# Enhanced Aspect Level Sentiment Classification with Auxiliary Memory

**Peisong Zhu,** **Tieyun Qian** *
School of Computer Science, Wuhan University, China
{zhups24, qty}@whu.edu.cn
*Contact author

## Abstract

In aspect level sentiment classification, there are two common tasks: to identify the sentiment of an aspect (category) or a term. As specific instances of aspects, terms explicitly occur in sentences. It is beneficial for models to focus on nearby context words. In contrast, as high level semantic concepts of terms, aspects usually have more generalizable representations. However, conventional methods cannot utilize the information of aspects and terms at the same time, because few datasets are annotated with both aspects and terms. In this paper, we propose a novel deep memory network with auxiliary memory to address this problem. In our model, a main memory is used to capture the important context words for sentiment classification. In addition, we build an auxiliary memory to implicitly convert aspects and terms to each other, and feed both of them to the main memory. With the interaction between two memories, the features of aspects and terms can be learnt simultaneously. We compare our model with the state-of-the-art methods on four datasets from different domains. The experimental results demonstrate the effectiveness of our model.

## 1 Introduction

Sentiment analysis (Pang and Lee, 2008; Liu, 2012) is a fundamental task in the field of natural language processing. As one of the subtasks of sentiment analysis, the goal of aspect level sentiment classification is to infer the sentiment polarity (e.g. positive, negative, neutral) of the aspect (e.g. food, service). Aspect level sentiment classification is arousing more and more researchers' attention, as it can provide more all-round and deeper analysis than document or sentence level sentiment classification.

There are two common scenes in aspect level sentiment classification. The first one aims to identify the sentiment of a predefined aspect, we call it ASC (aspect sentiment classification) for short. The other aims to identify the sentiment of an explicit term which occurs in the sentence, we call it TSC (term sentiment classification). We take the review "*A mix of students and area residents crowd into this narrow, barely there space for its quick, tasty treats at dirt-cheap prices.*" as an example. For ASC task, we need to infer the sentiment polarities for a set of predefined aspects including "*food*", "*price*", and "*ambience*". Here, the sentiment polarities for "*food*" and "*price*" are positive while that for "*ambience*" is negative. For TSC task, we are given three terms "*space*", "*treats*", "*prices*" in the review and need to infer the sentiment polarities for these terms. Here, the sentiment polarities for terms "*treats*" and "*prices*" are positive while that for "*space*" is negative.

In most cases, aspect level sentiment classification works in a supervised manner. Researchers usually extract features and use machine learning algorithms to train the sentiment classifier. Typical methods are usually based on the manually extracted features. Such methods need either laborious feature engineering works or massive linguistic resources. In recent years, neural networks have achieved significant performance in various NLP tasks like question answering (Sukhbaatar et al., 2015), machine translation (Bahdanau et al., 2014), text summarization (Rush et al., 2015), and text classification (Kim, 2014; Kalchbrenner et al., 2014; Yang et al., 2016). The key advantage of these methods is that they can automatically learn text features or representations from data. There are several pioneering studies introducing neural networks into the field of sentiment classification, including long short term memory (Tang et al., 2016a; Wang et al., 2016) and deep memory network (Tang et al., 2016b; Tay et al., 2017). The main intuition behind these studies is to capture the important context words related to the given aspects.

The terms and aspects in ASC or TSC are closely related to each other. For example, the term "*sandwich*" is surely about the aspect "*food*". However, existing neural network based methods fail to utilize the relevance between the aspects and the terms as very few datasets are annotated with both aspects and terms. On one hand, the number of terms is extremely large and they normally exhibit a long-tailed distribution. For example, in the review "*Not only did they have amazing, sandwiches, soup, pizza etc, but their homemade sorbets are out of this world!*", "*sandwiches*", "*soup*", "*pizza*", and "*sorbets*" are different words though they are all about the aspect of "food", and many of them rarely occur in the corpus. It is hard for neural networks to train high quality embeddings for the terms with low frequency. On the other hand, as the high level semantic concepts of terms, the number of aspects is small and the aspects are usually predefined. This ensures that the embeddings of aspects are of high quality and generalization ability. However, aspects do not explicitly occur in sentences, thus it is hard for a model to capture aspect related context words from the sentence.

