Enhancing Aspect-Based Sentiment Analysis With Capsule Network

JINDIAN SU1, SHANSHAN YU2,3, AND DA LUO1

1College of Computer Science and Engineering, South China University of Technology, Guangzhou 510640, China
2College of Medical Information Engineering, Guangdong Pharmaceutical University, Guangzhou 510640, China
3Guangdong Province Precise Medicine and Big Data Engineering Technology Research Center for Traditional Chinese Medicine, Guangdong Pharmaceutical University, Guangzhou 510640, China

Corresponding authors: Jindian Su (sujd@scut.edu.cn) and Shanshan Yu (susyu@139.com)

This work was supported in part by the Research and Development Program in the Key Areas of Guangdong Province under Grant 2018B010109004, in part by the National Natural Science Foundation of China under Grant 61936003, in part by the Applied Scientific and Technology Special Project of the Department of Science and Technology of Guangdong Province under Grant 20168010124010, in part by the Natural Science Foundation of Guangdong Province under Grant 2015A030310318, in part by the Medical Scientific Research Foundation of Guangdong Province under Grant A2015065, and in part by the Innovation Project of Guangdong Pharmaceutical University under Grant 52159433.

ABSTRACT Existing feature-based neural approaches for aspect-based sentiment analysis (ABSA) try to improve their performance with pre-trained word embeddings and by modeling the relations between the text sequence and the aspect (or category), thus heavily depending on the quality of word embeddings and task-specific architectures. Although the recently pre-trained language models, i.e., BERT and XLNet, have achieved state-of-the-art performance in a variety of natural language processing (NLP) tasks, they still subject to the aspect-specific, local feature-aware and task-agnostic challenges. To address these challenges, this paper proposes a XLNet and capsule network based model XLNetCN for ABSA. XLNetCN firstly constructs auxiliary sentence to model the sequence-aspect relation and generate global aspect-specific representations, which enables to enhance aspect-awareness and ensure the full pre-training of XLNet for improving task-agnostic capability. After that, XLNetCN also employs a capsule network with the dynamic routing algorithm to extract the local and spatial hierarchical relations of the text sequence, and yield its local feature representations, which are then merged with the global aspect-related representations for downstream classification via a softmax classifier. Experimental results show that XLNetCN outperforms significantly than the classical BERT, XLNet and traditional feature-based approaches on the two benchmark datasets of SemEval 2014, Laptop and Restaurant, and achieves new state-of-the-art results.

INDEX TERMS Aspect-based sentiment analysis, natural language processing, text analysis, deep learning, neural network.

I. INTRODUCTION

As a fine-grained task of the sentiment analysis in natural language processing, aspect-based sentiment analysis (ABSA) [1], [2], also known as target-specific sentiment classification [3], has attracted attentions from many researchers. Compared with traditional sentiment analysis, ABSA aims to identify more complete and in-depth sentiment polarity of a text sequence towards a given aspect, usually in the form of explicitly mentioned aspect terms or implicit aspect categories (e.g., [4]). Most of previous neural models for ABSA have mainly contributed to utilizing pre-trained word embeddings as additional features and designing task-specific neural structures with some complex syntactic features (e.g., [5]–[7]) or attention mechanisms (e.g., [3], [4], [8]–[11]). More recently, some scholars even propose complicated models with the help of more advanced neural structures, e.g., transformation networks [12], parameterized convolutional neural networks (PCNN) [13], gated convolutional networks (GCN) [14], memory networks [15], graph networks [16], [17] or semantic cognition networks (SCN) [18]. However, just as Li et al. pointed out in [19], the improvement of these models measured by the accuracy or F1 score has reached a bottleneck, because the commonly used embeddings are pre-trained via word2vec or GloVe, which can only provide context-independent word-level features and are insufficient for capturing the complex semantic dependencies in the sentence.

