Neural Subgraph Explorer: Reducing Noisy Information via Target-Oriented Syntax Graph Pruning

Bowen Xing¹ and Ivor Tsang²,¹

¹Australian Artificial Intelligence Institute, University of Technology Sydney, Australia
²Centre for Frontier AI Research, A*STAR, Singapore
bxing714@gmail.com, ivor_tsang@ihpc.a-star.edu.sg

Abstract

Recent years have witnessed the emerging success of leveraging syntax graphs for the target sentiment classification task. However, we discover that existing syntax-based models suffer from two issues: noisy information aggregation and loss of distant correlations. In this paper, we propose a novel model termed Neural Subgraph Explorer, which (1) reduces the noisy information via pruning target-irrelevant nodes on the syntax graph; (2) introduces beneficial first-order connections between the target and its related words into the obtained graph. Specifically, we design a multi-hop actions score estimator to evaluate the value of each word regarding the specific target. The discrete action sequence is sampled through Gumble-Softmax and then used for both of the syntax graph and the self-attention graph. To introduce the first-order connections between the target and its relevant words, the two pruned graphs are merged. Finally, graph convolution is conducted on the obtained unified graph to update the hidden states. And this process is stacked with multiple layers. To our knowledge, this is the first attempt of target-oriented syntax graph pruning in this task. Experimental results demonstrate the superiority of our model, which achieves new state-of-the-art performance.

1 Introduction

Target (or aspect) sentiment classification (TSC) [Tang et al., 2016] aims to infer the sentiment polarity of the specific target included in a review context. For example, in the review “The price is reasonable but the service is poor.”, there are two targets ‘price’ and ‘service’ with opposite sentiment polarities. Generally, TSC task is formulated as a classification task whose input is a given context-target pair.

Earlier dominant neural TSC models are based on attention mechanisms [Ma et al., 2017; Chen et al., 2017], which are designed to capture the correlations between the target and its relevant context words. Although promising progress has been achieved, researchers discovered that attention mechanisms may mistakenly attend to the target’s syntactically unrelated words and have difficulty capturing the distant while crucial context words [Zhang et al., 2019; Huang and Carley, 2019]. To this end, the syntax graph of the context is widely leveraged to incorporate the syntactic information via applying graph neural networks (GNNs), e.g. graph convolutional network (GCN) and graph attention network (GAT). With the combination of the BERT [Devlin et al., 2019] which has proven its power in heterogeneous tasks, the recent models of the BERT+Syntax paradigm has achieved state-of-the-art performances [Tang et al., 2020; Li et al., 2021; Xing and Tsang, 2022a].

However, despite the remarkable improvements brought by the incorporation of syntactic information, we find existing models suffer from two issues which are illustrated in Fig. 1:

- **Noisy information aggregation.** In existing models, the message passing process of GNNs is conducted on the whole syntax graph, and all words transfer information with their neighbors. Consequently, the noisy information contained in target-irrelevant works may be aggregated into the target nodes, which disturbs the prediction. As shown in example (1), ‘less expensive’ are crucial for inferring the positive sentiment of ‘USB3 Peripherals’ and their distance on the syntax graph is 2. However, the distance between ‘less expensive’ and another target ‘ThunderBolt ones’ is also 2, while this review expresses negative sentiment on ‘ThunderBolt ones’. In this case, the information of ‘less expensive’ is aggregated into both targets. For the first target, it is beneficial, while for the later one, its information is noisy, harming the prediction.

- **Loss of distant correlations.** After l-layer GNNs, the information of a node’s ≤ l/th-order neighbors can be aggregated into it, while further nodes cannot exchange information with it. Sometimes, the critical words may

![Figure 1: Illustration of two examples with syntax dependencies. Example (1) is from Laptop14 dataset while example (2) is from Restaurant14 dataset. Targets are underlined. Red color denotes the sentiment of the target is positive, while color denotes negative.](image-url)
be distant from the target on the syntax graph. Thus the target loses their crucial information. As shown in example (2), ‘excellent’ plays the key role in expressing the positive sentiment of ‘dishes’, while their distance is 4, which means that their correlation cannot be captured by the widely-adopted 2- or 3-layer GNNs [Zhang et al., 2019; Tang et al., 2020; Li et al., 2021].