In this paper, we propose a novel deep memory network with an auxiliary memory (**DAuM**) to solve the above problems. Specifically, we first adopts a basic memory to capture the important part of context words for sentiment classification. We then present an auxiliary memory to make the connections between aspects and terms. Due to the lack of supervised information, we cannot get term words for each sentence in ASC task, or get aspect words for each sentence in TSC task. Following the idea of (Lau et al., 2017) and (He et al., 2017), we implicitly generate aspects for terms or extract terms for aspects via the auxiliary memory. Finally, the original and generated aspects/terms are fed into the main memory to capture the sentiment words related to the aspects/terms. With the interaction between two memories, the output vectors will combine features of both aspects and terms. We compare our model with the state-of-the-art methods on four datasets. The experimental results show that our model performs well for different domains of data, and consistently outperforms all baselines.

## 2 Related Work

Aspect level sentiment classification is a challenging task which aims to identify the sentiment polarity of an aspect or a term in the sentence. Recent years have witnessed the boom of deep learning methods in in aspect level sentiment classification tasks. Tang et al. (2016a) proposed target-dependent LSTM to capture the aspect information when modeling sentences. A forward LSTM and a backward LSTM towards term words are used to capture the information before and after the aspect term. Obviously, this method is not suitable to ASC task as aspect words do not always explicitly occur in sentences.

Attention mechanism (Bahdanau et al., 2014) is an effective mechanism which enforces the model to focus on the important part of context words by computing a weight distribution for context words. Wang et al. (2016) proposed an attention-based LSTM method for the ASC task by concentrating on different parts of a sentence to different aspects. In many cases, given terms have multiple words which makes it difficult to build the semantic representations. Ma et al. (2017) proposed an interactive attention network to address this problem.

Tang et al. (2016b) introduced an end-to-end memory network for aspect level sentiment classification, which employs an attention mechanism over an external memory to capture the importance of each context word. Such importance degree and text representation are calculated with multiple computational layers, each of which is a neural attention model over an external memory. Tay et al. (2017) designed a dyadic memory network that models dyadic interactions between aspect and context with neural tensor compositions or holographic compositions for memory selection operation.

We distinguish our work with the memory networks (Tang et al., 2016b; Tay et al., 2017) in that our goal in to capture the relatedness between term and aspect to improve the sentiment classification performance and we design an auxiliary memory to this end. In contrast, existing studies only focus on the relatedness between context words the aspect/term using a sentiment memory, which is a single component in our model. Our study is inspired by the recent advances in neural networks, including the unsupervised aspect extraction method in (He et al., 2017), the deep multi-task learning framework in (Li and Lam, 2017) for aspect and opinion extraction tasks, the end-to-end deep memory network for attitude identification and polarity classification task in (Li et al., 2017), and the neural attention networks for stance classification task in (Du et al., 2017).

## 3 Deep Memory Network with Auxiliary Memory

In this section, we describe our proposed **DAuM** model for aspect level sentiment classification. We mainly introduce the model for TSC task, which has similar architecture for ASC task. We first give the overview of the model and then describe the single layer case. Finally, we introduce how to extend model to stack multiple layers.

### 3.1 An Overview

An illustration of 3-layer model for TSC is given in Figure 1.

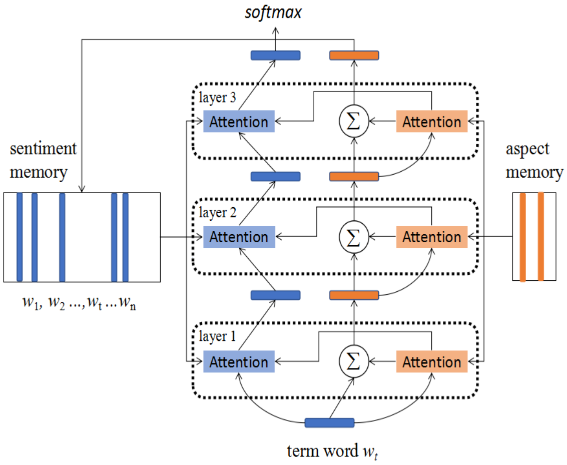


Figure 1: Architecture of **DAuM**

The left part in blue is the main memory for sentiment classification, and we call it sentiment memory. The right part in red is the auxiliary memory. In TSC, auxiliary memory takes a term as input and generates corresponding aspect representation. Therefore we call it aspect memory. Both the original term representation and generated aspect representation, which incorporate term and aspect information at the same time, are fed into sentiment memory to capture the important part of context words. The middle part consists in multiple computational layers. The output vector of a layer is taken as the input of the next layer.