To address context-independent problems and capture context-aware relations, some scholars propose contextu-
alized word embeddings by pre-training them over large-scale datasets with deep LSTM, e.g. CoVe (Contextualized Word Vectors) [20], ELMo (Embedding from Language Models) [21], ULM-Fit (Universal Language Model Fine-tuning) [22], or Transformer e.g. OpenAI GPT (Generative Pre-Training) [23], BERT (Bidirectional Encoder Representation from Transformers) [24] and XLNet [25]), which assign different contextual embeddings to each word and fine-tune them in a lightweight task-specific network.

More recently, some scholars have conducted some initial and fine-tune them in a lightweight task-specific network. Especially, Xu et al. in [26] propose a novel post-training approach on BERT to enhance the fine-tuning performance for aspect extraction and aspect sentiment classification. Chi et al. [27] also propose a BERT-based model for constructing auxiliary sentence from the aspect and converting the ABSA into a sentence-pair classification task. Based on the work of BERT, Zhilin et al. [25] further propose another denoising auto-encoding based pre-training method, XLNet, for overcoming the limitations of neglecting dependency between the masked positions and pretrain-finetune discrepancy. Although the experiment results from [25] show that XLNet outperforms BERT on 20 tasks, i.e., question answering, natural language inference, sentiment analysis, and document ranking, it still lacks of more extensive applications and analysis on ABSA task.

In this paper, we try to apply XLNet and capsule network [28] into the task of ABSA, aiming to alleviate the following challenges:

(1) Aspect-specific challenge. Most of feature-based works try to employ various complex attention mechanisms and neural networks for catching the relations between the aspect and the context, and generating aspect-specific representations for final classification. Canonical BERT or XLNet model usually takes the hidden state vector of some special token in the input sequence (e.g., the dummy token [CLS]) as the global aggregated representations of the sequence without explicitly considering the relations between the aspect and the sequence, thus subjecting to aspect-specific challenge.

(2) Local feature-aware challenge. Both XLNet and BERT are good at capturing the long-range dependency relations of the sequence with the help of Transformer-XL or Transformer, but they usually neglect the local features of the sequence such as partial and hierarchical relations that play an important role in traditional feature-based methods. Such local features might provide informative messages for final prediction.

(3) Task-agnostic challenge. To enhance the performance of fine-tuning methods, it is necessary to ensure fully learning task-related knowledge during the fine-tuning step. However, the limited number of supervised training data usually causes the failure of full task-awareness of the model. Consequently, how to make full use of the limited supervised training data for fine-tuning in the downstream task is of great importance. Sun et al. point out in [27] that constructing an auxiliary sentence from the aspect and converting ABSA to a sentence-pair classification task with the help of NSP (next sentence prediction) training objective can achieve further improvements over canonical BERT. But whether their work is effective for XLNet or not is still left for verifications due to the removing of NSP training objective in XLNet.

In this paper we propose a XLNet and capsule network-based model for ABSA by firstly extending the work of [27] to XLNet for verifying their aspect-specific and task-agnostic enhancing abilities. Then, we propose a capsule network with the corresponding dynamic routing algorithm to explore the partial and hierarchical relations of the sequence. We also present some merging strategies to yield the final representations of the sequence, which are fed to a softmax classifier. Finally, we conduct extensive experiments over some benchmark corpus to show the effectiveness of our proposed method. To the best of our knowledge, this is the first work that capsule networks and XLNet are empirically investigated for ABSA.

The rest of this paper is organized as follows: Section II briefly summarizes some related works about pre-training language models and capsule network. Section III presents the details of our model, including the structure and its implementations. Section IV conducts extensive experiments over two widely-used datasets and gives in-depth analyses. The conclusions and the directions for the future research are provided in Section V.