To solve the first issue, target-irrelevant nodes are supposed to be removed from the syntax graph. For the second issue, an alternative solution is to increase the layer number of GNNs. However, this would lead to over-fitting problem and exacerbates the first issue. Another alternative is leveraging the fully-connected self-attention graph [Tang et al., 2020; Li et al., 2021] to introduce first-order connections. In this case, massive correlative information between the words is introduced. Although the beneficial first-order connections between the target and its related words are introduced, the noisy ones between the target and noisy words are also integrated, which exacerbates the first issue. Thus we should introduce the beneficial first-order connections and eliminate the noisy first-order connections simultaneously.

In this paper, we argue that it is urgent to conduct target-oriented syntax pruning. On the one hand, it can reduce the noisy information via pruning the target-irrelevant nodes. On the other hand, pruning the syntax graph and self-attention graph then merging them can adaptively introduce the beneficial first-order connections between the target and its related words rather than all first-order correlations, which include the noisy ones. To this end, we propose Neural Subgraph Explorer, whose core is a stacked target-oriented syntax graph pruning layer. We design a multi-hop action score estimator to evaluate the contribution of each node regarding the target. And we leverage the Gumble-Softmax [Jang et al., 2017] for stable and differentiable discrete action sampling. To obtain the first-order connections between the target and its related words, we apply the non-local self-attention to the pruned syntax graph has isolated nodes which harms the message passing. Experimental results on benchmark datasets show that our model achieves new state-of-the-art performance, significantly surpassing existing models.

2 Methodology
The architecture of our model is illustrated in Fig. 2. We propose the position weighted GCN for graph message passing. And the core of our model is the proposed target-oriented syntax-graph pruning layer. Next, we introduce the details.

2.1 Contextual and Syntactic Encoding

BERT Encoder. Following the up-to-date models [Wang et al., 2020; Tian et al., 2021a], we employ BERT encoder to obtain the initial word hidden states. Given the review context word sequence $x_1, x_2, \ldots, x_{N_c}$ and the target word sequence $a_1, \ldots, a_{N_t}$, the input of BERT is the target-context pair:

$$\langle [CLS]; x_1, x_2, \ldots, x_{N_c}; [SEP]; a_1, \ldots, a_{N_t}; [SEP] \rangle$$

where $N_c$ and $N_t$ are the length of review context and target respectively, and $\langle \cdot \rangle$ denotes sequence concatenation. Then we obtain the hidden state sequence of the review:

$$H_c = [h^c_1, \ldots, h^c_{N_c}] \in \mathbb{R}^{N_c \times d},$$

including the target word hidden states $H_t = [h^t_1, \ldots, h^t_{N_t}] \in \mathbb{R}^{N_t \times d}$, where $d$ denotes the dimension of the hidden state.

Position Weighted Graph Convolutional Network. Graph convolutional network (GCN) has been widely used to encode the syntax graph to integrate syntactic information into hidden states. In this work, based on standard GCN, we introduce a weight for each word to indicate its position relative to the target, forming the position-weighted GCN. By this means, the potential relative words for the target can be highlighted. Specifically, the node updating process can be formulated as:

$$h^t_i = \text{ReLU} \left( \sum_{j=0}^{N} A^{syn}_{ij}(w^0_jW^1_{ij}^{-1}) + b^0 \right)$$

where $A^{syn}$ is the adjacent matrix derived from the dependency parsing result and $A^0_{ij} = 1$ if there is a dependency from node $j$ to $i$; $\mu_j$ is the relative offset between $j$-th word and the target and $w^0_j$ is the position weight of $j$-th word; $d_i$ denotes the degree of $i$-th node; $W^1_{ij}$ and $b^0$ are parameters.
If the target is a phrase, $t - \tau$ is calculated with its left or right boundary index according to which side the word locates.

Now we obtain the syntax-enhanced context hidden states $H_t$ and target hidden states $H_\tau$. And by applying mean pooling over $H_t$, we obtain the initial target representation $r_t$.

2.2 Target-oriented Syntax Graph Pruning

The process of target-oriented syntax graph pruning includes four steps: (1) the multi-hop action score estimator evaluates the value of each word regarding the target, producing the degree that whether the word should be pruned or reserved; (2) the Gumble-Softmax is leveraged for differentiable discrete action sampling and generate the action sequence; (3) the action sequence is used to mask the adjacent matrix of the syntax graph and self-attention graph; (4) the position weighted GCN is adopted for message passing on the obtained graph, updating the hidden states. Next we present the details of each module following the above order.