### 3.2 Single Layer

**Input Embedding**: To apply deep-learning method to text data, we first map each word into a low-dimensional continuous vector, which is also known as word embedding (Mikolov et al., 2013; Pennington et al., 2014). Specifically, let $\mathbb{L} \in \mathbb{R}^{d \times |V|}$ be an embedding matrix made up of all the word embeddings, where $d$ is the dimensionality of word embedding and $|V|$ is vocabulary size. Given a sentence $s = \{w_1, w_2, ..., w_n\}$ consisting $n$ words, we first use a lookup layer to get the embedding $x_i \in \mathbb{R}^d$ of word $w_i$, which is a column in the embedding matrix $\mathbb{L}$.

Since terms occur in sentences and a term may contain one single word or multiple words as a phrase, we use word embedding matrix $\mathbb{L}$ to get the representations for terms. In case that a term contains only one word, the term representation will be the embedding of the word. For a phrase-form term, its representation is the average of all constituting word embeddings. We use $v_t \in \mathbb{R}^d$ to denote the term representation.

**Aspect Memory**: In order to implicitly infer the corresponding aspect for a given term, we define an auxiliary memory to stack the aspect information. The memory takes a term as input, and generates an output representation, which is combined with aspect information. Formally, we define the aspect memory as $M_a \in \mathbb{R}^{k \times d}$ , where $k$ is the memory size or the number of defined aspects, and $m_i$ represents the *i*-th piece of the memory.

Taking the term embedding $v_t \in \mathbb{R}^d$ as input, the aspect memory generates a continuous output vector $o_a \in \mathbb{R}^d$. The vector $o_a$ is computed as a weighted sum of each piece of memory $m_i$ via attention mechanism.

$$o_a = \sum_{i=1}^{k} m_i \alpha_i, \tag{1}$$

where $\alpha_i$ is the weight for a piece of memory $m_i$. The weight $\alpha_i$ is computed based on the semantic relatedness between the term and each aspect.

$$u_i = m_i \cdot W_a \cdot v_t \tag{2}$$

$$\alpha_i = \frac{exp(u_i)}{\sum_{j=1}^{k} exp(u_j)} \tag{3}$$

Eq.2 is a bilinear function used to compute the semantic relatedness. The output of the bilinear function is then fed into a softmax function to generate the final weight $\alpha_i$ in Eq.3.

**Sentiment Memory**: The goal of TSC is to identify the sentimental polarity of given terms. To this end, we build another external memory, called sentiment memory, to capture the important context words which may be helpful to sentiment classification. Different from the memory network in (Tang et al., 2016b), the sentiment memory in our model takes both term representation and generated aspect representation as the input. Hence our input contains richer information for sentiment classification than that in (Tang et al., 2016b).

We build sentiment memory $M_s \in \mathbb{R}^{n \times d}$ via word embeddings of a sentence , where $n$ is the memory size or the length of the sentence. To differentiate a piece of sentiment memory from that of aspect memory, we use $x_i$ to represent the *i*-th piece in sentiment memory, which is indeed the embedding of the *i*-th word. Similar to aspect memory, the output vector of sentiment memory is also computed as a weighted sum of each piece of memory $x_i$ via attention mechanism. When the term representation $v_t \in \mathbb{R}^d$ is fed into sentiment memory, a feed forward neural network is applied to computing the semantic relatedness between the term and each memory piece.

$$\beta_i^t = softmax(\tanh(W_s^t[x_i; v_t] + b_s^t)), \tag{4}$$

where $W_s^t \in \mathbb{R}^{1 \times 2d}$ and $b_s^t \in \mathbb{R}^{1 \times 1}$ are transformation parameters. To make full use of the information of the term and implicit aspect, sentiment memory further takes the generated aspect representation $o_a$ as input, and generates a weight distribution via attention mechanism.

$$\beta_i^a = softmax(\tanh(W_s^a[x_i; o_a] + b_s^a)), \tag{5}$$

where $W_s^a \in \mathbb{R}^{1 \times 2d}$ and $b_s^a \in \mathbb{R}^{1 \times 1}$ are transformation parameters.

The final weight and output of sentiment memory is computed based on term level weight $\beta^t$ and aspect level weight $\beta^a$. Formally,

$$\beta = (1 - \lambda)\beta^t + \lambda\beta^a, \tag{6}$$

$$o_s = \sum_{i=1}^{n} x_i \beta_i, \tag{7}$$

where $\beta$ is the final weight combining both term and implicit aspect information, $0 < \lambda < 1$ is the hyperparameter controlling the importance of the aspect or the term, and $o_s \in \mathbb{R}^d$ is the output vector of sentiment memory.

