2018 IEEE International Conference on Big Data (Big Data)

Lifelong Learning Memory Networks for Aspect Sentiment Classification

Shuai Wang[†], Guangyi Lv[‡], Sahisnu Mazumder[†], Geli Fei[†], and Bing Liu[†]
[†] *University of Illinois at Chicago, Chicago, USA*
[‡] *University of Science and Technology of China, Hefei, China*
shuaiwanghk@gmail.com, gylv@mail.ustc.edu.cn, sahisnumazumder@gmail.com, gfei2@uic.edu, liub@cs.uic.edu

***Abstract*—Aspect sentiment classification (ASC) is a fundamental task in sentiment analysis. It aims at classifying the sentiment expressed on some target aspects/features of entities (e.g., products and services). Although a great deal of research has been done, this task remains to be very challenging. Recently, memory networks, a type of neural model, have been used for this task and have achieved state-of-the-art results. However, such neural models usually require a large amount of well-annotated training data for producing reasonably good results. Unfortunately, for the ASC task, the human-annotated data with aspect-level labels are scarce and costly to obtain. In this work, we aim to use big unlabeled data to help. The key idea is to make a memory network learn knowledge from the big unlabeled data (treated as past tasks) and use the learned knowledge to better guide its future task learning. To achieve this goal, we propose a novel lifelong learning approach that can automatically meta-mine knowledge from multiple past domains. In addition, a new model named lifelong learning memory network (L2MN) is proposed to incorporate the mined knowledge into its learning process, where two types of knowledge are involved, namely, aspect-sentiment attention and context-sentiment effect. Extensive experimental results using real-world review datasets demonstrate the effectiveness of our approach.**

***Index Terms*—Lifelong Learning, Aspect Sentiment Classification, Sentiment Analysis, Memory Network, Neural Network**

I. Introduction

Aspect sentiment classification (ASC), also known as aspect-based sentiment classification, is a fundamental task in sentiment analysis [1]. Given a sentence and an aspect discussed in the sentence, it aims to identify the sentiment polarity on the aspect (i.e., aspect sentiment). More specifically, it is to determine whether the sentence conveys a positive, negative or neutral aspect sentiment. For instance, in the sentence "*clear voice but the screen is scratched*", the sentiment polarity on aspect *voice* is positive while the one on aspect *screen* is negative. Note that aspects are also referred to as opinion targets (or *targets*) in the literature, which are usually product features/attributes. We thus use term *aspect* and *target* interchangeably in this article. In practice, aspects are either given by the user or automatically extracted using aspect extraction techniques [1]. In this work, we assume the aspects are given and focus only on the classification problem [2]–[4].

To address ASC, there are two main approaches, namely, lexicon-based and supervised learning. We will discuss related works in section II. This work lies in the supervised learning direction, which is data-driven and domain-specific. Specifically, a machine learning (ML) based classifier will be trained to capture sentiment features towards aspects, with aspect-based sentiment (or *aspect-sentiment*) labels provided. Examples are shown in Fig. I. However, unlike document-level or sentence-level classification, which is to estimate an overall sentiment polarity for an entire document/review or a single sentence, building an ML-based classifier for aspect-level sentiment analysis is somewhat tricky, because a classifier needs to consider and encode aspect information. This requirement is very important. Recall the aforementioned review sentence, where different sentiments would be inferred towards different aspects, i.e., positive on aspect *voice* but negative on aspect *screen*. Failure to encode such aspect information will be problematic for ASC. To involve aspect information, earlier studies relied on carefully engineered features [5]–[7], which require pattern designs, feature templates, or external resources.

Memory network [8], [9], a neural ML model, has recently become a better alternative for the ASC task. One key reason is that it can eliminate the sophisticated feature engineering, and meanwhile, achieve state-of-the-art results [3], [10]. Its key advantages to ASC are its ability to learn aspect and context representation (in an embedding manner) and its attention mechanism [10], [11]. Let us use the same example to explain. When *voice* is the target aspect and represented in the embedding space, the context word "clear" will be assigned a higher attention weight than the word "scratched", under its attention mechanism. In contrast, when *screen* is the target aspect, more attention will be put on "scratched" instead of "clear". Next, the aspect-oriented sentiment can be inferred based on the weighted sum of the sentiment effect from its context words in the sentence (or called its *contexts*).

In spite of the suitability of the memory network for ASC, we observed that in practice, two crucial issues hinder its performance. First, the attention is sometimes wrongly placed. For example, a model fails to identify that "scratched" is an important context word for aspect *screen* and thus gives it no or small attention weight. Second, when the attention is correctly assigned (i.e., a high weight is given to a correct context word), the sentiment of that word could be learned in a misleading polarity direction. For instance, a model may learn that the context word "scratched" to aspect *screen* is important but mistakenly regards it as a positive sentiment word (while a scratched screen should be negative) so the final sentiment prediction would still be wrong. We will show more examples regarding these two issues from our experiments in section VI.

978-1-5386-5035-6/18/$31.00 ©2018 IEEE

These two issues are caused by the fact that ASC is a fine-grained analysis task and requires a large amount of aspect-sentiment labeled data, but such labeled data are often scarce. Its *data scarcity* problem can be found or explained from multiple perspectives: (1) In practice, aspect sentiment annotation is a labor-intensive and time-consuming task. Some example labeled data are shown in Fig. I, from which we see such annotation requires substantial human effort and is often difficult to scale up. (2) In reality, one may have limited or small training data at hand (associated with gold aspect-sentiment labels) for a particular domain, while performing the ASC task. Suppose that *smartwatch* is a newly-released product and there is almost no large-scale labeled data, but manufacturers still want to analyze public opinions in time with (available) limited customer reviews. (3) In a real-world domain corpus, we should note that many product aspects are only discussed/covered by a small portion of the entire data. That is, an aspect or its context could be mentioned only a few times in the given corpus, even if the corpus itself is relatively big and well-annotated. In this case, we still have the scarcity problem (at the aspect level). To sum up, from the above or other more perspectives, the statistically insufficient information can lead to the failure of capturing the correct attention or sentiment polarity of a word.

Given the above problem observation and analysis, this work aims at using big unlabeled data to help memory networks for ASC. The key idea is to make memory networks learn as humans do. We humans learn knowledge from our past experience and use the learned knowledge to guide our future learning. Likewise, we hope a memory network can accumulate aspect sentiment knowledge by itself from big (past) data and then use it to better guide its new/future task learning. Below, we exploit *lifelong machine learning* (LML, or *lifelong learning*) to realize this idea and propose a novel lifelong learning approach for the ASC task.

Here we first introduce the general concept of LML and then illustrate our specific solution for ASC. LML is a machine learning paradigm that enables an ML model to retain the past results as knowledge and utilize it to help future learning [12], [13]. In other words, a learner can continuously accumulate knowledge and use it to help a new task. With regard to ASC, we treat the classification task of each particular domain/product as a single learning task (we thus will use the term *domain* and *task* interchangeably). Specifically, at any point in time a learner has worked on N domains/tasks and is going to learn to perform the $(N+1)$th task (called new domain), it uses the **knowledge** obtained from the past N domains to help get a better classification result for the $(N+1)$th one. This idea is workable for ASC because although every domain is distinct, there is a considerable amount of aspect overlapping across domains. For example, many electronic products share the aspect *voice* and *screen*. If certain knowledge is properly accumulated from the past domains and incorporated into the new domain, the issues discussed above in memory networks can be alleviated. For instance, when a learner has learned from the past domains like *Cellphone* and *Camera* that “scratched” is an important context word for *screen*, it will be less likely to assign wrong attention for *screen* in a new domain like *Laptop*.