Multi-hop Action Score Estimator

To decide whether a node on the graph should be pruned, each word is supposed to be assigned a score to represent its contribution for expressing the sentiment semantics of the given target. To this end, inspired from [Chen et al., 2017], we design a multi-hop action score estimator. At each hop, it produces a gate score for each word and a summarized target-centric context vector which is used at next hop. The details are given bellow:

$$s^t_j = W_s [w^t_j h^t_j, \mu_j, C_{t-1}, r_t] + b_s$$

$$g^t_j = \text{Sigmoid}(s^t_j)$$

$$\alpha^t_j = \frac{\exp(s^t_j)}{\sum^N_k \exp(s^t_k)}$$

$$t^t = \sum^N_j \alpha^t_j w^t_j h^t_j$$

$$C_t = \text{GRU}(t^t, C_{t-1})$$

where $W_s$ and $b_s$ are parameters; GRU denotes gated recurrent unit. $C_0$ is initialized as a zero vector.

Then the output gate $g^{T^t}_j$ of final step $T^t$ can derive

$$p^{0}_j = 1 - g^{T^t}_j, p^{1}_j = g^{T^t}_j$$

which represent the possibilities that node $j$ should be pruned or reserved, respectively.

Action Sampling

Now we have a problem of discrete action selecting. Although REINFORCE algorithm [Williams, 1992] are commonly used for this problem, it causes model instability and hard training. To this end, we leverage the Gumble-Softmax [Jang et al., 2017] trick for differentiable action sampling:

$$\text{act}_j = \frac{\exp((\log(p^{0}_j, a) + \epsilon_1) / \pi)}{\sum^{N_a} \exp((\log(p^{0}_j, a) + \epsilon_1) / \pi)}$$

where $\epsilon_1$ is randomly sampled from Gumble distribution and $\pi$ is the temperature coefficient which is set 0.1 in this work. $\text{act}_j = 0$ denotes pruning while $\text{act}_j = 1$ denotes reserving.

Graph Pruning and Merging

Given the set of actions $\{act^t_j\}_{i=1}^{N_t}$ corresponding to the context word, we use it to mask $A^{syn}$:

$$A^{syn,t}_j = act^t_j \cdot A^{syn}_j$$   (6)

where $A^{syn,t}_j$ is the adjacent matrix of the pruned syntax graph of the $t$-th layer.

Now in the pruned syntax graph, some target-irrelevant nodes (words) are removed. However, the issue of loss of distant correlations is not tackled. Besides, the pruning operation on the whole syntax graph may lead to a potential problem that the pruned syntax graph may include isolated nodes, which hinders the message passing on the obtained graph.

To solve the above two issues, we propose to exploit the self-attention graph (SatGraph), which is a fully-connected semantic graph consisting of all words in the sentence and derived by the self-attention [Vaswani et al., 2017]. SatGraph can provide the first-order connections between each two nodes. Therefore, merging the pruned SatGraph with the pruned SynGraph can not only guarantee the connectivity but also can directly connect the target with its related words, promoting the aggregation of target-related information, which is beneficial for prediction.

An intuitive way to obtain the self-attention graph is to retrieve the self-attention matrix at the last layer of BERT [Devlin et al., 2019]. However, considering the self-attentions in BERT\(_{BASE}\) (BERT\(_{LARGE}\)) are 12-head (16-head), and the word representations are segmented into 12 (16) local subspaces, one of the 12 (16) self-attention matrices can only reflect the local word correlations in the local subspace rather than general global semantic space in which the subsequent modules work. Therefore, due to the segmentation bias between the multi-head local subspaces and the general \emph{global} semantic space, the self-attention matrix derived by BERT cannot be used for the self-attention graph.

To this end, we add a non-local self-attention layer between BERT and position weighted GCN, as shown in Fig. 2, to obtain the \emph{global} self-attention matrix $A^{sat} \in \mathbb{R}^{N_t \times N_t}$ representing the correlations between words in the general semantics space which is consistent with subsequent modules. Specifically, $A^{sat}$ is obtained as follows:

$$A^{sat} = \text{Softmax} \left( \begin{pmatrix} \tilde{H}_c & M_q \end{pmatrix} \begin{pmatrix} \tilde{H}_c & M_v \end{pmatrix}^T / \sqrt{d} \right)$$

Then the updated hidden states are obtained by:

$$\tilde{H}_c = A^{sat} \tilde{H}_c M_v$$

where $M_q, M_v \in \mathbb{R}^{d \times d}$ are parameters. And then $\tilde{H}_c$ is fed to the position weighted GCN before the target-oriented syntax graph pruning layer.