### 3.3 Multiple Layers

Previous studies (LeCun et al., 2015) show that multiple processing layers can learn higher level representations. Hence we extend our model to stack multiple layers. In the first computational layer, we take the term vector as the input of aspect memory to generate corresponding aspect vector based on each memory piece in aspect memory through an attention layer. Both the original term vector and the generated aspect vector are then fed into the first layer of sentiment memory to capture the important part of context words through another attention layer. The model is then stacked as follows. In aspect memory, the output of the attention layer and the aspect vector in $N_{th}$ layer are summed as the input of *N+1*$_{th}$ layer. Then the newly updated aspect vector in *N+1*$_{th}$ layer of aspect memory and the output of $N_{th}$ attention layer of sentiment memory are taken as the input of *N+1*$_{th}$ layer of sentiment memory.

### 3.4 Output and Model Training

After the computation in multiple layers, we get the final output vector $o_s^f$ of sentiment memory and the final output vector $o_a^f$ of aspect memory. Here $o_s^f$ is the sentiment feature, which takes both term and aspect information into consideration. $o_a^f$ is the high level abstraction of a given term. We apply them to different tasks to train the model.

**Term Sentiment Classification**: Given a term $w_t$, a sentence $s$, and the sentiment category set $C$, the TSC task aims to identify the sentiment polarity of the term in its context. In our model, we regard it as a probability problem, i.e., estimating the conditional probability of $w_t$ in $s$ to a category $c \in C$. We take $o_s^f$ as the final sentiment feature of a sentence, and use a linear layer to transform $o_s^f$ into a real-valued vector which has the same length as the category number $|C|$. We then apply a softmax layer to calculating the conditional probability distribution.

$$P_c(s, w_t) = softmax(W_c o_s^f + b_c), \tag{8}$$

where $(s, w_t)$ denotes a sentence-term pair, and $P_c(s, w_t)$ is the probability of predicting category $c$ given the sentence $s$ and the term $w_t$. The model is trained in an end-to-end supervised way by minimizing the cross entropy between predicted and real probability distribution. The loss function is written as:

$$L_{cla} = -\sum_{(s,w_t)\in T}\sum_{c\in C} y_c(s, w_t) \cdot logP_c(s, w_t) \tag{9}$$

where $T$ is the set of training data, $C$ is the collection of sentiment categories, and $y_c(s, w_t)$ is the ground truth for $(s, w_t)$ pair.

**Term Prediction**: $o_a^f$ is the high level aspect representation of a given term. Inspired by (Lau et al., 2017), we believe that the aspect representation generated via a term should be able to predict the term in turn. In other words, the term embedding is more similar to the aspect representation than the embeddings of other words in the sentence. We include this into our objective and define a hinge loss function which maximizes the semantic relatedness between $o_a^f$ and the term while simultaneously minimizing the semantic relatedness between $o_a^f$ and other words in the sentence.

$$L_{pre} = \sum_{(s,w_t)\in T}\sum_{i\in\{1\ldots n\}/\{t\}} max(0, 1 - o_a^f \cdot W_p \cdot v_t + o_a^f \cdot W_p \cdot x_i) \tag{10}$$

**Aspect Regularization**: We aim to learn the aspect embeddings which can represent different aspects in real-life data. However, the aspect embeddings may suffer from the redundancy problem (He et al., 2017) during the training process. We add a regularization term to ensure the diversity of aspect embeddings.

$$L_{reg} = \|M_a M_a^T - I\|, \tag{11}$$

where $I$ is the identity matrix with the non-diagonal element $a_{ij}$ $(i \neq j)$. $M_a M_a^T$ corresponds to the dot product of two different aspect embeddings, which reflects the relevance between two aspects. $L_{reg}$ reaches its minimum value when the dot product between any two different aspect embeddings is zero. Thus the regularization term encourages orthogonality among the rows of the aspect embedding matrix and penalizes redundant representation. Our final loss function is the linear combination of $L_{cla}$, $L_{pre}$, $L_{reg}$:

$$L_{final} = L_{cla} + \gamma_1 L_{pre} + \gamma_2 L_{reg}, \tag{12}$$

where $\gamma_1$, $\gamma_2$ are hyperparameters that control the weight of different parts.