Clear voice but the screen is scratched.
Aspect: voice, Sentiment: Positive
Aspect: screen, Sentiment: Negative

The player has had problems since the first day.
Aspect: player, Sentiment: Negative

The battery lasts very long when playing music.
Aspect: battery, Sentiment: Positive
Aspect: music, Sentiment: Neutral

Fig. 1. Sample data with aspects and aspect-sentiment labels.

To be concrete, we propose a new three-step lifelong learning approach to ASC. First, we design an automated aspect sentiment annotation strategy so as to make use of big (unlabeled) data from multiple domains. We call them assisting or past domains. Second, we retain aspect-specific attention and sentiment information from the classification results of the assisting/past domains, which are treated as raw knowledge. Third, we carry out knowledge mining to generate reliable knowledge for the new/current domain. Two different types of knowledge are considered, namely, Aspect-Sentiment Attention (ASA) and Context-Sentiment Effect (CSE). In order to leverage the mined knowledge, we propose a novel memory network named Lifelong Learning Memory Network (L2MN).

In summary, this paper makes the following contributions:

1) It indicates and analyzes the issues caused by the data scarcity problem of ASC while using memory networks, i.e., learning incorrect attention and sentiment orientation of context words. To address them, it suggests incorporating reliable knowledge mined from big unlabeled data into the learning process of memory networks.
2) It proposes to use the lifelong learning paradigm to realize the above idea, which helps memory networks work better and more stably on the ASC task. To our knowledge, no previous attempt has been made.
3) It designs a three-step lifelong learning approach to ASC, which can automatically meta-mine two types of knowledge from multiple past domains, namely, Aspect-Sentiment Attention (ASA) and Context-Sentiment Effect (CSE), without human involvement.
4) It develops a novel model named lifelong learning memory network (L2MN) that can leverage the learned knowledge to new domains. Experimental results show its effectiveness on multiple real-world datasets.

II. Related Work

A. Sentiment Analysis

Aspect sentiment classification (ASC) is a fundamental task in sentiment analysis [1]. Different from document-level or sentence-level sentiment classification [14]–[16], ASC identifies sentiment polarity on a target aspect. The above studies [14]–[16] thus cannot be directly applied to ASC (they neither consider nor encode the aspect information). In the context of addressing ASC, there are two major types of approaches, the *lexicon-based* and the *supervised learning* approaches. Lexicon-based approaches use opinion lexicons and human-crafted rules [17], [18] to build a general classifier, while supervised learning approaches learn domain-specific classifiers and do not require opinion lexicons. Our work belongs to the latter. In regard to concrete supervised learning solutions (for ASC), early works mainly used pattern designs, feature templates, dependency relations, etc. [6], [7], [19], where manual feature engineering and external resources are required. Recently, some neural network approaches [2], [20] have been applied to ASC to eliminate the sophisticated engineering process. Memory network [9] is such a type of neural models which achieves state-of-the-art results. Our work is based on it to address its shortcoming and improve its performance on ASC.

B. Memory Network and Attention

A memory network [8], [9] includes an external memory and an attention mechanism, which can improve many application tasks like question answering and machine translation. Its key advantages are that it can learn representations with its large external memory and the modeling of attention [11]. It has been recently applied to ASC [3] as discussed. Another popular attention-based model is the attention-based LSTM/RNN [2]. These two state-of-the-art solutions will be included in our experiments. There are other related studies [4], [21]–[23] using memory networks or the attention mechanism but with different focuses. [21] considered learning an additional set of attention for aspect words, [22] suggested a recurrent attention mechanism, [23] differentiated attention from left and right context, and [4] provided solutions for target-sensitive sentiment. More details can be found in a survey article [10]. While their works are more about learning sophisticated attention or capturing additional signals from a single domain, they do not aim at solving the two fundamental issues caused by data scarcity as we do. Moreover, none of them considered the knowledge accumulation or lifelong learning for ASC.

C. Lifelong Machine Learning

Our work is also related to lifelong machine learning (LML) [12], [24], [25]. First, notice that LML distinguishes itself from other related paradigms like multitask learning and transfer learning. Multitask learning [26] optimizes the learning of multiple related tasks at the same time, but not in a continual/lifelong learning setting. Transfer learning [27] aims at using the information from a source domain to assist the learning of a target domain. It does not accumulate knowledge nor does it tackle multiple tasks continuously. Further discussions about their difference (also with other paradigms) can be found in a survey book [13]. Second, in terms of sentiment analysis, LML has been used to tackle aspect extraction [28], [29], opinion mining [30] and document-level sentiment classification [31]. However, their works are essentially different from ours as they are not concerned with sentiment classification at the aspect level (i.e., ASC). Their methods thus cannot solve our problem. In fact, ASC could be more challenging as a fine-grained analysis problem. To the best of our knowledge, we are the first to explore the lifelong learning of aspect-sentiment knowledge to help ASC.

III. ASC Memory Network (AMN/NLL)

In this section, we briefly describe how a basic end-to-end memory network works for the ASC task. The primary model design follows a previous study [3]. Notice that this basic model does not use the lifelong learning solution, but it can be easily integrated into our proposed lifelong learning memory network (L2MN, detailed later). So it can also be viewed as a non-lifelong learning (NLL) version of L2MN.

Input Representation and Attention: Given an aspect $a \in \mathbb{R}^V$, an embedding matrix $E \in \mathbb{R}^{d \times V}$ is used to convert it to a vector representation t ($t = Ea$), where V indicates the size of vocabulary and d is the embedding dimension. Similarly, each context word (each of the other non-aspect words in the sentence) $x_i \in \{x_1, x_2, ...x_n\}$ is also projected to the continuous space and stored in memory, denoted as m_i ($m_i = Ex_i$) $\in \{m_1, m_2, ...m_n\}$. Here n is the number of words in a sentence and i indicates the word position. Attention is acquired based on the input representations. Specifically, an attention score α_i for the context word x_i is computed as:

$$\alpha_i = \frac{\exp(e_i)}{\sum_{j=1}^{n}\exp(e_j)}, e_i = \tanh(W_{att}[m_i; t] + b_{att}) \quad (1)$$

where $W_{att} \in \mathbb{R}^{1\times 2d}$ is a weight matrix and $b_{att} \in \mathbb{R}^{1\times 1}$ is a bias term. In this way, attention $\alpha = \{\alpha_1, \alpha_2, ..\alpha_n\}$ is represented as a vector of probabilities, indicating the weight/importance of different context words towards an aspect.

Output Representation and Sentiment Score/Logit: Another embedding matrix C is used for each context word x_i to generate its individual continuous vector c_i ($c_i = Cx_i$) $\in \mathbb{R}^d, C \in \mathbb{R}^{d\times V}$. An output vector o is produced by summing over the transformed vectors, each of which is weighted by its attention α_i. An aspect-based sentiment score is then calculated:

$$s = W(o + t), o = \sum_i \alpha_i c_i \quad (2)$$

where $W \in \mathbb{R}^{K\times d}$ is a sentiment weight matrix. The sentiment score/logit is represented as a vector $s \in \mathbb{R}^K$, where K is the number of (sentiment) classes. The final sentiment probability y is produced with a softmax operation $y = softmax(s)$.

