Learn to Select via Hierarchical Gate Mechanism for Aspect-Based Sentiment Analysis

Xiangying Ran , **Yuanyuan Pan** , **Wei Sun** and **Chongjun Wang**
National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, China
Department of Computer Science and Technology, Nanjing University, Nanjing, China
{lebronran, yypan112}@gmail.com, weisun_@outlook.com, chwang@nju.edu.cn

Abstract

Aspect-based sentiment analysis (ABSA) is a fine-grained task. Recurrent Neural Network (RNN) model armed with attention mechanism seems a natural fit for this task, and achieves the state-of-the-art performance recently. However, previous attention mechanisms proposed for ABSA may attend irrelevant words and thus ruin the performance, especially when dealing with long and complex sentences with multiple aspects. In this paper, we propose a novel architecture named Hierarchical Gate Memory Network (HGMN) for ABSA: firstly, we employ the proposed *hierarchical gate mechanism* to learn to select the related part about the given aspect, which can keep the original sequence structure of sentence at the same time. After that, we apply Convolutional Neural Network (CNN) on the final aspect-specific memory. We conduct extensive experiments on the SemEval 2014 and Twitter dataset, and results demonstrate that our model outperforms attention based state-of-the-art baselines.

1 Introduction

Aspect-based sentiment analysis (ABSA) is an important subtask in sentiment classification, and has attracted much attention in recent years [Pontiki *et al.*, 2014; Liu, 2012]. The main task is to identify the sentiment polarity (i.e., positive, neutral, or negative) expressed towards the given aspect. Given that the polarities could be same or opposite when different aspects are considered, in order to predict sentiment polarity correctly, an essential step is to select *aspect-specific effective text spans*[1].

Considering have achieved great success in many areas, Recurrent Neural Networks (RNNs) armed with attention mechanism, proposed in machine translation [Bahdanau *et al.*, 2014], seem a natural solution for this problem, and actually achieve the state-of-the-art performance recently. For example, recent works [Wang *et al.*, 2016; Tang *et al.*, 2016b; Chen *et al.*, 2017; Liu and Zhang, 2017; Ma *et al.*, 2017] apply attention mechanism to enforce the model to pay more attentions on the related part of the sentence about the given aspect, and finally get a weighted contextual representation for sentiment prediction. However, these attention-based methods have some drawbacks. Firstly, the attention mechanism can be viewed as the process of one-shot soft selection in general and thus may attend irrelevant words and introduce noise in predicting the sentiment polarity of the specific aspect especially when dealing with long and complex sentences with multiple aspects. Secondly, the aggregated contextual representation obtained by the attention mechanism is usually used as input of following fully connected layer. We conjecture that this simple fully connected layer may not have enough capacity to learn to distinguish complex linguistic structures such as negation sentence, comparative sentence and even the noise introduced by the attention mechanism effectively. Moreover, after such one-shot soft selection, the original sequence structure of sentence is destroyed and a lot of useful information such as word order is lost, which prevents the use of more effective text classifier. We show such drawbacks by several misclassified examples by ATAE-LSTM [Wang *et al.*, 2016], let us consider the following sentences with attention scores shown in Figure 1 (The color depth represents the attention scores and the given aspects are in bold): In sentence (1), ATAE-LSTM attends the incorrect words "worst"; while in sentence (2), although ATAE-LSTM learns to attend the correct words "not" and "dissapointed", the prediction is still wrong mainly because the model can't learn a meaningful semantic combination of different parts mainly due to the mentioned simple fully connected layer.

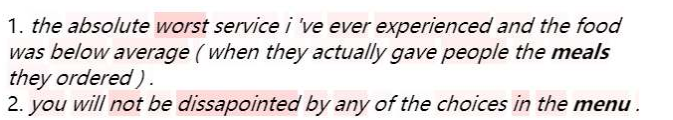


Figure 1: Attention Visualization.

In this paper, we propose a new architecture, named **H**ierarchical **G**ate **M**emory **N**etwork (HGMN), to solve the above issues based our observations. Firstly, we introduce a novel *Hierarchical Gate Mechanism* (HGM) to distill out aspect-specific effective text spans to construct an aspect-

[1]In this paper, we call the related part about the given aspect of the sentence as aspect-specific effective text spans, or effective text spans shortly.

specific memory representation layer by layer. For example, in the sentence "*Granted the space is smaller than most, it is the best service you will find in even the largest of restaurants.*", the sentiment polarity about aspect *space* can be determined by "*Granted the space is smaller than most,*" only. The goal of HGM is to distill out such aspect-specific effective text spans in sentence instead of only the aggregated contextual representation. During the process of selection, HGM also builds the contextualized memory with RNNs model on effective text spans per layer, and the process can be repeated by the hierarchical structure to generate a dynamically adjustable aspect-specific memory representation. As we mention earlier, the main advantage is that it can keep the sequence structure while selecting the aspect-specific effective text spans. After that, we apply Convolutional Neural Network (CNN) on the final aspect-specific memory for sentiment prediction.

