Abstract

Aspect-level affective classification (ASC) aims to detect the affective polarity of a given viewpoint target in a sentence. In the ASC method based on neural network, most of the work uses the attention mechanism to capture the sentiment words corresponding to the opinion target, and then gather them as evidence to infer the emotion of the target. However, due to the complexity of annotation, the scale of aspect level data sets is relatively small. Data scarcity leads to the attention mechanism sometimes unable to pay attention to the sentiment words corresponding to the target, which finally weakens the performance of the neural model. In order to solve this problem, this paper proposes a complete High Interpretable Transfer Network (HTN), which adopts methods such as data enhancement, attention adjustment and transfer to effectively improve the performance of ASC model. A large number of experimental results show that our method has always been all the previous migration methods in this field, even compared with some complex models.

1 Introduction

Aspect-level sentiment classification (ASC) is a fundamental subtask in sentiment analysis (Pontiki et al., 2014). Given a sentence and an opinion target (also called an aspect term) occurring in the sentence, aspect-level sentiment classification aims to infer the sentiment polarity in the sentence towards the target aspect. An opinion target, also known as aspect term, refers to a word or a phrase in review describing an aspect of an entity. For example, the electronic product comment “The speed is fast, but the screen is dark” consists of two aspect terms, namely “speed” and “screen”, and they are associated with positive and negative sentiment respectively.

Traditional methods usually first artificially define a set of features, such as word bags, and then use machine learning methods to train a classifier (such as SVM) (Jiang et al., 2011). These methods depend on artificially defined features, and also need rich priori knowledge, such as constructing an sentiment dictionary. In recent years, with the development of deep learning technology, a number of neural network models (Tang et al., 2016) have been proposed and used in ASC tasks. Usually, these models use supervised learning to train classifiers, so a large amount of labeled data is necessary to obtain promising results. The cost of a large number of annotations is unbearable.

The number of training samples of the existing ASC publicly available data sets is very limited, which limits the performance of the neural network model. As a contrast, online websites such as Yelp contain a huge number of comments. These comments are usually accompanied by a rating score, from one star to five stars, which can intuitively show the user's satisfaction with something. After some simple preprocessing, these comments can be used as training data for document sentiment classification.

Generally speaking, the model for ASC task inputs a sentence and its contained aspect term to output sentiment category, while the model for DSC task inputs a sentence and outputs sentiment category. It can be seen that ASC task and DSC task are actually highly similar. Considering that ASC task lacks data and DSC task has a large number of available training data, a very direct idea is to use DSC data to improve the training of ASC model.

In view of this, He et al., 2018 proposed the PRET+MULT method to improve the shared embedded layer and LSTM layer by combining ASC task training and DSC task training. Similarly,
Chen et al., (2019) also adopted multi task joint training to improve their shared capsule layer. Different from the previous two methods, Zhao et al.,(2020) proposed ATN tried to transfer the attention learned from DSC model to ASC task. The above three works have good performance, but they have some defects in data preprocessing and interpretability, which we will explain in detail in our model part. In this paper, we propose a novel knowledge transfer framework, High Interpretable Transfer Network (HITN), which is more perfect and interpretable. We have made detailed and reliable explanations from data preprocessing to attention transfer. Compared with these three models, we have achieved better results, which also shows the effectiveness of our method. At the same time, it is worth noting that in order to focus on knowledge transfer, our network model structure is not complex. Nevertheless, we still achieved very good results.

In our framework, according to the inherent characteristics of ASC data, we firstly propose a novel DSC data preprocess method, which improves the disadvantage of large amount but insufficient diversity of DSC data, and makes DSC data closer to ASC data in terms of data distribution. In this section, we will also explain the defects of the previous three models in DSC data preprocess. Secondly, inspired by the multi-instance learning method (Ma et al., 2018), we use a simple LSTM model to train the attention weight and sentiment polarity score of each word in the text on the DSC data after data preprocess. Thirdly, we process the text attention distribution obtained in the DSC model to obtain an attention distribution more suitable for the characteristics of ASC tasks. Finally, we take the processed attention distribution as a priori knowledge and inject it into the training of ASC model. We conducted experiments on two SemEval datasets. The final results show that our method can be significantly improved by combining the two attention transfer methods, and is superior to all comparison methods in ASC tasks.