From Sentiment Logit to Context-Sentiment Logit: Note that s is the final aspect-oriented sentiment score/logit. If we drop t out from Eq. 2, we can factorize Eq. 2 as the weighted sentiment contribution of each context word with $\sum_i \alpha_i Wc_i$, where the contribution weight is determined by the importance of a context word to the aspect, i.e., α_i. As α_i is for assigning the attention weight of a context word, the sentiment effect of this word can be presented as Wc_i and we refer to it as *Context-Sentiment Effect (CSE)*. We also define the $\alpha_i Wc_i$ as the *Context-Sentiment Logit/Score*, or sentiment logit/score for brevity. These terms will be used in the following sections.

IV. Lifelong Learning Algorithm

This section presents our proposed lifelong learning algorithm. Notations are first introduced as follows. We define a collection of sentences in a new domain (indexed by i) as D_i^{TL}, where T indicates *target* and L indicates *labeled*. This means the sentences in D_i^{TL} have real labels (aspects and sentiments, annotated by humans). The annotated aspects and sentiments for those sentences are denoted as A_i^{TL} and S_i^{TL}. In addition, we define a collection of sentences in a past/assisting domain (indexed by j) as D_j^{PU}, where P means *past* and U means *unlabeled*. We thus have two corpora $\boldsymbol{D^{TL}} = \cup_i D_i^{TL}$, $i \in \{1, 2, ..g\}$, and $\boldsymbol{D^{PU}} = \cup_j D_j^{PU}$, $j \in \{1, 2, ..l\}$, where g and l denote the number of domains in these two corpora. Note that l is usually much larger than g because it is much easier to collect an unlabeled dataset for one domain rather than annotating detailed aspect sentiment for one domain. Associated with $\boldsymbol{D^{TL}}$, we have a collection of aspects $\boldsymbol{A^{TL}} = \cup_i A_i^{TL}$ and sentiment labels $\boldsymbol{S^{TL}} = \cup_i S_i^{TL}$. They are the input for our lifelong learning algorithm as shown in Alg. 1. From an overview perspective, Alg. 1 works in a three-step manner: first, assigning the aspect sentiment labels automatically for the big (unlabeled) data; second, building memory network classifiers and retaining (raw) knowledge; third, knowledge mining and utilization, where a newly-designed lifelong learning memory network will be introduced for integrating the learned knowledge into its learning process.

Algorithm 1 Overview of Lifelong Learning Algorithm

```
Input: $\boldsymbol{D^{TL}}, \boldsymbol{D^{PU}}, \boldsymbol{A^{TL}}, \boldsymbol{S^{TL}}$
1: $\boldsymbol{D^{PL}}, \boldsymbol{A^{PL}}, \boldsymbol{S^{PL}} \leftarrow$ AutoLabelingFull($\boldsymbol{D^{PU}}$)
2: or
3: $\boldsymbol{D^{PL}}, \boldsymbol{A^{PL}}, \boldsymbol{S^{PL}} \leftarrow$ AutoLabelingLite($\boldsymbol{D^{PU}}, \boldsymbol{A^{TL}}$)
4: $\boldsymbol{RK} \leftarrow \varnothing$
5: for $D_j^{PL} \in \boldsymbol{D^{PL}}$ do
6:     $RK_j^{PL} \leftarrow$ L2MN($D_j^{PL}, A_j^{PL}, S_j^{PL}, NLL\_MODE$)
7:     $\boldsymbol{RK} \leftarrow \boldsymbol{RK} \cup RK_j^{PL}$
8: end for
9: for $D_i^{TL} \in \boldsymbol{D^{TL}}$ do
10:     $ASA_i^T, CSE_i^T \leftarrow$KnowMining ($\boldsymbol{RK}, D_i^{TL}, A_i^{TL}$)
11:     L2MN($D_i^{TL}, A_i^{TL}, S_i^{TL}, ASA_i^T, CSE_i^T, LL\_MODE$)
12: end for
```

Step 1: Automatic Machine Labeling (lines 1-3) Note that initially no aspect or sentiment labels are available for input $\boldsymbol{D^{PU}}$. In order to make use of these unlabeled data, we design an automatic aspect sentiment labeling strategy that does not need human intervention. We refer to it as *auto-labeling*. Its idea is quite intuitive. That is, although an online review itself does not provide explicit aspect-level labels, it often contains/shows document-level rating to indicate its overall opinion. According to the theory of sentiment consistency [32] that the mentioned aspects should have consistent or similar sentiment orientation as shown by the whole review [1], aspect-based sentiment can be inferred to a great extent.

Specifically, while using Amazon review data[1] whose rating scores range from 1 to 5, we regard reviews with the rating of 5 (strongly positive) as reliable positive reviews and assume that opinions about the aspects discussed in each such review are also positive. Likewise, we deem reviews with the rating of 2 or 1 (strongly negative) as reliable negative reviews and consider the opinions on the aspects mentioned in each such review as negative. Certainly, this assumption may not always hold well and the resulting aspect-based sentiment labels are likely to contain noises. However, notice that we will have the following learning steps to mine reliable knowledge, instead of directly using the raw results generated from the past/assisting domains (for helping a new domain). Also, even with these (likely) noisy labels, our lifelong learning algorithm can produce reasonably good results, which will be shown in our experiments.

There are two possible ways to generate auto-labels, which are presented in lines 1 and 3. The $AutoLabelingFull$ function in line 1 is to extract all aspects mentioned in D^{PU} by using an unsupervised aspect extraction approach [1] while the $AutoLabelingLite$ function is a relaxed version that only focuses on the target aspects in $\boldsymbol{A^{TL}}$. $AutoLabelingLite$ is more efficient as we only need to match and keep the sentences containing the target aspects. We use $AutoLabelingLite$ for our experiments. As a result, we have auto-labeled sentences for all past domains $\boldsymbol{D^{PL}} = \cup_j D_j^{PL}, D_j^{PL} \subseteq D_j^{PU}$ along with their corresponding aspects $\boldsymbol{A^{PL}}$ and sentiments $\boldsymbol{S^{PL}}$.

Step 2: Building Classifiers and Raw Knowledge Retention (lines 4-8) For each past/assisting domain, we build a basic memory network (AMN/NLL) and retain its raw knowledge. It is important to note that here the knowledge retention does not mean that we simply save the classification results for each domain. Instead, we design proper representation to collect structured information learned by the model, which is defined as *knowledge* in this study. Specifically, two types of information will be collected, namely, attention and sentiment.