Meanwhile, we also make some attempts to enhance the selective ability of HGM. Intuitively, the part-of-speech tag (POS tag) information could be helpful for this task, that is, words with different POS tag such as "food" and "the" should play different roles in determining the final sentiment polarity. Similarly, the relevant distance between the aspect and the context word may also have different roles. Therefore, we leverage the POS tag and position information to enhance the selective capacity.

2 Related Work

Aspect-based sentiment analysis (ABSA) can be considered as a fine-grained sentiment analysis task which aims at identifying the sentiment polarity of a sentence expressed towards a specific aspect [Pontiki *et al.*, 2014]. The traditional methods for aspect-based sentiment analysis mostly rely on rich manual features and syntactic dependencies [Nasukawa and Yi, 2003; Pang *et al.*, 2002; Turney, 2002], which are labor intensive and can't model the relatedness between the aspect and context words. Jiang et. al [Jiang *et al.*, 2011] first emphasize the importance of targets according to their manual evaluation results that 40% of sentiment classification errors are caused by not considering aspects. Since then, research on ABSA has focused on how to better take aspect information into account, some related works are described below.

2.1 Neural Network for Aspect-based Sentiment Analysis

Neural Networks have achieved competitive result on ABSA in recent years [Dong *et al.*, 2014; Tang *et al.*, 2016a; Xue and Li, 2018; Li *et al.*, 2018]. For example, Dong et al. [Dong *et al.*, 2014] propose an Adaptive Recursive Neural Network (AdaRNN) to model the adaptive propagations of the sentiment words to specific aspect which totally rely on syntactic dependency tree. Tang et al. [Tang *et al.*, 2016a] divide a sentence into two parts and use two LSTMs to model the relatedness of a target with its left context and right context respectively.

2.2 Attention and Memory Network for Aspect-based Sentiment Analysis

Attention mechanism has been successfully used in ABSA. Wang et al. [Wang *et al.*, 2016] first introduce attention mechanism to attend the important parts of a sentence about specific aspect and indeed achieve state-of-the-art performance at that time. To emphasize the importance of separate modeling of aspect, Ma et al. [Ma *et al.*, 2017] design an interactive attention network (IAN) which use two attention networks to model the aspect and context interactively, they argue that the aspect and context can be modeled separately but learned from their interaction and their experimental results also prove this point. All these attention-based models have the same problem that they eventually generate a weighted contextual representation which may involve irrelevant noise and destroy the sequence structure of the sentence.

Memory Network is first proposed for question answering and language modeling by Weston et al. [Weston *et al.*, 2014]. Tang et al. [Tang *et al.*, 2016b] first introduce deep memory network to ABSA task by applying multi-attentions on the memory consisted of stacked context word vectors. Based on Tang et al. [Tang *et al.*, 2016b]'s work, Chen et al. [Chen *et al.*, 2017] develop a tailor-made memory which is first produced by a bidirectional LSTM and then take the position information into consideration, after that they pay multiple attentions on the memory and combine the results of attentions by GRU to promote the power of handling complicated sentences. But their memory built for specific aspect simply use the shallow representation of the sentence and merely consider the position information without aspect information.

3 Proposed Model

Aspect-based sentiment analysis task can be formulated as follows: given a sentence $\mathcal{S}$ with n words $\{w_1, w_2, \ldots, w_n\}$ and an aspect $\mathcal{A}$ with m words $\{w_1^a, w_2^a, \ldots, w_m^a\}$, where the sentence contains the aspect, the task aims at predicting the sentiment polarity $y \in \{positive, neutral, negative\}$ expressed on the aspect by the sentence. In this section, we detailedly describe our proposed model designed for this problem. The architecture of our model is shown in Figure 2.

3.1 Input

In our model, the inputs have four kinds of representation: word embedding, POS tag embedding, position embedding and aspect encoding[2].

Word embedding. For each word w_i in the considered sentence, we create a word representation $x_i \in \mathbb{R}^{d_w}$. The considered sentence is denoted as $\mathbf{X} = \{x_1, x_2, \ldots, x_n\}$, which is viewed as the initial memory block. In a similar way, for each word w_i^a in the considered aspect, we also create a word representation $x_i^a \in \mathbb{R}^{d_w}$ and the considered aspect is denoted as $\mathbf{X^a} = \{x_1^a, x_2^a, \ldots, x_m^a\}$.

[2]One natural extension is taking the dependency parsing trees into consideration to model the relationship between the aspect and context words. Here for the simplicity of our model, we leave that for the future work.