### 3.5 Applied to ASC

The model for ASC task has similar architecture as the model introduced above. With a certain number of aspects, we don't need hyperparameter $k$ for aspect embedding. In addition, auxiliary memory here is used to stack implicit term information, and is initialized with embeddings of words in the sentence. The auxiliary memory takes an aspect as input and generates corresponding term representation with the same method.

## 4 Experiment

We conduct both the TSC and ASC experiments to check whether our proposed model improve the performance of aspect-level sentiment classification.

### 4.1 Settings

#### 4.1.1 Dataset

we conduct experiments on four datasets for ASC task and TSC task. In case that the same dataset is used for both tasks, we use a suffix (ASC)/(TSC) to distinguish them.

- **Restaurants(ASC)** is a dataset obtained from SemEval 2014 (Pontiki et al., 2014). This dataset contains customer reviews from the restaurant domain. Each review is annotated a sentiment polarity towards a given aspect. Sentiment polarity set includes "*Positive*", "*Negative*" and "*Neutral*". Predefined aspect set includes "*food*", "*price*", "*service*", "*ambience*" and "*anecdotes/miscellaneous*".
- **Restaurants(TSC)** is the same dataset as Restaurants(ASC) except that terms are given in Restaurants(TSC) while aspects are not annotated.
- **Laptops** is a dataset obtained from SemEval 2014. This dataset contains customer reviews pertaining to the computers and laptops domain. The sentiment polarity set is the same as that in Restaurants(TSC). As terms are given in Laptops, they are used for TSC task.
- **TweetNews** is a dataset obtained from Semeval 2016 Task 6 (Mohammad et al., 2016). It contains English tweets which are annotated for the stance towards one of five topics "*Atheis*", "*Climate Change is a Real Concern*", "*Feminist Movement*", "*Hillary Clinton*", and "*Legalization of Abortio*". The stance set includes "*Favor*", "*Against*" and "*None*" (We treat them as "*Positive*", "*Negative*" and "*Neutral*" for simplification). This dataset is used for ASC task.

We use the official training/testing split in our experiments. To prevent the model from overfitting, we randomly sample one-sixth samples from the training set as the development set. The statistics of four datasets are showed in Table 1.

Table 1: Statistics of four datasets.

| Dataset | Set | Total | Pos. | Neg. | Neu. |
|---|---|---|---|---|---|
| Restaurants(ASC) | Train | 2927 | 1806 | 697 | 424 |
| | Test | 973 | 657 | 222 | 94 |
| | Dev | 591 | 372 | 142 | 77 |
| Restaurants(TSC) | Train | 3017 | 1806 | 669 | 542 |
| | Test | 1120 | 728 | 196 | 196 |
| | Dev | 591 | 358 | 138 | 95 |
| Laptops | Train | 1934 | 823 | 730 | 381 |
| | Test | 638 | 341 | 128 | 169 |
| | Dev | 394 | 171 | 140 | 63 |
| TweetNews | Train | 2416 | 627 | 1163 | 626 |
| | Test | 1249 | 303 | 716 | 230 |
| | Dev | 498 | 125 | 233 | 140 |

#### 4.1.2 Metrics

In aspect level sentiment classification task, each dataset has multiple sentiment polarities, and each sample is classified into one of polarities. So we adopt *Accuracy*, *Precision*, *Recall*, as the evaluation metrics for multi-class classification. We also adopt the macro-averaged *F-score* to show a more thorough result following previous studies (Tay et al., 2017; Tang et al., 2016a).

### 4.1.3 Experimental Setting

In our experiments, we use 300-dimension word vectors pre-trained by GloVe (Pennington et al., 2014). The dimensionalities of word embedding, aspect embedding are all set to 300, which are same as those in the baselines AE-LSTM and ATAE-LSTM (Wang et al., 2016). $\lambda$, $\gamma_1$, $\gamma_2$ are set to 0.4, 1.0, and 0.5, respectively. We randomize other parameters with uniform distribution $U(-0.01, 0.01)$. We train the model with Stochastic Gradient Descent optimization algorithm and set learning rate as 0.01. We use grid search to get hyper-parameters settings for our method.

## 4.2 Baselines

In our experiments, we compare our model with the following baseline methods on each dataset.