In terms of attention, we formulate its knowledge as distribution. That is, for each aspect in a domain, an aspect-sentiment attention distribution over words will be generated and retained. It basically reflects and summarizes the importance of all possible context words for a specific aspect in one domain. More concretely, the attention score of context (word) v_i for target t under sentiment r is denoted as $\alpha_{v_i,t,r}$, which is the sum over its attention divided by its total number of occurrences. It is calculated as:

$$\alpha_{v_i,t,r} = \begin{cases} 0, \sum_q^{N_D} \sum_p^{W_q} I(w_{q,p} = v_i) I(a_q = t) I(s_q = r) = 0 \\ \frac{\sum_q^{N_D} \sum_p^{W_q} \alpha_{q,p} I(w_{q,p}=v_i) I(a_q=t) I(s_q=r)}{\sum_q^{N_D} \sum_p^{W_q} I(w_{q,p}=v_i) I(a_q=t) I(s_q=r)}, otherwise \end{cases} \tag{3}$$

where N_D is the number of sentences in domain D and W_q is the number of words in sentence q. $w_{q,p}$ is the word in position

[1] Rating ranges can vary from different sites. In such cases, reviews with the highest and lowest scores from one site are used to obtain aspect-level labels.

q, p and v_i is the word i in the vocabulary. a_q and s_q denote the aspect and aspect-specific sentiment in the sentence q. $I()$ is an indicator function. Here the intuition is: if a (context) word is more positively or negatively correlated to an aspect, it should be assigned more attention most of the time when it co-occurs with the aspect. We thus can collect a set of aspect-specific attention distributions $\boldsymbol{\alpha}^{(j)}$ from domain j for all aspects ($\boldsymbol{A^{PL}}$), for example, $\alpha_{a,s}^{(j)}$ is the distribution of aspect a under sentiment s in domain j.

In terms of sentiment, the context-sentiment effect is the focus of accumulation. Recall that we can factorize the overall sentiment logit to the individual sentiment contribution of each context in memory networks (discussed in section III). We thus construct the knowledge as a context-sentiment matrix $M \in \mathbb{R}^{V \times K}$, which is the dot product of weight matrix W and output embeddings C, i.e., $M = WC$, in each domain. So a value in $M_{v,k}^{(j)}$ indicates the sentiment effect of a context word v for sentiment k in domain j.

For each past domain, such structured attention and sentiment information are accumulated and added to the knowledge set $\boldsymbol{RK}$. However, what we have collected thus far is treated as raw knowledge, and it is not ready for use to help a new domain. Given the noises from auto-labels and mis-classification results, raw knowledge inevitably contains errors. To ensure the knowledge quality, we need further knowledge mining.

Step 3: Knowledge Mining and Application (lines 9-12) The knowledge mining (KnowMining) step mines *reliable knowledge* from the raw knowledge. Such reliable knowledge will then be used in building the lifelong classifier (LL-mode L2MN) for new domains. The reliable knowledge contains two parts, the Aspect-Sentiment Attention (ASA) knowledge, and Context-Sentiment Effect (CSE) knowledge, corresponding to the two types of raw knowledge discussed above.

To distill reliable knowledge, we employ the theory of *Frequent Pattern Mining* (FPM) [33]. A frequent pattern is a set of items that appear frequently in a database of transactions above a minimum frequency threshold, called *minimum support*. Each transaction is a set of items. In our case, we treat words with non-zero attention values ($\alpha_{v_i,t,r} \neq 0$) as items and regard a set of words in one attention distribution as one transaction. As we have accumulated a number of attention distributions (i.e., transactions) towards aspects from the past domains, FPM can filter many errors (e.g., wrong words with high aspect-sentiment attention values) that happen only in few domains. They are infrequent patterns and will be filtered based on the minimum support. In other words, the remaining frequent patterns are regarded as reliable.

This conventional data mining technique (i.e., FPM) turns out to be highly effective, because its rationale aligns well with sentiment analysis. For example, if "scratched" is assigned with negative attention towards the aspect *screen* frequently in many domains like *Cellphone, Laptop, and Camera*, we would have more confidence that "scratched" is highly correlated to aspect *screen* (negatively).

With the infrequent items/words removed, we have *denoised* knowledge. We then calculate the distributional values from the denoised knowledge for all aspects. Specifically, we average the distribution values $\alpha_{v_i',t,r}^{(j)}$ learned from past domains, where v_i' stands for a frequent word under aspect t and sentiment r, to obtain a final set of (denoised) aspect-sentiment attention distribution $\{\alpha_{t_1,r}^{(i)}, \alpha_{t_2,r}^{(i)}, ...\}$ for a new/target domain i.

The above process results in the ASA_i^T. We also acquire CSE_i^T from the raw sentiment effect knowledge M in a similar way using FPM, to filter the words with high context-sentiment values but appearing infrequently across domains (i.e., likely noises). They will be stored in a knowledge base (KB) and used in a new domain as prior knowledge.

V. LIFELONG LEARNING MEMORY NETWORK (L2MN)

Here we present our proposed lifelong learning memory network (L2MN), which can leverage the learned prior knowledge to a new domain. Its model architecture is presented in Fig. 2. Recall that t is the vector representation of aspect a. m_q and c_q are the input and output representation of sentence q where $m_q, c_q \in \mathbb{R}^{d \times W_q}$. W_q denotes the number of words in q. $\alpha, o, s,$ and y are the attention, output vector, sentiment score, and class distribution respectively as introduced in section III. Without considering other factors, they construct a basic memory network. In other words, if we use no knowledge, L2MN can reduce to its NLL mode, i.e., the basic model AMN.

The ASA knowledge is incorporated into L2MN as two sets of knowledge-driven attention. To be concrete, given an aspect a, two types of aspect-sentiment attention distribution can be extracted from KB (with one-time creation effort) for the current domain, namely, aspect-positive attention distribution F_a^+ and aspect-negative attention distribution F_a^-. Next, the words in a sentence of current domain will be assigned with the prior aspect-positive and aspect-negative attentions. That is, additional positive attention α_p and negative attention α_n are produced for current sentence q. In this way, L2MN can utilize the accumulated attention knowledge from the past domains and provide properly estimated attention values to the context words in the sentence q of the current domain. Following the aforementioned example in section I, the negative-attention of "scratched" towards aspect *screen* can be encoded here as a type of prior information. So even if the provided data in the current domain are statistically insufficient to learn such attention ("scratch" for *screen*), this attention can still be possibly indicated by α_n from the self-accumulated ASA knowledge in L2MN.

With the involvement of α_p and α_n, L2MN moves forward to produce output representations o_p and o_n. Based on them and W, two additional sentiment scores s_p and s_n can be inferred. However, different from generating s (see Eq. 2), two additional vectors A_p and B_p are used here for producing s_p, and two additional vectors A_n, and B_n are used for creating s_n. $A_{p/n}$ is a polarity-projection vector and $B_{p/n}$ is a polarity-selection vector. We mainly explain how A_n and B_n work below as A_p and B_p work similarly. In a binary classification case (K=2), $y = [1, 0]$ denotes the negative class and $y = [0, 1]$