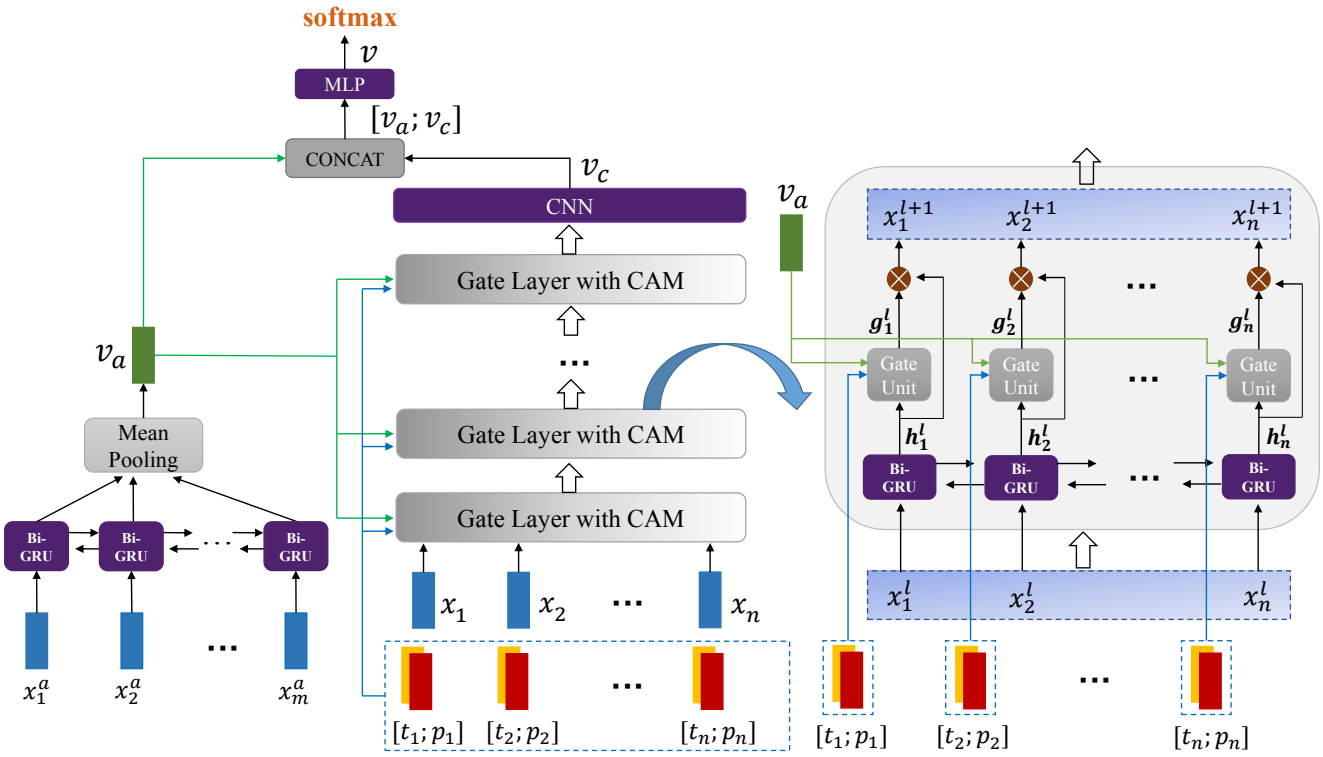


Figure 2: Model architecture. The yellow and red blocks represent the POS tag and position embedding respectively.

POS Tag embedding. Here the main idea of POS tag embedding is to map each POS tag type into a real-valued vector. For each word w_i in the considered sentence, we create a POS tag embedding $t_i \in \mathbb{R}^{d_t}$ based on its POS tag. The POS tag embedding matrix of the considered sentence is denoted as $\mathbf{T} = \{t_1, t_2, \ldots, t_n\}$. The POS tag embedding is fine-tuned during the training phase to learn a tailor-made representation.

Position embedding. For each word w_i in the considered sentence, we create a position embedding $p_i \in \mathbb{R}^{d_p}$ based on the relevant distance v_i between w_i and aspect terms and the position embedding matrix of the considered sentence is denoted as $\mathbf{P} = \{p_1, p_2, \ldots, p_n\}$. The position embedding is fine-tuned during the training phase to learn a tailor-made representation. Specifically, we calculate v_i as follows[3]:

$$v_i = \begin{cases} 0, & \text{if } l \leq i \leq r \text{ or } i > n \\ \min(abs(i-l), abs(i-r)), & \text{otherwise} \end{cases}$$

where l, r is the index of the first and the last aspect word respectively and $abs\,(\cdot)$ means absolute value function.

Aspect Encoding. Here we employ a bidirectional GRU model to learn a meaningful aspect representation instead of simply using vanilla word embedding. For the considered aspect, we create the aspect representation $v_a \in \mathbb{R}^{2d_h}$ by averaging the hidden states of bidirectional GRU run on the aspect word embeddings $\mathbf{X}^a = \{x_1^a, x_2^a, \ldots, x_m^a\}$, where d_h denotes the dimension of the hidden state of the bidirectional GRU here.

[3]The index i may be bigger than the length n of the sentence due to sentence padding.

3.2 Gate Layer

In order to select the aspect-specific effective text spans and keep the sequence structure of the sentence, we propose to design a gate layer instead of previous attention-based approaches.