2 Related Work

2.1 Aspect-level Sentiment Classification

Early sentiment classification methods focused on document level sentiment classification tasks (DSC). Later, more fine-grained aspect sentiment classification tasks were proposed separately. At first, most ASC models define features manually, and then use machine learning methods for classification. The definition of these features is very dependent on expert experience and usually takes a lot of time and energy to obtain (Wang et al.,2004).

In recent years, deep learning methods have been widely used in various studies, and ASC is certainly no exception (Tang et al., 2016; Chen et al., 2017). Among these methods, neural network models widely used mainly include: LSTM based, CNN based, GNN based (Tang et al., 2020) and transformer based. Especially in recent years, many models have achieved excellent results by combining dependency tree of the sentence with GNN model.

However, for ASC tasks, no matter how complex the model is, it is difficult to mine enough information to help the model classify correctly in front of limited data. On the other hand, even a simple model can achieve good results after using a large amount of data. Different from many previous models that constantly try to improve the network structure, our HITN uses rich DSC annotation data to help ASC model improve the effect with simple network models.

2.2 Transfer Learning

Transfer learning aims to extract knowledge from one or more source tasks and then apply it to the target task. Previously, in the field of image processing, transfer learning has been proved to be effective. He et al., 2018 was the first person to transfer knowledge from document level data to improve ASC tasks by sharing the embedded layer and LSTM layer. Similar to their approach, Chen et al., 2019 used a capsule network to share bottom features between ASC and DSC tasks. Zhao et al., 2020 guided the training of ASC model by transferring attention. In this paper, our goal is to mine useful information from DSC data as much as possible according to the characteristics of ASC task. Therefore, from data preprocessing to training model design, we have fully considered the respective characteristics of ASC task and DSC task. Finally, experiments show that our method has better performance and better interpretability than the previous three models.

3 HITN

Figure 1 shows the overall architecture of the High Interpretable Transfer Network (HITN). It mainly consists of four parts: the DSC data
3.1 Task Formalization

**DSC Formalization** For a review document \(d\) from the DSC dataset \(D\), we regard it as a long sentence \((w_1^d, w_2^d, ..., w_n^d)\) consisting of \(n\) words. DSC aims to determine the overall sentiment polarity of the review document \(d\).

**ASC Formalization** Formally, given a sample \(<s, t>\) from the ASC dataset \(A\), \(s = \{w_1, w_2, ..., w_n\}\) is a review sentence consisting of \(n\) words and \(t = \{w_1, w_{l+1}, ..., w_{r}\}\) is a given aspect term containing \(|r - l|\) words. The aspect term \(t\) is a continuous subsequence of \(s\). The goal of ASC is to predict the sentiment polarity (i.e., positive, neutral and negative) of the aspect term \(t\) in the sentence \(s\).

3.2 DSC Data Preprocessing

In the previous three models using DSC data to help ASC training, there is a problem that is often overlooked, that is, how to label data with sentiment. They usually classify comments according to the number of stars. One star and two-star comments are divided into negative, four-star and five-star comments are regarded as positive, and three-star comments are regarded as neutral. Then label each text according to this classification method and send it to training. All this seems logical, but there is a crucial question here, that is, whether three-star comments should be regarded as neutral comments. In fact, the neutral comment in ASC task means that the comment does not express any sentiment towards the target word. For example, the sentence "I had a meal in this restaurant at 3 p.m." is neutral to the target word "restaurant", because it does not make any evaluation on the restaurant. However, this is not the case for neutral comments in DSC tasks. Usually, commentators think something is good, but it is not so good or so bad. For example, we find that many commentators use "good" in three-star

![Figure 1: The overall architecture of HITN, which consists of four parts: DSC data preprocessing module, pre-trained DSC module, ASC module, and attention transfer module.](image-url)
comments and "great" in five-star comments. Obviously, we cannot regard "good" as a neutral evaluation.

Therefore, strictly speaking, there is no neutral comment in DSC comments for ASC tasks. To this end, we give up using three-star comments and only use other star comments. We divide one-star and two-star comments into negative comments, while four-star and five-star comments into positive comments.

Further, we found that both the four-star comments and the five-star comments are actually very satisfied with all aspects of the product, and there are few negative parts in these comments. At the same time, both one-star comments and two-star comments are actually dissatisfied with all aspects of the product, and there are few positive comments in these comments. This is completely different from ASC. Many comments in ASC often have both positive and negative parts. For example, the sentence "food is delicious, but service is terrible" gives a positive evaluation of "food" and a negative evaluation of "service".

