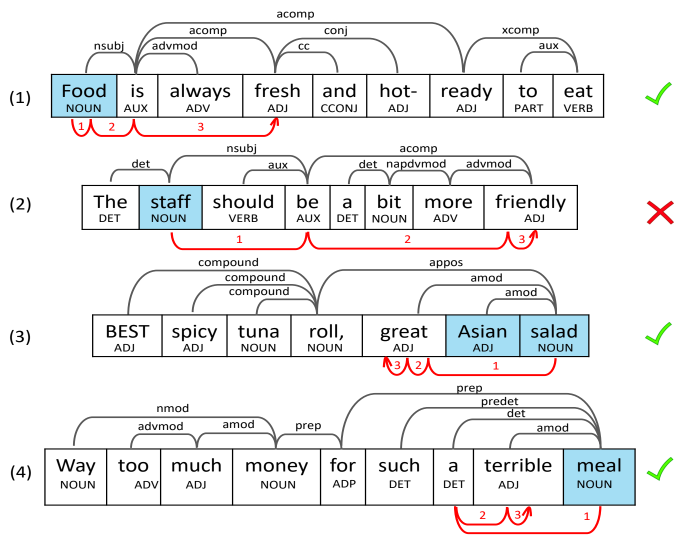

Fig. 5. A case study demonstrating traces from aspect words to their related sentimental words. For example, in case 1, the agent trace is *Food*→*Food*→*is*→*fresh*. Grey edges denote extracted syntactic dependencies, and red ones denote traces generated by a learned agent. Cases 1,3,4 are samples correctly predicted by SentRL, while case 2 is an incorrect prediction.

walk stops. Case 3 and Case 4 illustrate this operation of the agent. In Case 3, when agent arrived to *great* and recognized the sufficiency of the walk and no extra walks are required, then it keeps walking to the same token. In Case 4, the agent walks again to the word, *terrible*, in the last walk.

V. Conclusion

In this paper, we propose SentRL, a novel aspect-based sentiment classification approach via reinforcement learning. By investigating input texts from their syntactic structures (e.g., dependency graphs), we are able to effectively reduce input variance introduced by diverse expressions in natural languages. On top of the syntactic structures, an agent is deployed to discover the most effective paths that link aspects with their sentimental descriptions. By collecting evidences from the discovered paths, a semantic understanding module in SentRL learns to make the accurate sentimental classification. All the modules in SentRL are simultaneously trained in an end-to-end fashion. Experimental results illustrate the high performance of our approach and extensive case studies demonstrate the effectiveness and efficiency of our model.

References

[1] L. Dong, F. Wei, C. Tan, D. Tang, M. Zhou, and K. Xu, “Adaptive recursive neural network for target-dependent twitter sentiment classification,” in *ACL*, 2014, pp. 49–54.

[2] C. Zhang, Q. Li, and D. Song, “Aspect-based sentiment classification with aspect-specific graph convolutional networks,” in *EMNLP*, 2019, pp. 4567–4577.

[3] M. A. Covington, “A fundamental algorithm for dependency parsing,” in *ACM Southeast Conference*, 2001, pp. 95–102.

[4] T. T. Thet, J.-C. Na, and C. S. Khoo, “Aspect-based sentiment analysis of movie reviews on discussion boards,” *Journal of Information Science*, vol. 36, no. 6, pp. 823–848, 2010.

[5] S. Kiritchenko, X. Zhu, C. Cherry, and S. Mohammad, “NRC-Canada-2014: Detecting aspects and sentiment in customer reviews,” in *ACL*, 2014, pp. 437–442.

[6] D. Tang, B. Qin, X. Feng, and T. Liu, “Effective LSTMs for target-dependent sentiment classification,” in *International Conference on Computational Linguistics*, 2016, pp. 3298–3307.

[7] D. Tang, B. Qin, and T. Liu, “Aspect level sentiment classification with deep memory network,” in *EMNLP*, 2016, pp. 214–224.

[8] D. Ma, S. Li, X. Zhang, and H. Wang, “Interactive attention networks for aspect-level sentiment classification,” in *IJCAI*, 2017.

[9] B. Huang, Y. Ou, and K. M. Carley, “Aspect level sentiment classification with attention-over-attention neural networks,” in *International Conference on Social Computing, Behavioral-Cultural Modeling and Prediction and Behavior Representation in Modeling and Simulation*. Springer, 2018, pp. 197–206.

[10] X. Li, L. Bing, W. Lam, and B. Shi, “Transformation networks for target-oriented sentiment classification,” in *ACL*, 2018.