The gate layer is composed of n gate units. The gate units take the word embeddings $\mathbf{X} = \{x_1, x_2, \ldots, x_n\}$, aspect encoding v_a, POS tag embedding $\mathbf{T} = \{t_1, t_2, \ldots, t_n\}$ and position embedding $\mathbf{P} = \{p_1, p_2, \ldots, p_n\}$ as inputs, then they output scalar gate values $\mathbf{G} = \{g_1, g_2, \ldots, g_n\}$ for every word in the sentence[4]. Here we combine the position and POS tag embedding to calculate gate value. After the gate layer, the new memory representation for w_i is calculated as $x_i^{new} = g_i \otimes x_i$, where the $\otimes$ operation means elementwise multiplication and $\mathbf{G}$ is computed as follows:

$$\mathbf{H} = [\mathbf{X}; \mathbf{T}; \mathbf{P}; v_a \odot e_N] \quad (1)$$

$$\mathbf{M} = W_m tanh\,(W_h \mathbf{H} + b_h) \quad (2)$$

$$\mathbf{G} = \sigma(\mathbf{M}) \quad (3)$$

where v_a is the aspect representation. The $\odot$ operation means: $v_a \odot e_N = [v_a, v_a, \ldots, v_a]$, that is, the operator repeats v_a for n times. σ denotes the logistic sigmoid function. $tanh\,(\cdot)$ means the hyperbolic tangent function. e_N is a column vector with n 1s, and parameter matrix $W_h \in \mathbb{R}^{d_w \times (d_w + d_t + d_p + 2d_h)}$, $W_m \in \mathbb{R}^{1 \times d_w}$, $b_h \in \mathbb{R}^{d_w}$.

[4]We call $\mathbf{G}$ gate values in the remainder of this paper.

3.3 Gate Layer with Context-Aware Memory

The gate layer takes the vanilla pre-trained word embedding as the initial memory representation, similar to the MemNet[Tang *et al.*, 2016b] which utilizes the pre-trained word embeddings as memory block. In this way, although the word vector can reveal the words' similarity in the original embedding space, it can't obtain the context-aware representation, which is very important for text classification intuitively. It is straightforward to tackle the drawback with powerful deep recurrent models, in particular their gated versions, Long Short-Term Memory (LSTM) networks [Hochreiter and Schmidhuber, 1997] and Gated Recurrent Units (GRU) [Cho *et al.*, 2014], as done in [Chen *et al.*, 2017; Li *et al.*, 2018]. We employ bidirectional GRU to obtain the context-aware memory representation. We denote bidirectional GRU here as two GRUs: the forward one GRU_f and the backward one GRU_b. Take GRU_f as an example, the backward one does the same thing, except that its input sequence is reversed, at time step t, the GRU_f observes an tth element x_t of $\mathbf{X}$ and the update process of its internal hidden state h_t is as follows:

$$r_t = \sigma\left(W_r x_t + U_r h_{t-1}\right) \tag{4}$$

$$z_t = \sigma\left(W_z x_t + U_z h_{t-1}\right) \tag{5}$$

$$\tilde{h}_t = tanh\left(W_h x_t + U_h\left(r_t \otimes h_{t-1}\right)\right) \tag{6}$$

$$h_t = z_t \otimes h_{t-1} + (1 - z_t) \otimes \tilde{h}_t \tag{7}$$

where $W_r \in \mathbb{R}^{d_h \times d_w}$, $U_r \in \mathbb{R}^{d_h \times d_h}$, $W_z \in \mathbb{R}^{d_h \times d_w}$, $U_z \in \mathbb{R}^{d_h \times d_h}$, $W_h \in \mathbb{R}^{d_h \times d_w}$, $U_h \in \mathbb{R}^{d_h \times d_h}$ are weight parameters and σ denotes the logistic sigmoid function. The update gate r_t controls the update extent of the output from a new hidden state $\tilde{h}_t$ and the reset gate z_t controls how much information from the previous hidden state is allowed to be kept.

The forward one GRU_f outputs a hidden state list $H_f = \{h_1^f, h_2^f, \ldots, h_n^f\}$, similarly, the backward one GRU_b also outputs another hidden state list $H_b = \{h_1^b, h_2^b, \ldots, h_n^b\}$. We concat two hidden state lists to produce the final context-aware memory representation $H = \{h_1, h_2, \ldots, h_n\}$, where $h_i = \left[h_i^f; h_i^b\right] \in \mathbb{R}^{2d_h}$. The final context-aware memory representation is used as the input to gate units to calculate gate values, as described in Section 3.2.

3.4 Hierarchical Gate Memory Network

Considering the advantage that the gate layer can keep the sequence structure of sentence while selecting the effective text spans, it's natural to extend the single gate layer into hierarchical structure. Intuitively the selective ability of the gate layer can be enhanced through hierarchical structure, because such hierarchical structure can shorten the length of the effective text spans layer by layer, that is, the higher layer can benefit from the output of the lower layer given that the lower layer may have filtered out some unrelated parts and shortened the length of the effective text spans, which is helpful for handling the long and complex sentences.

The structure of HGMN is shown in Figure 2 (bottom right). As we can see, the output of previous gate layer l is $\mathbf{X}^l = \{x_1^l, x_2^l, \ldots, x_n^l\}$ is used as the input to the next gate layer to do further selection and the gate layer $l+1$ output $\mathbf{X}^{l+1} = \{x_1^{l+1}, x_2^{l+1}, \ldots, x_n^{l+1}\}$ similarly. The internal structure of each gate layer here is described in Section 3.3, which is context-aware. Specifically, the first gate layer after input layer doesn't employ bidirectional GRU for memory building[5].