- **ContextAVG** is a simple baseline which averages the vectors of all context words as the representation of a sentence. The result is then concatenated with the aspect vector and finally fed into softmax function.
- **LSTM** is an implementation of standard LSTM where aspect information cannot be captured in this model.
- **TD-LSTM** (Tang et al., 2016a) takes aspect information into consideration. It uses a forward LSTM and a backward LSTM towards term words to capture the information before and after the term. We run it for only TSC task, as aspect words do not explicitly occur in sentences in ASC task.
- **AE-LSTM** (Wang et al., 2016) takes into account aspect information by embedding aspects into another vector space. The embedding vectors of aspects are learnt during the process of training.
- **ATAE-LSTM** (Wang et al., 2016) is an extension of AE-LSTM. It uses attention mechanism to capture the most important information in response to a given aspect. In addition, ATAE-LSTM can capture the important and different parts of a sentence when different aspects are provides.
- **TAN** (Du et al., 2017) is a model proposed for stance classification which is similar to ASC task. Here we use it as one of baselines to evaluate the effectiveness our model on the task and dataset of other domain.
- **MemNN** (Tang et al., 2016b) is an end-to-end deep memory network which employs an attention mechanism with an external memory to capture the importance of each context word. Such importance degree and text representation are calculated with multiple computational layers.

We re-implemented all baseline methods to make their results as similar as possible to those in the original papers. Each competitor was optimized independently. For the parameters like dimensions, learning rate, and optimization method, we follow their settings in the original paper of the baselines.

## 4.3 Main Results

In this section, we discuss the experimental results on both ASC task and TSC task. Table 2 reports the results for ASC task on Restaurants (ASC) and TweetNews dataset. Table 3 reports the results for TSC task on Restaurants (TSC) and Laptops dataset.

Table 2: Experimental results for TSC

| | Restaurants(TSC) | | | | Laptops | | | |
|---|---|---|---|---|---|---|---|---|
| Method | Accuracy | Precision | Recall | F-score | Accuracy | Precision | Recall | F-score |
| ContextAVG | 75.09 | 68.93 | 62.91 | 63.96 | 67.24 | 65.56 | 60.96 | 61.86 |
| LSTM | 74.28 | 68.72 | 61.89 | 62.21 | 66.46 | 65.04 | 60.54 | 61.72 |
| TD-LSTM | 75.63 | 69.18 | 63.05 | 64.16 | 68.18 | 66.86 | 61.15 | 62.28 |
| AE-LSTM | 76.25 | 69.76 | 63.21 | 64.32 | 68.97 | 67.12 | 61.32 | 62.50 |
| ATAE-LSTM | 77.23 | 70.83 | 63.95 | 64.95 | 68.65 | 66.98 | 61.18 | 62.45 |
| MemNN | 80.09 | 72.10 | 65.68 | 67.82 | 72.21 | 70.82 | 65.03 | 66.75 |
| DAuM | **82.32** | **74.68** | **70.18** | **71.45** | **74.45** | **72.96** | **69.21** | **70.16** |