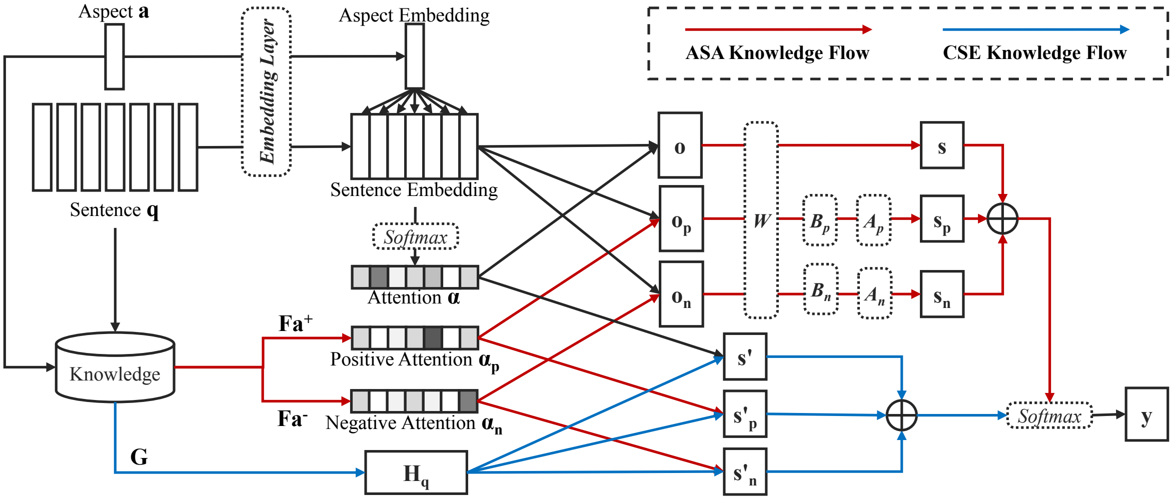

Fig. 2. Lifelong Learning Memory Network (L2MN). Color printing preferred.

denotes the positive class. The sentiment output s_p and s_n are calculated by:

$$\begin{aligned} s_p &= A_p B_p^T W o_p, A_p = [-1, 1]^T, B_p = [0, 1]^T \\ s_n &= A_n B_n^T W o_n, A_n = [1, -1]^T, B_n = [1, 0]^T \end{aligned} \tag{4}$$

where $o_p = \sum_i \alpha_{p,i} c_i$ and $o_n = \sum_i \alpha_{n,i} c_i$. Here B_n pinpoints the negative-sentiment effect and A_n promotes it towards the negative class and demotes it towards the positive class. That also explains why B_n is called polarity-selection vector and A_n is called polarity-projection vector. In the three-class classification case (K=3), we will have $A_n = [1, -1, -1]$ and $B_n = [1, 0, 0]$ (for negative sentiment), where the negative, neutral, and positive classes are denoted as [1, 0 ,0], [0, 1 ,0], and [0, 0 ,1] respectively.

The CSE knowledge is incorporated as a context-sentiment matrix $G \in \mathbb{R}^{K \times V}$, where V is the vocabulary size in the current domain. Note G is derived from $M(s)$ (context-sentiment matrices from past domains) with knowledge mining and vocabulary mapping, i.e., only the reliable knowledge and the words occurring in the current domain are used. It can be directly extracted from KB as well (with one-time creation effort). In regard to a particular sentence q, a sentence-specific matrix $H_q \in \mathbb{R}^{W_q \times K}$ encodes the prior sentiment effect of the context words in sentence q. Together with the attention α, another sentiment score s' will be produced, i.e., $s' = \alpha H_q$.

Furthermore, two other sentiment scores s'_p and s'_n can be added if we consider incorporating both types of knowledge simultaneously, where $s'_p = \alpha_p H_q$ and $s'_n = \alpha_n H_q$. They are used to encode the joint aspect-context sentiment effect learned from ASE and CSE. With them jointly considered, the final sentiment score for the aspect a in sentence q is calculated as:

$$\begin{aligned} s_{joint} &= s + s_p + s_n + s' + s'_p + s'_n \\ &= Wo + Wt + A_p B_p^T W o_p + A_n B_n^T W o_n \\ &+ \alpha H_q + \alpha_p H_q + \alpha_n H_q \end{aligned} \tag{5}$$

The final sentiment probability y is produced with a softmax operation $y = softmax(s_{joint})$. Note that ASA knowledge and CSE knowledge are used in both training and testing stages, which enables L2MN to consider the prior and in-domain information jointly.

Learning: The L2MN model is trained in an end-to-end manner by minimizing the cross entropy loss and using stochastic gradient descent. Let us denote a sentence and a target aspect as x and t respectively. They appear together in a pair format (x, t) as input and all such pairs construct the dataset D. $g_{(x,t)}$ is a one-hot vector and $g^k_{(x,t)} \in \{0, 1\}$ denotes a gold sentiment label, i.e., whether (x, t) shows sentiment k. $y_{x,t}$ is the model-predicted sentiment distribution for input (x, t). $y^k_{x,t}$ denotes the probability of being class k. Finally, the training loss is constructed as:

$$loss = -\sum_{(x,t) \in D} \sum_{k \in K} g^k_{(x,t)} \log y^k_{(x,t)} \tag{6}$$

VI. Experiments

A. Candidate Models for Comparison

The candidate models we compare can be categorized into four general groups: long short-term memory networks (LSTMs), memory networks (MNs), non-lifelong knowledge memory networks (NLKs) and lifelong learning memory networks (L2MNs). Note that while both NLKs and L2MNs are knowledge-based models, the difference is that L2MNs use the knowledge learned from our proposed lifelong learning algorithm but NLKs use other information as their (fed) knowledge. By comparing L2MNs and NLKs, we can gain an insight into the importance of knowledge mining in the lifelong learning setting.

AT-LSTM: This is a state-of-the-art LSTM/RNN based model with aspect embedding and attention modeling for ASC [2].

ATAE-LSTM: Another LSTM based model with aspect embedding used in both the input representation and hidden layer representation [2].

Memory Network (MN): End-to-end memory network [9].

Memory Network Layer-wise (MNL): A multiple-hops MN (a hop means a computational layer), where the embedding matrices are typed the same across different layers.

Memory Network Adjacent (MNA): Another version of multiple-hops MN, where the output embedding of one layer is the input embedding of its next layer.

ASC Memory Network (AMN/NLL): This is a memory network particularly proposed for the ASC task following [3]. It is used as our basic model without any knowledge incorporation, i.e., it can be viewed as the non-knowledge version of L2MN.

ASC Memory Network Multi-hops (AMNM): The multiple-hops version of AMN [3].

Raw-knowledge Memory Network (RKMN) : This is the first NLK model, which directly uses the raw knowledge extracted from past domains without further knowledge mining.

Lexicon-enhanced Memory Network (LexMN): A NLK model using an opinion lexicon as its knowledge. We use the opinion lexicon from [17], which consists of 2007 positive and 4873 negative sentiment words. These words play the role of ASA knowledge. Since these sentiment words are not learned from the past domains, we do not know their values in the aspect-sentiment attention distribution. We thus set a constant value ranging from {0.1, 0.2, ..., 1.0} for estimation and report the best result.

Universal-knowledge Memory Network (UKMN): Instead of applying the aspect-specific sentiment knowledge, this NLK model uses a form of universal sentiment knowledge. That is, the knowledge is an aspect-independent (or universal) sentiment attention distribution, which is the average sentiment distribution (of words) over all aspects from all past domains.

Universal-domain-knowledge Memory Network (UDKMN): This model is similar to UKMN but computes universal sentiment knowledge in another way. It first collects N sets of sentiment attention distribution from N domains, each of which is the average sentiment attention distribution over all aspects in each domain. It then averages these N distributions for the final universal knowledge. Note different domains may cover different numbers of aspects. In this case, UDKMN can mitigate the impact of domain difference.

Aspect-Sentiment Attention L2MN (ASA): Our proposed lifelong learning memory network using ASA knowledge.

Context-Sentiment Effect L2MN (CSE): Our proposed lifelong learning memory network using CSE knowledge.