In order to make the distribution of DSC data more in line with the characteristics of ASC data, we adopt the method of sentence splicing. Specifically, a positive comment and a negative comment are randomly selected from DSC comments, and they are spliced into a sentence. When the positive comments are in the front and the negative comments are in the back, the spliced sentences are labeled with [0.5, - 0.5], while when the negative comments are in the front and the positive comments are in the back, the spliced sentences are labeled with [-0.5, 0.5]. In addition, positive comments are labeled 1, while negative comments are labeled -1.

Through the above methods, the number of DSC data is increased, and its distribution is more in line with the needs of ASC tasks.

3.3 Pre-trained DSC Module

Before transferring attention knowledge, we first pre-train a DSC module on the large-scale preprocessed DSC datasets. In this work, we employ a conventional attention-based BiLSTM as our DSC module.

For a review document \(d = \{w^1, w^2, ..., w^d\}\), we map it into the corresponding word representations \(\{w^d_1, w^d_2, ..., w^d_n\}\) by looking up an embedding table \(E_{\text{emb}} \in \mathbb{R}^{|v| \times d_e}\), where \(|v|\) is the vocabulary size and \(d_e\) denotes the word embedding dimension. Then we use a BiLSTM network to obtain the contextual information for each word and generate a sequence of hidden states \(\{h^d_1, h^d_2, ..., h^d_n\}\). Usually, the attention mechanism is used to get the expression of the sentence by aggregating the word contextual representations.

But we didn't do this because we wanted not only the attention of each word, but also the sentiment polarity of each word.

Therefore, next, we directly use a common classification layer to get the sentiment polarity of each word, and use a vector \(q\) to calculate the attention of each word where \(q\) is a randomly initialized trainable parameter vector. Its significance is to help get the attention of each word.

It is worth noting that the word sentiment here is a scalar, and its value is obtained as follows:

\[
w^d_i = (\sigma(h^d_i) - 0.5) \times 2, \quad i = 1, ..., n
\]

Where \(w^d_i\) is the word sentiment polarity of \(h^d_i\), \(\sigma\) denotes sigmoid function. It can also be seen from the formula, \(w^d_i \in [-1, 1]\).

The word attention here is obtained as follows:

\[
w^a_i = \frac{\exp(f(h^d_i, q))}{\sum_{j=1}^{n} \exp(f(h^d_j, q))}, \quad i = 1, ..., n
\]

\[
f(h^d_i, q) = h^d_i \cdot q + b_d, \quad i = 1, ..., n
\]

Where \(q\) denotes the query vector, which is a trainable parameter vector. \(b_d\) denotes bias.

Finally, we get the sentiment of the whole sentence in the following ways:

\[
d_s = \sum_{i=1}^{n} w^d_i w^a_i
\]

In addition, as mentioned earlier, DSC comments have no neutral data. But at the same time, in each sentence, in fact, there are few words with sentiment polarity, and most words do not actually have sentiment polarity, that is, neutral in the common sense. Therefore, we add a regular term to force the emotion of most words close to 0.

The regularizer term is as follows:

\[
r_d = \sum_{i=1}^{n} w^d_i
\]

The final loss function is as follows:

\[
\text{loss}_d = \frac{1}{|D|} \sum_{d \in D} (d_s - y)^2 + \lambda r_d^2
\]

where \(|D|\) is the size of document data after DSC data preprocessing. \(\lambda\) is hyper-parameter. \(y\) denotes the label of DSC data preprocessing.
Please note that, For a comment spliced by a positive comment and a negative comment, if its label is [0.5, -0.5], it means that the output of the positive part of the first half should be close to 0.5, and the output of the negative part of the second half should be close to -0.5, and vice versa. The value of 0.5 here shows that we force the whole sentence to pay attention to different parts of the sentence at the same time. In other words, we force the attention distribution of the whole sentence to pay attention to words with different sentiment polarities in a sentence at the same time.

3.4 Attention Transfer Module

The attention obtained through the DSC module focuses on the sentiment words in the whole sentence, while the ASC task only focuses on those sentiment words related to aspect term, so it cannot be used directly and needs to be changed. We use our proposed syntax distance decay method to convert DSC attention into ASC task attention.