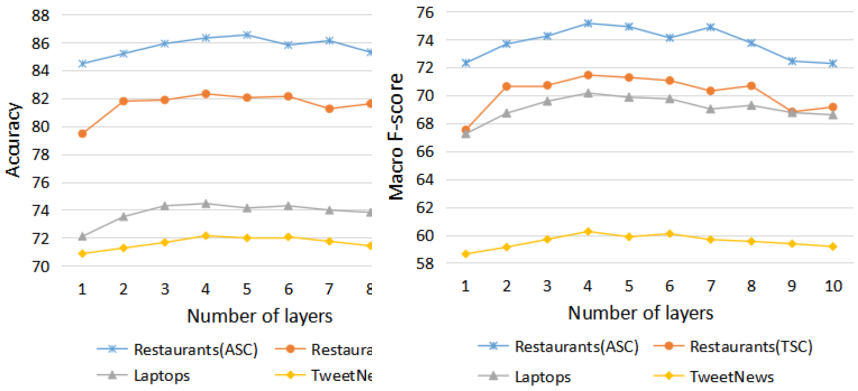
Table 3: Experimental results for ASC

| | Restaurants(ASC) | | | | TweetNews | | | |
|---|---|---|---|---|---|---|---|---|
| Method | Accuracy | Precision | Recall | F-score | Accuracy | Precision | Recall | F-score |
| ContextAVG | 80.37 | 72.86 | 67.58 | 68.72 | 65.25 | 57.52 | 51.96 | 52.88 |
| LSTM | 82.01 | 74.20 | 69.16 | 70.20 | 66.86 | 58.76 | 53.12 | 54.32 |
| AE-LSTM | 82.53 | 74.56 | 69.36 | 70.48 | 69.42 | 60.96 | 55.18 | 56.28 |
| ATAE-LSTM | 83.98 | 75.92 | 70.88 | 71.76 | 69.58 | 61.10 | 55.45 | 56.72 |
| TAN | 82.53 | 74.48 | 69.50 | 70.55 | 68.78 | 60.42 | 55.06 | 56.25 |
| MemNN | 84.28 | 76.06 | 70.24 | 72.38 | 70.14 | 62.21 | 57.05 | 58.62 |
| DAuM | **86.33** | **77.54** | **73.82** | **75.16** | **72.14** | **64.52** | **58.96** | **60.24** |

As shown in Table 2 and Table 3, our method consistently outperforms all baselines on four datasets for both ASC task and TSC task. ContextAVG performs poorly. Since averaging context pay equivalent attention to all words, it is unable to capture the sentiment words which are particularly important to sentiment classification. LSTM does not perform well either. The reason may be that it ignores aspect information which plays key role in aspect level sentiment classification. ATAE-LSTM and MemNN perform relatively better as they not only utilize aspect information but also capture important words related to aspect via an attention mechanism. Our model DAuM outperforms ATAE-LSTM and MemNN, as implicit aspect/term representations are generated in our model, which provide more abundant features for sentiment classification. In addition, our model performs better than TAN on TweetNews dataset. This further proves that our model is suitable to some tasks from other domain like stance classification.

### 4.4 Effects of Multiple Layers

The number of multiple layers has an important effect to the performance of our model. We evaluate our model with 1 to 10 layers with $k = 5$. The results are shown in Figure 2. In general, multiple layers can help compute high level representation and improve the performance. Our model with 4 or 5 layers already works pretty well. However, the performance does not always enhance with the increasing number of layers. It is because too many layers make the training of the model difficult. In addition, the model becomes less generalizable with more parameters introduced.



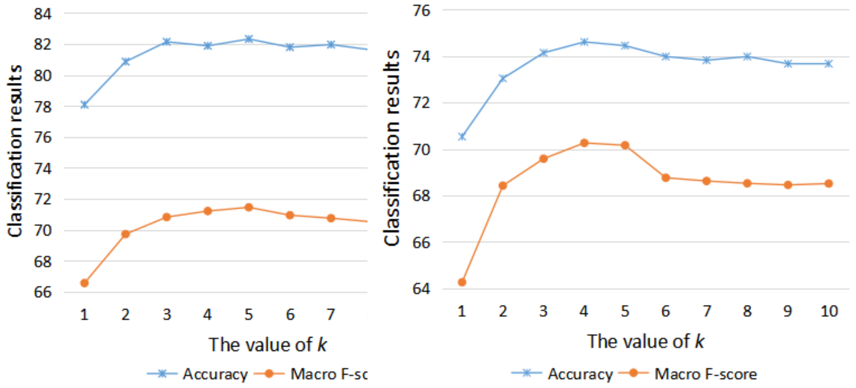
(a) Accuracy

(b) Macro F-score

Figure 2: Effects of multiple layers

### 4.5 Effects of Aspect Number

The aspect number $k$ is a main parameter in our model. To evaluate the effects of aspect number, we fix the number of layers to 4 and change $k$ from 1 to 10. The results are shown in Figure 3. In general, our model performs well in Restaurants(TSC) and Laptops when $k = 5$ and $k = 4$ respectively. When $k < 3$, the result is not as good as when $k$ is large in both datasets. The reason may be that terms are converted into similar aspect representations when $k$ is small. This leads to similar predicted results even



(a) Results on Restaurants(TSC)

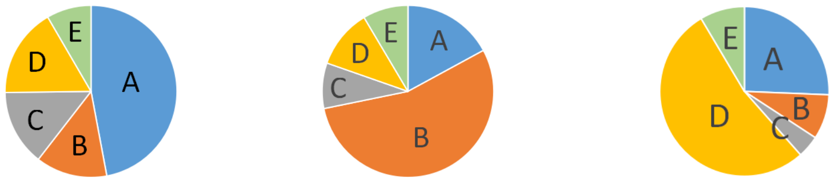
(b) Results on Laptops

Figure 3: Effects of aspect number $k$

though different terms are given. It proves that, a limited number of aspects is helpful to our tasks as they have more generalizable representations, but if the number of aspects is too small, it will lose the capability of representing the semantics of different aspects in real-life datasets.