3.5 CNN

CNN should be a suitable method for feature extractor, whose capability for extracting informative n-gram features as sentence representations has been verified in [Kim, 2014]. The final gate layer's output is fed into a CNN layer to extract the most informative n-gram features as sentence representation. Specifically, a filter $w_c \in \mathbb{R}^{2kd}$ is applied to k concatenated consecutive hidden states $h_{i:i+k-1} \in \mathbb{R}^{2kd}$ to compute c_i, one value in the feature map corresponding to the filters:

$$c_i = f\left(w_c^{\mathrm{T}} h_{i:i+k-1} + b\right) \tag{8}$$

where f is the rectified linear unit function and $b \in \mathbb{R}$ is a bias term. A max-pooling operation is further applied over the feature map $c = (c_1, c_2, \ldots, c_{N-k+1})$, to capture the most important semantic feature $\tilde{c}$ in each feature map:

$$\tilde{c} = max\{c\} \tag{9}$$

and $\tilde{c}$ is the feature generated by filter w_c. We denote the extracted vector representation as $v_c \in \mathbb{R}^s$, where s is the number of feature map.

3.6 Model Training

The final representation of the sentence about specific aspect, v is obtained as follows:

$$v = W\left[v_c; v_a\right] + b \tag{10}$$

where parameter matrix $W \in \mathbb{R}^{r \times (s+2d_h)}$, $b \in \mathbb{R}^r$, then it serves as the feature and is fed into a softmax layer to predict the aspect sentiment distribution $\tilde{y}$.

The model is trained by minimizing the cross entropy using back propagation algorithm, the loss function is defined as follows[6]:

$$\mathcal{L} = -\sum_{(x,y)\in D}\sum_{c\in C} y_c \log \tilde{y}_c \tag{11}$$

where C is the sentiment label set, $y \in \mathbb{R}^{\|C\|}$ is a one-hot vector where only the element for the true label is 1, $\tilde{y} \in \mathbb{R}^{\|C\|}$ is the predicted sentiment distribution.

4 Experiments

4.1 Experimental Setup

We conduct experiments on three open standard datasets. The restaurant and laptop dataset are from SemEval 2014 [Pontiki

[5]Under such setting, the experiment results are usually better.

[6]One natural extension of our model is to add some regularization terms in order to guide the hierarchical gate mechanism to select more accurately, and actually we have tried to add a heuristic one to encourage the model to select less parts as the layer gets higher, which only improves the performance slightly. Here for simplicity, we omit such regularization terms and leave how to design better regularization terms for a future work.

Dataset		Postive	Negative	Neutral	Total
Laptop	Train	994	870	464	2328
	Test	341	128	169	638
Restaurant	Train	2164	807	637	3608
	Test	728	196	196	1120
Twitter	Train	1561	1560	3127	6248
	Test	173	173	346	692

Table 1: Statistics of the experiments datasets.

et al., 2014]. Twitter dataset is a collection of tweets, introduced by [Dong *et al.*, 2014]. For the first two datasets, following [Tang *et al.*, 2016b]'s setting, we remove the training examples with the "conflict" label. All of the three datasets have three labels: positive, negative, and neutral. The detailed statistics of the datasets are shown in Table 1.

We first obtain the POS tag results from Stanford Parser [Manning *et al.*, 2014] and we use pre-trained GloVe vectors [Pennington *et al.*, 2014] to initialize the word embeddings, the dimension is 300 and the vocabulary size is 2.2M[7]. For out-of-vocabulary words, we randomly initialize their embeddings from the uniform distribution $\mathbf{U}(-0.01, 0.01)$. The convolutional kernel size k and the number of feature map s here we adopt is 3 and 50. The hidden state size, postion and POS tag embedding dimensions, i.e., d_h, d_p, d_t are set to 50. To ease overfitting, we apply dropout on the input word embeddings of the bidirectional GRU and the final sentence representation v with dropout rate 0.5. All weight matrices are initialized with the uniform distribution $\mathbf{U}(-0.01, 0.01)$ and the biases are initialized as zeros. Adam [Kingma and Ba, 2014] is adopted as the optimizer here. The batch size is set to 25 and the learning rate is set to 0.001. Specifically, all hyperparameters are tuned on 20% randomly held-out training data and the hyper-parameter collection producing the highest accuracy score is used for testing.