Then we prune SatGraph in the same way as SynGraph and merge them:

$$A^{sat,t}_j = act^t_j \cdot A^{sat}_j$$

$$A^t = (A^{syn,t} + A^{sat,t}) / 2$$   (9)

In this process, not only the beneficial first-order connections are introduced, but also the noisy first-order connections are removed due to pruning operation on SatGraph. Then $A^t$ will be used for messaging passing later.
Node Updating
For message passing on the obtained graph, we apply another position weighted GCN (noted as (b) in Fig. 2) over $A^t$.

Multi-layer Stacking
In order to let our model gradually improve the graph pruning and learn deep features, we stack the target-oriented syntax graph pruning module in a multi-layer manner.

2.3 Prediction and Training
After $T$ layers of target-oriented syntax graph pruning, we obtain the final representations of each context word, which includes $N_t$ target words. We apply mean pooling over all target words to generate the final target representation, getting the final target representation $R_t$.

Then we fed $R_t$ to a multi-layer perception (MLP) and then softmax for classification:

$$P = \text{softmax}(W^2(W^1R_a + b^2) + b^1)$$

Finally, by applying arg max on the class vector $P$, we can get the produced sentiment polarity of the target in the review. Given $D$ training samples, the training objective is:

$$\ell = - \sum_{i=1}^{D} \sum_{c \in \mathcal{C}} I(y = c) \log (P(y = c))$$

where $y$ is the ground-truth class, $I$ is an indicator function, and $\mathcal{C}$ denotes the sentiment polarity class set.

3 Experiment
3.1 Experimental Setup

Dataset
We conduct experiments on three public benchmark datasets to obtain reliable and authoritative results. Restaurant14 and Laptop14 are from [Pontiki et al., 2014], and Restaurant15 is from [Pontiki et al., 2015]. We pre-process the datasets following the same way as previous works [Zhang et al., 2019; Wang et al., 2020; Tian et al., 2021a]. The statistics of all datasets are shown in Table 1.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Positive</th>
<th>Neutral</th>
<th>Negative</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Train</td>
<td>Test</td>
<td>Train</td>
</tr>
<tr>
<td>Laptop14</td>
<td>994</td>
<td>341</td>
<td>464</td>
</tr>
<tr>
<td>Restaurant14</td>
<td>2164</td>
<td>728</td>
<td>637</td>
</tr>
<tr>
<td>Restaurant15</td>
<td>912</td>
<td>326</td>
<td>36</td>
</tr>
</tbody>
</table>

Table 1: Dataset statistics of the three datasets.

Implementation Details
We adopt the BERT$_\text{base}$ uncased version as the BERT encoder and it is fine-tuned in the experiments. We train our Neural Subgraph Hunter using AdamW optimizer. The dependency parser used in our experiments is from spaCy toolkit. In our experiments, the dimension of hidden units are 768. The dropout rate for BERT encoder is 0.1, while the dropout rate for other modules is 0.3. The batch size is 16 and epoch number is 30. The learning rates are 1e-5, 5e-5, 3e-5 for Lap14, Res14 and Res15 datasets respectively. The weight decay rages are 0.05 for Res14 and Res15 datasets while 0.001 for Lap14 datasets. The layer number of the position weighted GCN (a) and (b) are both 2. The hop number of the multi-hop action score estimator is 3. The layer number of target-oriented syntax graph pruning is 2. And our source code is available at https://github.com/XingBowen714/Neural-Subgraph-Explorer.

Accuracy (Acc) and Macro-F1 (F1) are used as evaluation metrics. Since there is no official validation set for the datasets, we report the average results over three random runs.

Baseline for Comparison
The baselines can be divided into four categories regarding whether BERT and syntax are leveraged:

(A) BERT × Syntax ×: 1. IAN [Ma et al., 2017] separately encodes the target and context, then models their interactions through an interactive attention mechanism. 2. RAM [Chen et al., 2017] uses a GRU attention mechanism to recurrently extracts the target-related semantics.