## 4.6 Case Study

In our model, auxiliary memory takes term/aspect as input and generates corresponding aspect/term representation. To make it more intuitive, we visualize the aspect attention when different terms are fed into aspect memory. The experiment is conducted on Restaurants(TSC) with $k = 5$. “A”, “B”, “C”, “D” and “E” represent the five aspects. Three terms “*pasta*”, “*waitress*”, “*price*”, which belong to aspect “*food*”, “*service*”, “*price*” respectively, are taken as an example. The pie charts are drawn based on the attention weight for each aspect. The results are shown in Figure 4.



(a) Given “*pasta*”

(b) Given “*waitress*”

(c) Given “*price*”

Figure 4: Aspect attention weight for different terms

As can be seen, given a term“*pasta*”, the aspect memory generates an attention distribution assigning the largest weight to aspect “A”. Similarly, the generated attention distribution gives the largest weight to aspect “B” and “D” respectively when other two terms ‘*waitress*” and “*price*” are fed into aspect memory. To a certain degree, these three cases prove that the aspect memory can effectively compute the semantic relatedness between the given term and each aspect, and consequently generate the most related high level representation.

# 5 Conclusion

In this paper, we propose a novel deep memory network with auxiliary memory to handle with aspect level sentiment classification tasks. To address the problem that aspect and term information cannot be captured at the same time, we utilize an auxiliary memory to generate aspect or term representation implicitly. The original and generated aspects/terms are all fed into the main memory to capture sentiment words related the aspects/terms. With the interaction between two memories, aspects provide more generalizable representations for terms. And vise versa, terms provide clues for aspects to focus on nearby context words. We demonstrated the effectiveness of our model on four datasets. The results show that it can significantly outperform the state-of-the-art methods. Besides the improvements of classification performance, our method has a key advantage that it does not need extra annotated aspects or terms. Furthermore, our model can be easily extended to solve other related problems in this domain, or modified with other advanced basic models.

## Acknowledgments

The work described in this paper has been supported in part by the NSFC projects (61572376, 91646206), and the 111 project(B07037).

## References

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

Jiachen Du, Ruifeng Xu, Yulan He, and Lin Gui. 2017. Stance classification with target-specific neural attention. In *Twenty-Sixth International Joint Conference on Artificial Intelligence*, pages 3988–3994.

Ruidan He, Wee Sun Lee, Hwee Tou Ng, and Daniel Dahlmeier. 2017. An unsupervised neural attention model for aspect extraction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pages 388–397.

Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. 2014. A convolutional neural network for modelling sentences. *arXiv preprint arXiv:1404.2188*.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*.

Jey Han Lau, Timothy Baldwin, and Trevor Cohn. 2017. Topically driven neural language model. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pages 355–365.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *Nature*, 521(7553):436.

Xin Li and Wai Lam. 2017. Deep multi-task learning for aspect term extraction with memory interaction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2886–2892.

Cheng Li, Xiaoxiao Guo, and Qiaozhu Mei. 2017. Deep memory networks for attitude identification. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, pages 671–680. ACM.

Bing Liu. 2012. Sentiment analysis and opinion mining. In *Synthesis Lectures on Human Language Technologies*, pages 152–153.

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

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Saif Mohammad, Svetlana Kiritchenko, Parinaz Sobhani, Xiaodan Zhu, and Colin Cherry. 2016. Semeval-2016 task 6: Detecting stance in tweets. In *International Workshop on Semantic Evaluation*, pages 31–41.

Bo Pang and Lillian Lee. 2008. Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2(1):459–526.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014. Semeval-2014 task 4: Aspect based sentiment analysis. *Proceedings of International Workshop on Semantic Evaluation at*, pages 27–35.

Alexander M Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. *arXiv preprint arXiv:1509.00685*.

Sainbayar Sukhbaatar, Jason Weston, Rob Fergus, et al. 2015. End-to-end memory networks. In *Advances in neural information processing systems*, pages 2440–2448.

Duyu Tang, Bing Qin, Xiaocheng Feng, and Ting Liu. 2016a. Effective lstms for target-dependent sentiment classification. In *Proceedings of COLING 2016*, pages 3298–3307.

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

Yi Tay, Luu Anh Tuan, and Siu Cheung Hui. 2017. Dyadic memory networks for aspect-based sentiment analysis. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 107–116. ACM.

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

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1480–1489.