ASA + CSE L2MN (JOINT): Our proposed lifelong learning memory network using both ASA and CSE knowledge.

B. Experimental Setup

Datasets: We use two groups of Amazon review data. The first group provides real aspect-level manual annotation of aspects and their corresponding sentiment polarities. This group of data is used for model evaluation since it contains gold labels. We also call it **Gold Data**. Specifically, four products *Camera*, *DVD Player*, *MP3*, and *Laptop* are used as four different **target domains (or target datasets)**. The first three datasets are from [17], each of which is split into training and test sets by 70% and 30%. Their data sizes are also different which help to test the model generality. The fourth dataset (*Laptop*) from SemEval 2014 [34] is a benchmark dataset that has been used in related studies [2], [3]. Its training and test sets have already been separated. Full data statistics are reported in Table I.

TABLE I
Statistics of the datasets in Gold Data.

Data		Positive		Neutral		Negative	
Dataset	**Size**	**Train**	**Test**	**Train**	**Test**	**Train**	**Test**
Camera	649	164	61	250	113	36	25
DVD Player	828	135	60	273	124	173	63
MP3	2016	349	167	834	334	225	107
Laptop	2966	994	341	464	169	870	128

The second group of data from [24] consists of reviews from 50 domains (50 datasets about different electronic products), but their reviews only have document-level ratings. So we use our proposed auto-labeling strategy to create aspect-level annotations. Since they are not gold labels, the data are not used for evaluation. However, they are still split into two sets as training and validation sets, so as to track the model learning performance. As discussed, they are used as **assisting/past domains** to help a target domain. We also call them **AST Data**.

Settings: For LSTMs and MNs, the models using no knowledge, only Gold data are used. For NLKs and L2MNs (except LexMN), Gold data and AST data are used together: a target domain (from Gold data) is regarded as the new domain and the 50 assisting domains (from AST data) are treated as the past domains. We then conduct experiments on the four different target domains independently to form four sets of evaluation.

Note that when a knowledge-based model starts to process a target domain, only the target domain data and self-accumulated knowledge can be used. No additional data from past domains are available, i.e., previous data cannot be accessed. So there is no specific source domain, which is different from other settings like transfer learning. Our experimental setup follows prior research about lifelong learning [13], [24], [29].

For all models, we use the same set of pre-trained word embeddings[2] learned from a Google News corpus for initialization. We randomize other model parameters from a uniform distribution U(-0.05, 0.05). The dimension of word embeddings and the size of hidden layers are 300 and the learning rate is 0.01. For the multiple-hops models, we set the hop number to 3 following the previous study [9]. For each model, its hyper-parameters are set by using the *Laptop* dataset, with 10% of its training data used as the validation set. All MN models use the location attention as suggested in [3]. For FPM, we empirically set the minimum supports to 8 and 3 for positive and negative sentiment knowledge, as the positive reviews are usually much more than the negative reviews according to the real-world data distribution (at the document level). Notice that this is a general FPM setting we suggest as it can basically work well for most domains, but we also found that fine-tuning the minimum supports for the four different target datasets/domains individually could lead to better results.

[2] https://github.com/mmihaltz/word2vec-GoogleNews-vectors

TABLE II
BINARY CLASSIFICATION RESULTS ON FIRST THREE DATASETS OF GOLD DATA.

Model Description			Camera			DVD Player			MP3		
Knowledge	**Group**	**Model**	**Mac.**	**Neg.**	**Pos.**	**Mac.**	**Neg.**	**Pos.**	**Mac.**	**Neg.**	**Pos.**
No Knowledge	LSTMs	AE-LSTM	72.98	57.89	88.06	78.75	80.30	77.19	82.53	78.26	86.80
		ATAE-LSTM	73.94	60.00	87.88	78.79	80.00	77.59	82.63	79.46	85.80
	MNs	MN	66.22	47.37	85.07	84.55	84.55	84.55	80.63	75.96	85.29
		MNL	56.96	32.43	81.48	83.74	83.61	83.87	81.31	77.42	85.20
		MNA	56.09	31.58	80.60	84.55	84.55	84.55	81.46	76.38	86.53
		AMN	72.98	57.89	88.06	81.22	82.44	80.00	82.39	77.83	86.96
		AMNM	73.94	60.00	87.88	83.69	84.62	82.76	82.46	78.05	86.88
With Knowledge	NLKs	RKMN	75.13	61.54	88.72	81.22	82.44	80.00	79.85	75.00	84.71
		LexMN	74.17	59.46	88.89	84.55	84.55	84.55	82.32	77.61	87.03
		UKMN	75.13	61.54	88.72	83.73	84.13	83.33	85.33	81.90	88.76
		UDKMN	77.20	65.00	89.39	82.93	82.93	82.93	85.64	82.13	89.15
	L2MNs	ASA	75.13	61.54	88.72	85.36	85.25	85.48	85.79	83.26	88.69
		CSE	79.84	69.77	89.92	87.79	87.39	88.19	**87.77**	**85.19**	**90.36**
		JOINT	**82.19**	**73.91**	**90.48**	**88.62**	**88.71**	**88.52**	87.37	84.65	90.09

We test all models with two aspect sentiment classification settings: (1) Binary classification: all models are trained and tested only using positive and negative samples. (2) Three-class classification: all models are trained and tested on the full data including positive, negative, and neutral samples.

Evaluation Measure: Since the class distribution is skewed in almost all settings on all target datasets (except the binary classification on the *DVD Player* dataset), F1 score is primarily used as our evaluation measure. Accuracy (Acc.) is not suggested for imbalanced datasets, as an inferior model may simply classify most samples as the majority class to achieve a high score. Specifically, both the F-Macro (averaged F1-score over all classes) and all individual class-based F1 scores will be reported. We denote F-Macro as Mac. in the following tables. The positive, neutral, and negative F1 scores are denoted as Neg., Neu., and Pos. respectively. We also provide a P&N measure to show the averaged F1 score of Pos. and Neg. for the three-class classification tasks.

C. Result Analysis

We provide quantitative results with analyses in this subsection. We first present the comprehensive results for *Camera*, *DVD Player*, and *MP3*. We then analyze *Laptop*, where we also report accuracy as it is used in previous studies.

Binary Classification Results: We report the binary classification results in Table II. The highest score in each measure is marked in bold. We have the following observations:

1) L2MNs consistently perform the best on Mac., Pos., and Neg. measures. Among L2MNs, the JOINT model achieves the best results on *Camera* and *DVD Player*. CSE performs slightly better than JOINT on *DVD Player* but their scores are very close. Notice that ASA, CSE, and JOINT can all improve AMN/AMNM markedly, which shows the effectiveness of both types of knowledge.
2) Comparing L2MNs with NLKs, we can see that L2MNs work better on all datasets. Although some NLKs have competitive performances with L2MNs on some datasets, they are unstable. For example, UDKMN performs better than ASA on *Camera* and achieves the highest Mac. score among NLKs, but it works poorly on *DVD Player*. Also, note while both L2MNs and NLKs utilize knowledge to help, the superior results from L2MNs indicate the necessity of knowledge mining. In other words, simply involving extra information from (past) data like what NLKs do does not guarantee performance gains.
3) Comparing NLKs with MNs, we can see the highest scores are always from the NLKs group on all datasets, which shows the involvement of proper prior knowledge can benefit this task. LSTMs have similar performances to AMN/AMNM, but are inferior to NLKs.