4.2 Compared Methods

HGMN is compared with the following methods:

- **AdaRNN** [Dong *et al.*, 2014]: It employs an Adaptive Recursive Neural Network (AdaRNN) to model the adaptive propagations of the sentiment words to specific target which rely on dependency tree;
- **ATAE-LSTM** [Wang *et al.*, 2016]: It uses attention mechanism to capture the key part of the sentence about a given aspect and incorporates the aspect embedding with each word embedding as input to make the better use of aspect information at the same time;
- **IAN** [Ma *et al.*, 2017]: It utilizes two attention blocks to model the target and context interactively and uses one's representation to help better generate the representation of the other;
- **TD-LSTM** [Tang *et al.*, 2016a]: It uses two LSTMs to model the relatedness of the target with its left context and right context respectively, and then concatenates the left and right target-dependent representation for sentiment prediction;
- **MemNet** [Tang *et al.*, 2016b]: It applies multiple attentions on the memory consisted of stacked context word vectors and uses the result of previous attention to help the next attention attend more accurate information;
- **RAM** [Chen *et al.*, 2017]: It uses multiple attentions on a tailor-made memory which first produced by a bidirectional LSTM and then take the position information into consideration, different hops' attention results are nonlinearly combined by GRU;
- **GCNN** [Xue and Li, 2018]: It is a model based on convolutional neural networks and gate mechanism, which proposes a Gated Tanh-ReLU Unit to select the sentiment features about specific aspect;
- **TNet** [Li *et al.*, 2018]: It first applies a target specific transformation component to better integrate target information into the word representation and then uses CNN classifier for sentiment classification;
- **HGM-MLP**: HGMN-MLP is the same model as HGMN described in Section 3 except that the CNN layer in HGMN is replaced by a fully connected layer. The average word representations is fed into the layer, which is similar to the way the attention mechanism works.

All of the neural approaches are implemented in Pytorch[8], and we adopt the parameters in corresponding reference paper.

4.3 Results and Analysis

Main experiment results can be seen in Table 2. The main evaluation metrics are Accuracy and Macro-averaged F1-score. Generally speaking, Our model outperforms all the attention-based models such as ATAE-LSTM, IAN, and RAM, which verifies the effectiveness of our HGMN model. TNet-AS achieves best performance among all baseline methods on Laptop and Twitter due to its target-specific transformation component. Our HGMN model achieves 1.83% and 1.16% improvements on Restaurant and Twitter dataset compared with TNet-AS.

On the other hand, HGM-MLP, HGMN model with the CNN layer replaced by a fully connected layer, also outperforms all the attention based models, the main difference between HGM-MLP and attention-based models is the difference between the hierarchical gate mechanism and the attention mechanism, which shows the effectiveness of the former in soft selection. HGM-MLP improves the performance about 3% on Laptop and Restaurant dataset compared with ATAE-LSTM. Moreover, Our HGMN model outperforms HGM-MLP further, which verifies the advantage of keeping the sequence structure of sentences and utilizing CNN as the feature extractor.

4.4 Impact of the Number of Gate Layer

To investigate the impact of the number of gate layers, we evaluate our model with 1 to 3 gate layers. As shown in Ta-

[7]https://nlp.stanford.edu/projects/glove/

[8]https://pytorch.org/

Model	Laptop		Restaurant		Twitter	
	Accuracy	Macro-F1	Accuracy	Macro-F1	Accuracy	Macro-F1
AdaRNN [Dong *et al.*, 2014]	N/A	N/A	N/A	N/A	66.30*	65.90*
TD-LSTM [Tang *et al.*, 2016a]	72.10	64.03	78.66	67.84	70.38	68.07
ATAE-LSTM [Wang *et al.*, 2016]	72.22	66.98	78.12	64.14	71.24	69.92
IAN [Ma *et al.*, 2017]	73.33	65.00	79.41	68.50	71.82	69.57
MemNet [Tang *et al.*, 2016b]	71.47	65.34	78.57	67.33	70.38	68.35
RAM [Chen *et al.*, 2017]	74.29	70.26	79.64	68.43	71.97	69.73
GCNN [Xue and Li, 2018]	72.84	59.70	79.31	65.55	72.11	70.09
TNet-AS [Li *et al.*, 2018]	**76.79**	71.75	80.50	70.60	72.83	71.49
HGM-MLP	75.08	70.19	81.08	71.79	72.54	70.56
HGMN w/o *POS tag*	75.71	71.80	81.17	71.29	72.69	71.41
HGMN w/o *position*	74.29	68.47	80.72	69.85	72.69	70.83
HGMN w/o *both*	73.70	69.90	80.27	68.22	71.99	70.00
HGMN	76.67	**72.22**	**82.33**	**73.34**	**73.70**	**72.89**

Table 2: Experimental results(%). Those with symbol $*$ are from [Dong *et al.*, 2014]. N/A means *not available*. The best results are in bold.

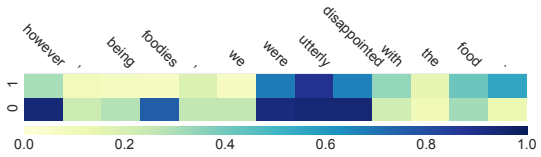
Model	Laptop		Restaurant		Twitter	
	Accuracy	Macro-F1	Accuracy	Macro-F1	Accuracy	Macro-F1
HGMN-1GL	76.03	71.48	80.99	72.71	72.98	71.76
HGMN-2GL	**76.67**	**72.22**	**82.33**	**73.34**	73.70	**72.89**
HGMN-3GL	75.24	70.36	81.26	72.20	**74.86**	72.78

Table 3: Experimental results of different gate layers setting. Specially, for the HGMN-1GL model, differnet from the first gate layer introduced in Section 3.4, here we employ the Bi-GRU to build memory instead of feeding original word embedding into following gate unit.