[11] G. Mittal, C. Liu, N. Karianakis, V. Fragoso, M. Chen, and Y. Fu, “HyperSTAR: Task-aware hyperparameters for deep networks,” in *CVPR*, 2020, pp. 8736–8745.

[12] L. Wang, B. Zong, Q. Ma, W. Cheng, J. Ni, W. Yu, Y. Liu, D. Song, H. Chen, and Y. Fu, “Inductive and unsupervised representation learning on graph structured objects,” in *ICLR*, 2020.

[13] Q. Ma and A. Olshevsky, “Adversarial crowdsourcing through robust rank-one matrix completion,” in *NeurIPS*, 2020.

[14] H. Tang, D. Ji, C. Li, and Q. Zhou, “Dependency graph enhanced dual-transformer structure for aspect-based sentiment classification,” in *ACL*, 2020, pp. 6578–6588.

[15] M. H. Phan and P. O. Ogunbona, “Modelling context and syntactical features for aspect-based sentiment analysis,” in *ACL*, 2020, pp. 3211–3220.

[16] L. Xu, L. Bing, W. Lu, and F. Huang, “Aspect sentiment classification with aspect-specific opinion spans,” in *EMNLP*. Online: Association for Computational Linguistics, Nov. 2020, pp. 3561–3567.

[17] R. S. Sutton, D. A. McAllester, S. P. Singh, and Y. Mansour, “Policy gradient methods for reinforcement learning with function approximation,” in *NeurIPS*, 2000, pp. 1057–1063.

[18] K. Narasimhan, T. Kulkarni, and R. Barzilay, “Language understanding for text-based games using deep reinforcement learning,” in *EMNLP*, 2015, pp. 1–11.

[19] W. Y. Wang, J. Li, and X. He, “Deep reinforcement learning for NLP,” in *ACL*, 2018, pp. 19–21.

[20] Q. Ma, Y.-Y. Liu, and A. Olshevsky, “Optimal lockdown for pandemic control,” *arXiv:2010.12923*, 2020.

[21] L. Wang, Y. Liu, C. Qin, G. Sun, and Y. Fu, “Dual relation semi-supervised multi-label learning,” in *AAAI*, 2020.

[22] L. Wang, Z. Ding, S. Han, J.-J. Han, C. Choi, and Y. Fu, “Generative correlation discovery network for multi-label learning,” in *IEEE ICDM*, 2019, pp. 588–597.

[23] W. Xiong, T. Hoang, and W. Y. Wang, “DeepPath: A reinforcement learning method for knowledge graph reasoning,” in *EMNLP*, 2017, pp. 564–573.

[24] J. Wang, C. Sun, S. Li, J. Wang, L. Si, M. Zhang, X. Liu, and G. Zhou, “Human-like decision making: Document-level aspect sentiment classification via hierarchical reinforcement learning,” in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2019.

[25] J. Johnson, B. Hariharan, L. Van Der Maaten, J. Hoffman, L. Fei-Fei, C. Lawrence Zitnick, and R. Girshick, “Inferring and executing programs for visual reasoning,” in *ICCV*, 2017, pp. 2989–2998.

[26] E. Jang, S. Gu, and B. Poole, “Categorical reparametrization with Gumble-Softmax,” in *ICLR*, 2017.

[27] M. Pontiki, D. Galanis, J. Pavlopoulos, H. Papageorgiou, I. Androutsopoulos, and S. Manandhar, “SemEval-2014 task 4: Aspect based sentiment analysis,” in *ACL Workshop on Semantic Evaluation*, 2014, pp. 27–35.

[28] M. Pontiki, D. Galanis, H. Papageorgiou, S. Manandhar, and I. Androutsopoulos, “Semeval-2015 task 12: Aspect based sentiment analysis,” in *ACL Workshop on Semantic Evaluation*, 2015, pp. 486–495.

[29] M. Pontiki, D. Galanis, H. Papageorgiou, I. Androutsopoulos, S. Manandhar, M. Al-Smadi, M. Al-Ayyoub, Y. Zhao, B. Qin, O. De Clercq *et al.*, “Semeval-2016 task 5: Aspect based sentiment analysis,” in *ACL Workshop on Semantic Evaluation*, 2016.

[30] J. Pennington, R. Socher, and C. D. Manning, “Glove: Global vectors for word representation,” in *EMNLP*, 2014, pp. 1532–1543.

[31] L. v. d. Maaten and G. Hinton, “Visualizing data using t-SNE,” *Journal of Machine Learning Research*, pp. 2579–2605, 2008.