Three-Class Classification Results: Results are reported in Table III. Note for all knowledge-based models, we so far only accumulate and incorporate positive knowledge and negative knowledge (the mining and utilization of neutral knowledge are left to future work). We can draw additional conclusions from the results:

1) L2MNs again achieve the best performance on Mac. and also on P&N. This means, even if only positive and negative knowledge are used, L2MNs can still improve the overall classification results. They generally have lower Neu. scores than AMN and AMNM, which is expected, but can attain much higher Neg. and Pos. scores. We anticipate that with proper neutral knowledge being mined and added, better results can be produced.
2) LSTMs and MNs are inferior to L2MNs and also NLKs. NLKs, whose results are omitted here due to space limit, work better than MNs but worse than L2MNs, very similar to the observations we have from binary classification. We thus do not repeat their analyses.
3) Last but not least, L2MNs generate consistently good results on datasets of different sizes. This supports the hypothesis we discussed in section I. That is, with

TABLE III
THREE-CLASS CLASSIFICATION RESULTS ON FIRST THREE DATASETS OF GOLD DATA.

	Camera					**DVD Player**					**MP3**				
Model	**Mac.**	**P&N**	**Neg.**	**Neu.**	**Pos.**	**Mac.**	**P&N**	**Neg.**	**Neu.**	**Pos.**	**Mac.**	**P&N**	**Neg.**	**Neu.**	**Pos.**
AE-LSTM	59.87	52.99	47.06	73.63	58.91	63.74	56.74	56.92	77.74	56.57	57.63	47.82	33.94	77.24	61.70
ATAE-LSTM	63.66	55.91	42.86	79.17	68.97	64.97	58.99	60.34	76.92	57.63	67.44	60.62	53.76	81.09	67.47
MN	64.17	56.61	41.67	79.30	71.54	62.94	57.20	59.54	74.40	54.87	65.13	60.97	55.81	73.45	66.13
MNL	59.01	50.40	38.30	76.23	62.50	64.08	60.58	62.82	71.07	58.33	65.76	61.45	57.40	74.38	65.51
MNA	61.60	53.56	42.55	77.68	64.57	64.06	59.46	62.77	73.25	56.14	65.72	61.28	57.51	74.62	65.05
AMN	67.44	59.81	48.65	82.70	70.97	67.43	60.81	66.67	**80.68**	54.95	69.22	63.27	58.38	81.11	68.17
AMNM	67.90	60.75	48.78	82.20	**72.73**	68.48	62.80	66.67	79.85	58.93	70.08	64.71	59.70	80.81	69.72
ASA	**69.07**	62.00	51.28	**83.19**	**72.73**	68.49	63.56	65.57	78.36	61.54	69.92	63.93	60.77	**81.91**	67.08
CSE	67.54	60.83	52.63	80.97	69.03	71.44	67.85	70.31	78.63	65.38	69.61	63.74	58.70	81.34	68.79
JOINT	68.59	**62.12**	**53.66**	81.51	70.59	**73.25**	**70.33**	**70.49**	79.07	**70.18**	**71.22**	**66.14**	**61.39**	81.38	**70.89**

lifelong learning, the self-accumulated knowledge can alleviate some shortcomings of memory networks and ensure their stabler or better performance on ASC.

Laptop Results: Table IV reports the results on the *Laptop* dataset, which has been tested in previous studies but only accuracy scores were provided. Here we want to gain further insights. For consistency, we also report accuracy but will shed more light on the performance of every individual class. Additionally, we provide a multiple-hop L2MN (JOINT-3 with 3 hops) as opposed to the regular single-hop L2MN (JOINT-1). We can observe that both JOINT-3 and JOINT-1 outperform the state-of-the-art baseline models, and JOINT-3 achieves the best scores on almost all measures. We have also tried JOINT-3 on the previous three datasets but found limited improvement, probably because a single-hop version of L2MN already works quite well on smaller datasets (with the help of knowledge accumulation), i.e., the performance gains achieved from NLL to JOINT-1 are already noticeably large. This also indicates that deeper lifelong memory networks like JOINT-3 are more suitable for bigger data.

TABLE IV
RESULTS ON LAPTOP.

Model	**Mac.**	**Neg.**	**Neu.**	**Pos.**	**Acc.**
AE-LSTM	62.45	55.26	50.35	81.74	68.50
ATAE-LSTM	59.41	55.27	42.15	80.81	67.40
AMN	61.77	56.78	48.78	79.76	67.08
AMNM	65.62	63.23	51.37	82.25	70.86
JOINT-1	67.02	63.43	**55.70**	81.91	71.32
JOINT-3	**67.92**	**65.57**	54.48	**83.70**	**72.73**

Knowledge Examples: Table V shows aspect-sentiment attention knowledge examples for aspects *product* and *software*. For each aspect, the top words are presented along with their attention distribution values. For *product*, words like "love", "excellent", and "amazing" have the strongest attentional correlation with positive polarity. That means, if the sentence "*I bought a new product and love it so much*" is given, L2MN can use its prior knowledge to better place the attention on "love" and assign stronger positive sentiment. On the other hand, when the sentence "*I have returned the product*" is given, L2MN is more likely to generate negative sentiment because "returned" is a word associated with strong negative attention toward *product*. Likewise, the sentence "*the software is intuitive to use*" would be better identified by L2MN as showing positive sentiment, since "intuitive" is learned/accumulated as the knowledge of strong positive attention towards aspect *software*.

TABLE V
KNOWLEDGE EXAMPLES.

Aspect-Sentiment Attention Knowledge		
Aspect	**Senti.**	**Attention Distribution**
product	Pos.	love(0.287), excellent (0.283), amazing (0.279), happy (0.263), definitely (0.228), highly (0.216) ...
	Neg.	disappointed (0.258), defective (0.237), poor (0.178), terrible (0.122), returned (0.117), waste (0.117) ...
software	Pos.	easy (0.173), intuitive (0.129), great (0.105), nice (0.097), good (0.07), simple (0.076) ...
	Neg.	horrible (0.170), bad (0.097), problem (0.087), poor (0.074), tried (0.070), barrel (0.069) ...

D. *Case Study*

We present a real case that is wrongly classified by AMN/NLL but corrected by L2MN in Fig. 3. Let us first take a look at the attention captured by the two models in the same sentence "*however it has failed to deliver on quality*", where "quality" is the given aspect. The attention is shown horizontally as a heat map. With the automatically accumulated knowledge, L2MN better identifies that "failed" is an important context for "quality" and assigns it a higher attention weight, shown in darker red color compared with AMN. The sentiment logit of context denotes the sentiment score of a word towards sentiment classes (negative, neutral, and positive), as discussed in section III. With stronger attention, we can see the sentiment logit towards the negative class becomes higher.