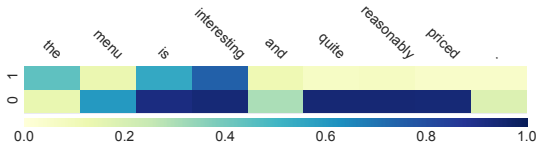
ble 3, in general, our model with 2 gate layers works better, but the advantage is not always there for different datasets. For example, on the Twitter dataset, our model with 3 gate layers performs the best in terms of Accuracy. We can conclude that using 1 gate layer is generally not as good as using more, which shows the effectiveness of the proposed hierarchical gate mechanism, on the other hand, as the model's layer increases, the complexity of model increases and becomes more difficult to train and less generalizable.

4.5 Impact of POS Tag Embedding and Position Embedding

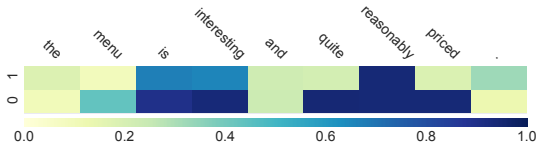
To investigate the impact of position embedding and POS tag embedding, we perform some ablation studies. HGMN w/o *sth* represents the HGMN model with *sth* removed. As shown in Table 2, we can see that both Accuracy and F1-score of HGMN w/o *both* drop significantly. However, even in this way, HGMN w/o *both* outperforms most of attention-based models such as ATAE-LSTM and IAN, which verifies the effectiveness of our hierarchical gate mechanism. While removing position embedding only, HGMN w/o *position* drops about 2% on all the datasets but still performs better than HGMN w/o both. On the other hand, HGMN w/o *POS tag* drops about 1% on all the datasets but still performs better than HGMN w/o *both*, from which we can conclude that both position embedding and POS tag embedding are beneficial for HGMN, i.e. both can help the gate mechanism select more accurately.



(a) An test example with single aspect only: "however , being foodies , we were utterly disappointed with the food .", where the "food" is the given aspect term.



(b) An test example contains more than one aspect: "the menu is interesting and quite reasonably priced .", where the "*menu*" is the given aspect term.



(c) Same test example as previous figure, where the "*priced*" is the given aspect term here.

Figure 3: Gate visualization.

Sample Sentences	ATAE-LSTM	RAM	HGMN
1. would you ever believe that when you complain about over an hour *wait*$_{\mathbf{N}}$, when they tell you it will be 20-30 minutes , the *manager*$_{\mathbf{N}}$ tells the *bartender*$_{\mathbf{O}}$ to spill the *drinks*$_{\mathbf{O}}$ you just paid for ?	(**N**, **N**, **N**, **O**)	(**N**, **N**, **N**, **N**)	(**N**, **N**, **O**, **O**)
2. how can they survive serving *mediocre food*$_{\mathbf{O}}$ at exorbitant *prices*$_{\mathbf{N}}$? !	(**N**, **N**)	(**N**, **N**)	(**O**, **N**)
3. the *decor*$_{\mathbf{P}}$ is what initially got me in the door .	(**N**)	(**N**)	(**P**)
4. *price*$_{\mathbf{N}}$ was higher when purchased on mac when compared to *price*$_{\mathbf{P}}$ showing on pc when i bought this product .	(**O**, **O**)	(**P**, **P**)	(**N**, **P**)
5. yes , they 're a bit more expensive then typical , but then again , so is their *food*$_{\mathbf{P}}$.	(**N**)	(**N**)	(**P**)
6. *new hamburger with special sauce*$_{\mathbf{P}}$ is ok - at least better than *big mac*$_{\mathbf{N}}$!	(**O**, **O**)	(**O**, **O**)	(**O**, **N**)

Table 4: Some test examples and predictions of ATAE-LSTM, RAM and HGMN, aspect terms are underlined with the true labels given as subscripts. Incorrect predictions are highlighted in red color.

4.6 Visualization

We randomly pick some examples from test set and visualize the gate values obtained by Equation (3) to check whether the proposed hierarchical gate mechanism conforms with our intuition. The visualization results are depicted in Figure 3.

Figure 3 presents several examples with only one aspect (Figure 3(a)) or multiple aspects (Figure 3(b) and 3(c)). As shown in Figure 3, generally speaking, the lower gate layer trends to select the meaningful parts, then the higher gate layer distills out effective text spans further and concentrates on the most related part the given aspect. For example, in Figure 3(a), the first gate layer selects the major part such as "however", "foodies", and "disappointed", then the second gate layer focuses on the really related part such as "utterly", and "disappointed".