First of all, let me define the syntax distance. A sentence is built into a dependency tree, also known as a dependency graph, according to the dependency of its words. Each word is also a node in the dependency graph. The shortest distance between two word nodes in the dependency graph is the syntax distance between the two words.

For any word, we calculate the syntax distance from it to the aspect term. If the aspect term includes multiple words, we calculate and take the shortest distance as the syntax distance from the word to the aspect.

Then we use the following methods to process the word attention obtained by DSC:

\[ w_i^{a'} = (1 - \frac{l_i}{\text{max}(l_i)}) w_i^a \]  

(7)

Where \( l_i \) means the syntax distance from the i-th word to the aspect and \( \text{max}(l_i) \) means the maximum value of all words’ syntax distance.

In order to make the sum of attention after syntax distance decay processing still equal to 1, we further do the following processing.

\[ \beta_i = \frac{w_i^{a'}}{\sum_{i=1}^{n} w_i^{a'}} \]  

(8)

Through the above processing, we can get a more attention distribution that meets the needs of ASC.

3.5 ASC Module

As shown on the right side of Figure 1, the architecture of the basic ASC module is similar to that of the DSC module. The difference lies in two points. One is that the ASC task needs to model the aspect term. The other is that we use the traditional method in the ASC model, that is, we obtain the context aware representation of aspect term through the attention mechanism, and then use it for classification. The classification here is still the traditional one hot method, rather than the scalar used in the previous DSC module.

Specifically, given a sentence \( s = \{w_1, w_2, ..., w_n\} \), and an aspect term \( t = \{w_0, w_1, ..., w_t\} \) in \( s \), we first map each word \( w_i \) into its word embedding \( w_i \) by looking up the word embedding table \( E_{emb} \in \mathbb{R}^{|v| \times d_e} \), where \( |v| \) is the vocabulary size and \( d_e \) denotes the word embedding dimension. Secondly, we send word embedding of each word to BiLSTM to get contextual representation of each word, and we make an average pooling to all words contained in aspect term to obtain the representation of aspect term. Thirdly, we use the attention mechanism to obtain attention of each word towards the aspect term as follows:

\[ a_i = \frac{\exp(f(h_i, t))}{\sum_{j=1}^{n} \exp(f(h_i, t))}, \quad i = 1, ..., n \]  

(9)

\[ f(h_i, t) = h_i \cdot W_s + b_s, \quad i = 1, ..., n \]  

(10)

Where \( t \) denotes the the representation of aspect term. \( W_s \) denotes the weight matrix and \( b_s \) denotes bias. \( a_i \) denotes the weight attention obtained from ASC model.

Now we have the processed DSC attention \( \beta_i \) and ASC attention \( a_i \). We use the following methods to fuse the two attentions to obtain the final attention distribution:

\[ y_i = (1 - g) \cdot a_i + g \cdot \beta_i, \quad i = 1, ..., n \]  

(11)

Where \( y_i \) means the fused word attention of the i-th word. \( g \) is a hyper-parameter, which controls the intensity of attention transfer.

Then, the aspect-aware sentence representation \( r_t \) is obtained by aggregating the word contextual representations. And through a classifier, we finally get the aspect-level sentiment classification results:

\[ r_t = \sum_{i=1}^{n} y_i h_i \]  

(12)

\[ \hat{y}_i = \text{softmax}(W_o r_t + b_o) \]  

(13)

\[ loss_a = -\sum_{i \in A} y_i \log(\hat{y}_i) \]  

(14)
Where $y_i$ and $y_t$ respectively are the predictive class distribution and golden class distribution. $W_o$ denotes the weight matrix and $b_o$ denotes bias.

4 Experiments

4.1 Datasets and Settings

Datasets We evaluate our model on two ASC benchmark datasets from SemEval 2014 Task 4 (Pontiki et al., 2014). They respectively contain reviews from Restaurant and Laptop domains. Following previous studies (Tang et al., 2016b; Chen et al., 2017; He et al., 2018), we remove samples with conflicting polarities in all datasets. The statistics of the ASC datasets are shown in Table 1.

Previous models that used DSC data to assist ASC training generally believed that DSC data in similar fields would be helpful to assist ASC data. They generally used the DSC model trained by Amazon comments to assist the aspect sentiment classification of laptop comments and Yelp comment to assist the aspect sentiment classification of restaurant comments.