II. RELATED WORKS

A. PRE-TRAINING LANGUAGE MODELS

Many previous works have been contributed to ABSA (i.e. the aspect-level sentiment classification) by using various canonical neural networks, e.g. MemNet [29], RAM [8], IAN [30], DSAN [6] and DTDa [7], all of which tried to explicitly model the aspect information and generate aspect-specific representations for improving the final classification performance. Ma et al. also propose termed Sentic LSTM in [11] by augmenting the long short-term memory (LSTM) network with a hierarchical attention mechanism consisting of a target-level attention and a sentence-level attention, as well as exploiting the integration of commonsense knowledge into the recurrent encoder. Although they all used various canonical pre-trained word embeddings, e.g. word2vec or GloVe, as an important component in downstream models to offer significant improvements over embeddings learned from scratch, the polysemous and task-specific structure dependency problems have still posed many restrictions.

In the past two years, with the rapid developments and successful applications of deep contextualized word embeddings such as CoVe [20], ELMo [21], ULM-Fit [22], OpenAI GP [23] and BERT [24], the methods based on pre-training language models on a large network with a great amount of unlabeled data and fine-tuning in downstream tasks have made a breakthrough in many natural language
understanding (NLU) tasks. Unlike ELMo and ULMFiT that are intended to provide additional features for a particular architecture that bears human’s understanding of the end task, BERT leverages heavily on language model pre-training and has achieved state-of-the-art performance in many NLU tasks ranging from sequence to sequence-pair classification to question answering. The pre-trained BERT can be easily fine-tuned with just one additional output layer to create a state-of-the-art model for a wide range of tasks. With the great success of BERT, some scholars have conducted initial attempts to couple BERT with downstream neural models for great success of BERT, some scholars have conducted initial state-of-the-art model for a wide range of tasks. With the assistance of XLNet on the ABSA task, enhancing the relations between the sequence and the aspect. They also apply pre-trained BERT and obtain new state-of-the-art results on SemEval 2014 dataset. Huang and Carley [32] propose a novel target-dependent graph attention network (TD-GAT) for ABSA by explicitly utilizing the dependency relationship among words and also demonstrate that using BERT representations further substantially boosts the performance.

Although the great success achieved by BERT, it might subject to the problems of neglecting dependency between the masked positions and pretrained-finetune discrepancy. As a result, Zhilin et al. [25] propose another denoising auto-encoding based pre-training method XLNet, which is shown to outperform BERT-related models on SQuAD, GLUE, RACE and other 17 datasets. Compared with BERT, XLNet has demonstrated its advantages in treating the long document and text generation tasks, e.g. language understanding [33] and question-answering [34]. But there still lack of further extensive applications of XLNet on the ABSA task, as well as comparisons with BERT-based works.

B. CAPSULE NETWORK

The most classical and representative approach to extract the spatial patterns of the sequence is CNN. However, CNN is unavoidably more restricted to encode rich structures presented in a sequence, especially hierarchical relations between different layers. As a result, Sara et al. [28] propose the notion of capsule network with an iterative routing process to decide the credit attributions between nodes from lower to higher layers, thus allowing to encode intrinsic spatial relations between the parts and the whole. Previous related works about capsule network have mainly focused on the fields of computer visions and graphs. In the past two years, some scholars began to demonstrate its promising potentials in NLP tasks. For example, Wei et al. [35] investigate capsule network with dynamic routing for text classification and propose three strategies to stabilize the dynamic routing processing. Their experimental results show that capsule network performs better than traditional max pooling, average pooling and self-attention mechanisms on the basis of recurrent neural network and CNN encoders, and achieve competitive results over the compared baseline methods on text classification benchmarks, especially multi-label text classifications. Zhang et al. [36] explore the using of attentional capsule network for relation extraction in a multi-instance multi-label learning framework and demonstrate that capsule network improves the precision of the predicted relations and particularly multiple entity-pair relation extraction. Some scholars also use capsule network for sentiment analysis, e.g. caps-BiLSTM [37] and SC-BiCapsNet [38], and demonstrate its effectiveness in improving the accuracy of classification results. However, these works mainly rely on the traditional word embeddings trained via word2vec or GloVe, and using capsule network and dynamic routing algorithm to improve the sensitive abilities about spatial and hierarchical information. There still lack of the further applications of capsule network on ABSA.