(B) BERT × Syntax ✓: 3. ASGCN [Zhang et al., 2019] utilizes GCN to leverage syntactic information. 4. BiGCN [Zhang and Qian, 2020] employs GCN to convolute over hierarchical syntactic and lexical graphs.

(C) BERT ✓ Syntax ×: 5. BERT-SPC [Devlin et al., 2019] takes the concatenated context-target pair as input and uses the output hidden state of [CLS] token for classification. 6. AEN-BERT [Song et al., 2019] employs multiple attention layers to learn target-context interactions.

(D) BERT ✓ Syntax ✓: 7. ASGCN+BERT [Zhang et al., 2019]. Since the backbone of our Neural Subgraph Explorer is BERT+GCN, we augment the ASGCN model with BERT encoder to form a baseline. 8. KG CapsAN-BERT [Zhang et al., 2020] utilizes multi-prior knowledge to guide the capsule attention process and use a GCN-based syntactic layer to integrate the syntactic knowledge. 9. R-GAT+BERT [Wang et al., 2020] uses the relational graph attention network to aggregate the global relational information from all context words into the target node representation. 10. DGEDT-BERT [Tang et al., 2020] employs a dual-transformer network to model the interactions between the flat textual knowledge and dependency graph empowered knowledge. 11. A-KVMN+BERT [Tian et al., 2021b] uses a key-value memory network to leverage not only word-word relations but also their dependency types. 12. BERT+T-GCN [Tian et al., 2021a] leverages the dependency types in T-GCN and uses an attentive layer ensemble to learn the comprehensive representation from different T-GCN layers. 13. DualGCN+BERT [Li et al., 2021] uses orthogonal and differential regularizers to model the interactions between semantics and syntax.

Note that all of the BERT encoders in baselines are BERT-base uncased version, the same as ours. And for fair comparison, we reproduce the average results of R-GAT+BERT, A-KVMN+BERT, BERT+T-GCN and DualGCN+BERT on three random runs because they report the best results rather than average results in their original paper.
### 3.2 Main Results

The performances of our Neural Subgraph Explorer and baselines are shown in Table 2. We can observe that BERT can significantly boost the performance while integrating syntactic information into BERT-based models can bring further significant improvement. And the best-performing models are all based on BERT+Syntax paradigm.

However, state-of-the-art models neglect the problem that not all words are useful for expressing the sentiment of the specific target, and they just adopt GNNs to conduct message passing on the whole syntax graph. As a result, the noisy information from the target-irrelevant words is aggregated into the target’s and its related words’ node representation, which affects the prediction of the specific target’s sentiment. To overcome this problem, we propose a Neural Subgraph Explorer model which can adaptively and dynamically prune the noisy nodes corresponding to the target-irrelevant words. As shown in Table 2, our model achieves new state-of-the-art performance, obtaining consistent improvements over all baselines in terms of both Acc and F1. Specifically, our model surpasses previous best scores by 1.30%, 0.71%, and 1.16% in terms of Acc on Lap14, Res14, and Res15 datasets, respectively. And in terms of F1, our model surpasses previous best scores by 1.16%, 1.28%, and 3.44% on Lap14, Res14, and Res15 datasets, respectively. All the improvements are attributed to the discarding of noisy information via pruning noisy nodes on the vanilla syntax graph and introducing first-order dependency to facilitate the aggregation of distant while crucial target-related information.

### 3.3 Ablation Study

We conduct ablation experiments to look into Neural Subgraph Explorer and investigate how it works well. The experiment results are shown in Table 3.

<table>
<thead>
<tr>
<th>Variants</th>
<th>Laptop14</th>
<th>Restaurant14</th>
<th>Restaurant15</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Acc</td>
<td>F1</td>
<td>Acc</td>
</tr>
<tr>
<td>Full Model</td>
<td>82.13</td>
<td>78.51</td>
<td>87.35</td>
</tr>
<tr>
<td>NoPrune</td>
<td>79.52</td>
<td>75.70</td>
<td>86.04</td>
</tr>
<tr>
<td></td>
<td>↓2.61</td>
<td>↓2.81</td>
<td>↓1.31</td>
</tr>
<tr>
<td>RandPrune</td>
<td>80.67</td>
<td>76.97</td>
<td>86.49</td>
</tr>
<tr>
<td></td>
<td>↓1.46</td>
<td>↓1.54</td>
<td>↓0.86</td>
</tr>
<tr>
<td>NoMerge</td>
<td>81.61</td>
<td>78.25</td>
<td>87.05</td>
</tr>
<tr>
<td></td>
<td>↓0.52</td>
<td>↓0.26</td>
<td>↓0.30</td>
</tr>
</tbody>
</table>