We present another example in Fig. 4. In this example, the given aspect is "remote" in the sentence "*my other gripe is the*

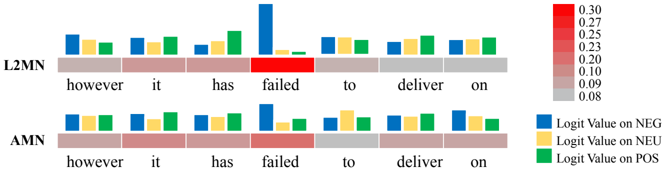

Fig. 3. Attention and sentiment logit. Attention is shown as a heat map horizontally. Darker color means higher attention. Sentiment logits of contexts are shown vertically. A higher value indicates a stronger sentiment score.

incredibly crappy remote which is worse than other cheaper apex units." This sentence is difficult to predict for two reasons. First, the attention becomes difficult to locate, as this sentence is relatively complicated and there are multiple sentiment-bearing context words. As we can see, AMN cannot place the attention well. In contrast, L2MN can capture attention better and find the correct sentiment context word "crappy". However, there is still another issue, even if a model detects the attention correctly. That is, a model needs to figure out the correct sentiment orientation of a context word. We found that "crappy" is a relatively infrequent word in its current domain (i.e., target dataset), which makes its sentiment polarity hard to judge. As shown in Fig. 4, "crappy" has a higher positive sentiment score (shown in green color) than other two sentiments in AMN. In contrast, L2MN still works well and identifies that "crappy" is a negative sentiment word (the negative sentiment logit of "crappy" shown in blue color is greater than other two sentiments). This is attributed to the accumulated context sentiment effect (CSE) knowledge.

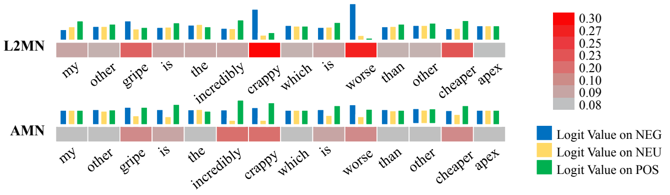

Fig. 4. Attention and Sentiment Logit.

VII. Conclusion

Memory networks are state-of-the-art neural models for the ASC task, but two crucial issues caused by data scarcity can hinder their performance. To address them, we aimed to propose a general solution that can make memory networks work consistently better. To achieve this goal, we employed the idea of lifelong learning and designed a novel three-step lifelong learning approach for ASC. In addition, a new lifelong learning memory network (L2MN) model was developed, which can leverage the meta-mined ASA knowledge and CSE knowledge to help future tasks. Experimental results using real-world datasets demonstrated the effectiveness of our approach.

Acknowledgment

This work was partially supported by National Science Foundation (NSF) under grant nos. IIS-1407927, IIS-1650900, and a research gift from Huawei Technologies Co. Ltd.

References

[1] B. Liu, "Sentiment analysis and opinion mining," *Synthesis lectures on human language technologies*, 2012.

[2] Y. Wang, M. Huang, X. Zhu, and L. Zhao, "Attention-based lstm for aspect-level sentiment classification." in *EMNLP*, 2016.

[3] D. Tang, B. Qin, and T. Liu, "Aspect level sentiment classification with deep memory network," in *EMNLP*, 2016.

[4] S. Wang, S. Mazumder, B. Liu, M. Zhou, and Y. Chang, "Target-sensitive memory networks for aspect sentiment classification," in *ACL*, 2018.

[5] L. Jiang, M. Yu, M. Zhou, X. Liu, and T. Zhao, "Target-dependent twitter sentiment classification," in *ACL*, 2011.

[6] S. Kiritchenko, X. Zhu, C. Cherry, and S. Mohammad, "Nrc-canada-2014: Detecting aspects and sentiment in customer reviews." in *SemEval*, 2014.

[7] J. Wagner, P. Arora, S. Cortes, U. Barman, D. Bogdanova, J. Foster, and L. Tounsi, "Dcu: Aspect-based polarity classification for semeval task 4," in *SemEval*, 2014.

[8] J. Weston, S. Chopra, and A. Bordes, "Memory networks," in *ICLR*, 2015.

[9] S. Sukhbaatar, J. Weston, R. Fergus *et al.*, "End-to-end memory networks," in *NIPS*, 2015.

[10] L. Zhang, S. Wang, and B. Liu, "Deep learning for sentiment analysis: A survey," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 2018.

[11] V. Mnih, N. Heess, A. Graves *et al.*, "Recurrent models of visual attention," in *NIPS*, 2014.

[12] S. Thrun, "Lifelong learning algorithms," *Learning to learn*, 1998.

[13] Z. Chen and B. Liu, "Lifelong machine learning," *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 2016.

[14] B. Pang, L. Lee, and S. Vaithyanathan, "Thumbs up?: sentiment classification using machine learning techniques," in *EMNLP*, 2002.

[15] R. Socher, J. Pennington, E. H. Huang, A. Y. Ng, and C. D. Manning, "Semi-supervised recursive autoencoders for predicting sentiment distributions," in *EMNLP*, 2011.

[16] Y. Kim, "Convolutional neural networks for sentence classification," in *EMNLP*, 2014.

[17] M. Hu and B. Liu, "Mining and summarizing customer reviews," in *SIGKDD*, 2004.

[18] X. Ding, B. Liu, and P. S. Yu, "A holistic lexicon-based approach to opinion mining," in *WSDM*, 2008.

[19] D.-T. Vo and Y. Zhang, "Target-dependent twitter sentiment classification with rich automatic features." in *IJCAI*, 2015.

[20] M. Zhang, Y. Zhang, and D.-T. Vo, "Neural networks for open domain targeted sentiment." in *EMNLP*, 2015.

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

[22] P. Chen, Z. Sun, L. Bing, and W. Yang, "Recurrent attention network on memory for aspect sentiment analysis," in *EMNLP*, 2017.

[23] J. Liu and Y. Zhang, "Attention modeling for targeted sentiment," in *EACL*, 2017.

[24] Z. Chen and B. Liu, "Topic modeling using topics from many domains, lifelong learning and big data," in *ICML*, 2014.

[25] T. M. Mitchell, W. W. Cohen, E. R. Hruschka Jr, P. P. Talukdar, J. Betteridge, A. Carlson, B. D. Mishra, M. Gardner, B. Kisiel, J. Krishnamurthy *et al.*, "Never ending learning." in *AAAI*, 2015.

[26] R. Caruana, "Multitask learning," in *Learning to learn*. Springer, 1998.

[27] S. J. Pan and Q. Yang, "A survey on transfer learning," *TKDE*, 2010.

[28] Q. Liu, B. Liu, Y. Zhang, D. S. Kim, and Z. Gao, "Improving opinion aspect extraction using semantic similarity and aspect associations." in *AAAI*, 2016.

[29] L. Shu, B. Liu, H. Xu, and A. Kim, "Lifelong-rl: Lifelong relaxation labeling for separating entities and aspects in opinion targets." in *EMNLP*, 2016.

[30] S. Wang, Z. Chen, and B. Liu, "Mining aspect-specific opinion using a holistic lifelong topic model," in *WWW*, 2016.

[31] Z. Chen, N. Ma, and B. Liu, "Lifelong learning for sentiment classification." in *ACL*, 2015.

[32] R. P. Abelson, "Whatever became of consistency theory?" *Personality and Social Psychology Bulletin*, 1983.

[33] R. Agrawal, R. Srikant *et al.*, "Fast algorithms for mining association rules," in *VLDB*, 1994.

[34] M. Pontiki, D. Galanis, J. Pavlopoulos, H. Papageorgiou, I. Androutsopoulos, and S. Manandhar, "Semeval-2014 task 4: Aspect based sentiment analysis," in *SemEval*, 2014.