Different from them, we only use Yelp comments as the training data of DSC model. Because on the one hand, we think that even in different fields, the expression of sentiment should be close. On the other hand, we pay more attention to data quality than similar fields. After our observation, we found that many sentences in the Amazon dataset are not complete sentences, but part of the original comments, and then labeled with the same label as the original sentence. We believe that if we randomly select a part from a positive sentence, we can not guarantee that the extracted part is still positive, it is likely to be neutral, because as we said before, neutral words are the main body of the sentence, and only a few words have positive or negative sentiment. Because of this, we abandoned Amazon dataset and only used Yelp dataset.

Settings In our experiments, word embeddings are initialized by 300-dimension GloVe (Pennington et al., 2014). All the weight matrices and biases are given the initial value by sampling from the uniform distribution $U(-0.1, 0.1)$.

The dimension of LSTM cell hidden states is set to 200. We employ stochastic gradient descent (SGD) with momentum (Qian, 1999) to train models. The initial learning rate and momentum parameter are respectively set to 0.0003 and 0.9. In addition, we apply dropout (Hinton et al., 2012) with probability 0.5. The hyper-parameter $\lambda$ and $g$ are respectively set to 1 and 0.5. Finally, we run each model five times and report the average result of them.

4.2 Compared Methods

To demonstrate the superiority of our HITN for ASC tasks, we compare it with followings baselines: ATAE-LSTM (Wang et al., 2016b), PBAN (Gu et al., 2018), PRET+MULT (He et al., 2018) and TransCap (Chen and Qian, 2019), ATN (Chen et al., 2020), DGEDT (Tang et al., 2020), RGAT (Wang et al., 2020).

4.3 Main Results

It can be clearly seen from table 2 that our model is superior to the previous three models that use DSC data to assist in training ASC. Even r-gat and dgedt, which have relatively complex comparative structure and excellent performance last year, also have certain advantages. These results prove the effectiveness of our model.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Pos</th>
<th>Neg</th>
<th>Neu</th>
</tr>
</thead>
<tbody>
<tr>
<td>Restaurant(Train)</td>
<td>2164</td>
<td>807</td>
<td>637</td>
</tr>
<tr>
<td>Restaurant(Test)</td>
<td>728</td>
<td>196</td>
<td>196</td>
</tr>
<tr>
<td>Laptop(Train)</td>
<td>994</td>
<td>870</td>
<td>464</td>
</tr>
<tr>
<td>Laptop(Test)</td>
<td>341</td>
<td>128</td>
<td>169</td>
</tr>
<tr>
<td>Yelp Review</td>
<td>266k</td>
<td>177k</td>
<td>N/A</td>
</tr>
</tbody>
</table>

Table 1: Statistics of the datasets.
<table>
<thead>
<tr>
<th>Model</th>
<th>Acc(Restaurant)</th>
<th>F1(Restaurant)</th>
<th>Acc(Laptop)</th>
<th>F1(Laptop)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATAE-LSTM</td>
<td>78.38</td>
<td>66.36</td>
<td>69.12</td>
<td>63.24</td>
</tr>
<tr>
<td>PBAN</td>
<td>78.62</td>
<td>67.45</td>
<td>71.98</td>
<td>66.91</td>
</tr>
<tr>
<td>PRET+MULT</td>
<td>78.73</td>
<td>68.63</td>
<td>71.91</td>
<td>68.79</td>
</tr>
<tr>
<td>TransCap</td>
<td>79.55</td>
<td>71.41</td>
<td>73.87</td>
<td>70.10</td>
</tr>
<tr>
<td>ATN</td>
<td>82.36</td>
<td>74.00</td>
<td>76.48</td>
<td>72.60</td>
</tr>
<tr>
<td>DGEDT</td>
<td><strong>83.90</strong></td>
<td>75.10</td>
<td>76.80</td>
<td>72.30</td>
</tr>
<tr>
<td>R-GAT</td>
<td>83.30</td>
<td><strong>76.08</strong></td>
<td>77.42</td>
<td>73.76</td>
</tr>
<tr>
<td>HITN(ours)</td>
<td>82.59</td>
<td>74.21</td>
<td><strong>78.50</strong></td>
<td><strong>74.92</strong></td>
</tr>
</tbody>
</table>

Table 2: Main experiment results.

References


Conference on Artificial Intelligence.


Yukun Ma, Haiyung Peng, and Erik Cambria. 2018. Targeted aspect-based sentiment analysis via embedding commonsense knowledge into an attentive.