We first study the effect of target-oriented syntax graph pruning via removing the pruning operation. In practice, we achieve this by setting all \( \alpha^t \) as 1 and this variant is termed NoPrune. From Table 3 we can observe that NoPrune obtains much poorer results compared with the full model. The reason is that without noisy node pruning, the main advantage of Neural Subgraph Explorer is lost and the noisy information conveyed in the target-irrelevant nodes harms the prediction.

Then we study the effect of multi-hop action score estimator, which determines whether a word should be pruned. We design a variant named RandPrune, which assigns a random value to each \( d_1^{i,T'} \). Therefore, the nodes in the syntax graphs are randomly pruned. From Table 3 we can find that RandPrune is significantly inferior to the full model. This is because without the multi-hop action score estimator, RandPrune cannot identify the crucial words that contribute to the expression of the target’s sentiment, and the words it prunes may be the critical words and it may reserve the noisy words that harm the prediction.
For each position weighted GCN in our model, the layer number of target-oriented syntax graph pruning, we vary the value of $T$. To investigate the impact of layer number of target-oriented syntax graph pruning module. On the one hand, it introduces first-order connections between the target and the crucial words via merging the pruned self-attention graph with the pruned syntax graph. In this way, more useful information can be removed and more crucial information can be captured. Finally, experiments are conducted on benchmark datasets and the effectiveness of our model has been proven.

4 Related Works

In recent years, dominant TSC models are all based on neural networks which can automatically learn representations. And attention mechanisms [Wang et al., 2016; Ma et al., 2017; Chen et al., 2017; Tang et al., 2019; Xing et al., 2019] are widely utilized to capture the target-related words. [Ma et al., 2017] propose an interactive attention mechanism to model the mutual interactions between target and context. [Chen et al., 2017] propose a GRU-based attention mechanism to aggregate the target-centric semantics.

Recently, researchers discovered that only relying on attention mechanism is insufficient for TSC, especially for the cases in which the target is distant to its related words. To this end, a bunch of GNN-based models [Zhang et al., 2019; Tang et al., 2020; Huang et al., 2020; Tian et al., 2021a; Li et al., 2021; Xing and Tsang, 2021; Wang et al., 2020; Xing and Tsang, 2022a; Xing and Tsang, 2022b] are designed to encode the syntactic information for capturing the dependencies between the target and the crucial words. [Huang et al., 2020] propose a GAT-based model to leverage the syntactic information via applying GAT on the syntax graph, which is obtained from an off-the-shelf dependency parser. [Li et al., 2021] propose a dual-GCN architecture to model the semantic and syntactic information. Besides, the orthogonal regularizer and the differential regularizer are proposed to learn deeper correlations between the words. And with the power of BERT, the models based on BERT+Syntax paradigm have achieved state-of-the-art performances.

However, since existing models apply GNNs on the whole syntax graph, they suffer from two problems: noisy information aggregation and loss of distant correlations. In this paper, we propose a novel model termed Neural Subgraph Explorer to solve these two problems.

5 Conclusion

This paper proposes a novel Neural Subgraph Explorer model to tackle the target sentiment classification task. On the one hand, it can discard the noisy information contained in the noisy words regarding the given target through the stacked target-oriented syntax graph pruning module. On the other hand, it introduces first-order connections between the target and the crucial words via merging the pruned self-attention graph with the pruned syntax graph. In this way, more useless information can be removed and more crucial information can be captured. Finally, experiments are conducted on benchmark datasets and the effectiveness of our model has been proven.

Acknowledgements

This work was supported by Australian Research Council Grant (DP180100106 and DP200101328). Ivor W. Tsang was also supported by A∗STAR Centre for Frontier AI Research.
References


[Xing and Tsang, 2022b] Bowen Xing and Ivor W. Tsang. Understand me, if you refer to aspect knowledge: Knowledge-aware gated recurrent memory network. IEEE Transactions on Emerging Topics in Computational Intelligence, pages 1–11, 2022.