4.7 Case Study

We pick some test examples from test set to convince the performance of HGMN further. Two attention-based models, i.e., ATAE-LSTM and RAM, are compared with HGMN in Table 4. Generally speaking, our HGMN is better at handling long and complex sentence structures. For example, sentence (1)(2)(4) are typical long reviews and mention multiple aspect terms. When facing with such long sentences, attention-based models usually can't attend to the related part well, ATAE-LSTM and RAM both make wrong prediction on some aspects, while our HGMN still can make correct predictions on all four aspect terms due to the hierarchical gate mechanism. What's more, when facing the cases where the semantics of the whole sentence needed to be understood, HGMN works pretty well. For example, sentence(3)(5) contain no obvious sentiment words about the given aspect but the whole sentence indeed expresses positive polarity on the given aspect, the model needs to capture the semantics in whole to correctly predict the sentiment. On the other hand, there are some cases HGMN struggles with. For example, HGMN misclassify sentence(6) as *neutral*, so do ATAE-LSTM and RAM. Actually most of error cases are similar, that is, misclassification between neutral and postive/negtive, as future work, we can consider improving the accuracy in such cases.

5 Conclusions and Future Work

In this paper, to solve some drawbacks of previous attention-based approaches for ABSA, we propose a novel architecture named Hierarchical Gate Memory Network (HGMN) which can keep the sequence structure of sentence instead of obtaining the aggregated contextual representation merely when selecting the effective text spans. At first, HGMN employs the hierarchical gate mechanism to filter out the unrelated part of sentences to build the dynamically adjustable aspect-specific memory by taking the aspect representation, word representation, POS tag and position information into consideration simultaneously. After that, we apply the effective CNN layer to extract the most informative n-gram features as sentence representation. Extensive experiments on the SemEval 2014 and Twitter dataset demonstrate that our model outperforms several attention-based state-of-the-art baselines. How to leverage dependency parsing trees and design better regularization terms to guide the hierarchical gate mechanism would be our future work.

Acknowledgments

This paper is supported by the National Key Research and Development Program of China (Grant No.2018YFB1403400), the National Natural Science Foundation of China (Grant No. 61876080), the Collaborative Innovation Center of Novel Software Technology and Industrialization at Nanjing University.

References

[Bahdanau *et al.*, 2014] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[Chen *et al.*, 2017] Peng Chen, Zhongqian Sun, Lidong Bing, and Wei Yang. Recurrent attention network on memory for aspect sentiment analysis. In *Proceedings of the*

2017 Conference on Empirical Methods in Natural Language Processing, pages 452–461, 2017.

[Cho *et al.*, 2014] Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014*, pages 1724–1734, 2014.

[Dong *et al.*, 2014] Li Dong, Furu Wei, Chuanqi Tan, Duyu Tang, Ming Zhou, and Ke Xu. Adaptive recursive neural network for target-dependent twitter sentiment classification. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 49–54, 2014.

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

[Jiang *et al.*, 2011] Long Jiang, Mo Yu, Ming Zhou, Xiaohua Liu, and Tiejun Zhao. Target-dependent twitter sentiment classification. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, pages 151–160, 2011.

[Kim, 2014] Yoon Kim. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1746–1751, 2014.

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

[Li *et al.*, 2018] Xin Li, Lidong Bing, Wai Lam, and Bei Shi. Transformation networks for target-oriented sentiment classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 946–956, 2018.

[Liu and Zhang, 2017] Jiangming Liu and Yue Zhang. Attention modeling for targeted sentiment. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*, pages 572–577, 2017.

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

[Ma *et al.*, 2017] Dehong Ma, Sujian Li, Xiaodong Zhang, and Houfeng Wang. Interactive attention networks for aspect-level sentiment classification. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*, pages 4068–4074, 2017.

[Manning *et al.*, 2014] Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, and David McClosky. The stanford corenlp natural language processing toolkit. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 55–60, 2014.

[Nasukawa and Yi, 2003] Tetsuya Nasukawa and Jeonghee Yi. Sentiment analysis: Capturing favorability using natural language processing. In *Proceedings of the 2nd international conference on Knowledge capture*, pages 70–77, 2003.

[Pang *et al.*, 2002] Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. Thumbs up? sentiment classification using machine learning techniques. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing*, pages 79–86, 2002.

[Pennington *et al.*, 2014] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1532–1543, 2014.

[Pontiki *et al.*, 2014] Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. Semeval-2014 task 4: aspect based sentiment analysis. In *Proceedings of the 8th International Workshop on Semantic Evaluation*, pages 27–35, 2014.

[Tang *et al.*, 2016a] Duyu Tang, Bing Qin, Xiaocheng Feng, and Ting Liu. Effective lstms for target-dependent sentiment classification. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers.*, pages 3298–3307, 2016.

[Tang *et al.*, 2016b] Duyu Tang, Bing Qin, and Ting Liu. Aspect level sentiment classification with deep memory network. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 214–224, 2016.

[Turney, 2002] Peter D. Turney. Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 417–424, 2002.

[Wang *et al.*, 2016] Yequan Wang, Minlie Huang, Li Zhao, et al. Attention-based lstm for aspect-level sentiment classification. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 606–615, 2016.

[Weston *et al.*, 2014] Jason Weston, Sumit Chopra, and Antoine Bordes. Memory networks. *CoRR*, abs/1410.3916, 2014.

[Xue and Li, 2018] Wei Xue and Tao Li. Aspect based sentiment analysis with gated convolutional networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 2514–2523, 2018.