III. METHODOLOGY

A. TASK DEFINITION

For ABSA classification task, a text sequence might contain different aspects associated with different sentiment polarities. For example, the sentiment polarity of “Staffs are not that friendly, but the taste covers all.” will be positive if considering the aspect food, but negative for the aspect service.

Let \( x = (x_1, \ldots, x_i, \ldots, x_j, \ldots, x_k) \) be a word sequence consisting of \( k \) tokens, \( a \) be an aspect (e.g. \( a_i \) or \( a_j \)), and \( y = \{y_1, \ldots, y_m\} \) be a set of pre-defined sentiment polarity categories (e.g. \{negative, neutral, positive\}), the goal of ABSA task is to identify the correct sentiment polarity of \( x \) with respect to \( a \), which can be formulated as a function \( f \) that measures the conditional probability distributions over all possible labels in \( y \) as follows:

\[
 f : (x, a) \rightarrow \{y_1, \ldots, y_m\}
\]

B. MODEL OVERVIEW

In this section, we propose a XLNet and capsule network-based model XLNetCN for ABSA and its overall architecture is illustrated in Figure 1.

XLNetCN consists of four parts:

1. Input layer. It aims to build the suitable input sequence in the required format of the XLNet model, which can be a single sequence or a pair of sequences. After that, the SentencePiece [39] are used for tokenization, and the position encoding and segment recurrence mechanism are employed to yield the final input representations. In XLNetCN, we use the aspect tokens to build auxiliary sentence and form it into a sequence pair with the target text sequence.

2. XLNet Layer. It uses XLNet as the encoder that consists of 12 Transformer-XL blocks and 12 self-attention heads, and then outputs the whole layer hidden representations. The vectors for the special token [CLS] are regarded as the global aspect-related representations of the text sequence.

3. CapsNet Layer. It takes a capsule network to further extract the spatial and hierarchical features from the
text sequence via a dynamic routing algorithm, which are compressed into the representation with pre-determined size aiming to cover more local n-gram features.

(4) Output layer. Some pooling strategies are explored to combine the global aspect-specific representations from the XLNet layer and the local spatial and hierarchical representations from the CapsNet layer, thus yielding the final representation of the text sequence that are fed to a simple softmax classifier for calculating the conditional probability distributions over pre-defined labels.

C. INPUT LAYER

Inspired by the work of [27], we propose a method of constructing auxiliary sentence and converting the original classification task (might be multi-label) into a sentence-pair one. To facilitate the comparisons with BERT-pair in [27] and other BERT-based models in [24] whose each input only consists of a text sequence, we construct different auxiliary sentences and denote them in Table 1

| Methods | Output | Auxiliary Sentence |
|---------|--------|--------------------|
| single  | S.P.   | None               |
| QA      | S.P.   | Pseudo-Question w/o aspect and w/o S.P. |
| QA-A    | S.P.   | Pseudo-Question w/o S.P., but w/ aspect |
| NS-A    | S.P.   | Words w/ aspect    |
| AS-AP   | (0, 1) | Pseudo-sentence w/ aspect and w/ S.P. |
| AS-P    | (0, 1) | Pseudo-sentence w/o aspect, but w/ S.P. |
| AW-AP   | (0, 1) | Words w/ aspect and w/ S.P. |
| AW-P    | (0, 1) | Words w/o aspect, but w/ S.P. |

More concretely, the auxiliary sentences listed above are further explained as follows:

- **Sentence for single** It means that the input consists of the original text sequence without auxiliary sentence, just as the canonical BERT model for text classification, e.g. [24], [26].
- **Sentence for QA** The auxiliary sentence to generate from the aspect is a pseudo-question without containing the aspect and sentiment polarity information, e.g. “what is the polarity?”.
- **Sentence for QA-A** The auxiliary sentence to generate is a pseudo-question that contains the aspect tokens. For example, given the aspect “service”, the generated sentence is “what is the polarity of the service?”.
- **Sentence for NS-A** The auxiliary sentence to generate is the aspect tokens. For example, given the aspect “service”, the generated sentence is “service”.
- **Sentence for AS-AP** The auxiliary sentence to generate from the aspect is a pseudo-sentence that contains both the aspect and sentiment polarity information, e.g. “The polarity of service is positive”. The task also needs to be converted into a binary classification one for obtaining the probability distribution over the set {0, 1}, in which the probability value of 1 is used as the matching score.
- **Sentence for AS-P** Similar like AS-AP except for removing away the aspect information, e.g. “The polarity is negative”, “The polarity is neutral” and “The polarity is positive”.
- **Sentence for AW-AP** Similar like AS-AP except that the auxiliary sentence only includes the aspect and sentiment polarity tokens, e.g. “service negative”, “service neutral” and “service positive”.
- **Sentence for AW-P** Similar like AW-AP except for only including the sentiment polarity tokens, e.g. “negative”, “neutral” and “positive”.

For AS-AP, AS-P, AW-AP and AW-P, the task also should be converted from the original predictions over a multi-label set (e.g. {“negative”, “neutral”, “positive”}) to a binary set (e.g. {0, 1}) that indicates the relations between two sequences in the pair. For example, given the aspect “service” with the pre-defined set {“negative”, “neutral”, “positive”} in the situation of AS-AP, three auxiliary sentences are generated as: “The polarity of service is negative”, “The polarity of service is neutral” and “The polarity of service is positive” respectively. We take the label of the sequence with the highest matching score for the predicted result. As shown in the latter experiments, this is a necessary operation for significantly improving the performance of the classifier.

After building the input sequence, we follow the same token pre-treatments of XLNet by taking SentencePiece [39] for tokenization and restricting the input sequence to be no more than the given length threshold \( t \) such that \( t \leq 512 \). For the sentence-pair situations, if the input sequence, e.g. \((x_{1:k}, a_{1:r})\), exceeds more than \( t \) tokens, we preserve the whole auxiliary sentence and only shorten the text sequence in the following way:

\[
(x_{1:k}, a_{1:r}) = \begin{cases} 
    x_1 \ldots x_k [SEP] a_1 \ldots a_r [SEP] [CLS] & \text{if } k + r + 3 < t \\
    x_1 \ldots x_{t-r} [SEP] a_1 \ldots a_r [SEP] [CLS] & \text{if } k + r \geq t 
\end{cases}
\]

(2)
Similarly like BERT, XLNet can model multiple segments by treating the concatenation of two segments as one sequence to perform permutation language modeling. But XLNet doesn’t contain the NSP training language model anymore.

**D. XLNet Layer**

XLNet is a multi-layer Transformer encoder based on the work of BERT, and consists of 12 (or 24) layers (Transformer-XL blocks) and 12 (or 16) self-attention heads. The XLNet layer can output a sequence of embeddings for each token in the input or an aggregated embedding vector that pools the input. Let $H = \{H_1, H_2, \ldots, H_t\} \in R^{t \times d_h}$ be the last hidden representation of the input $E = \{E_1, E_2, \ldots, E_t\}$ and denoted as:

$$H = XLNet((E_1, E_2, \ldots, E_t)), \quad H \in R^{t \times d_h} \quad (3)$$

where $d_h$ is the embedding dimension with the default value of 768 or 1024.

Most of the related works usually take the hidden vector $H_{[CLS]} \in R^{d_h}$ of the special dummy symbol [CLS] as the global aggregated representations of the input, e.g., $H_1$ in Formula 3. However, we believe that the $H$ sequence might contains intrinsic spatial and hierarchical information that would be informative for the model. As a result, we further use a capsule network to extract such local features from $H$.

**E. CapsNet Layer**

The CapsNet layer uses a capsule network to automatically learn child-parent (or part-whole) relationships from the output embedding sequence $H$ of the XLNet layer that are viewed as input capsules, and composes them into the representations with pre-determined size. Formally, for the child capsule $H_i$, each parent capsule $V_j$ aggregates all the incoming messages $u_j$ from each child capsule and squares $u_j$ to $||u_j|| \in (0, 1)$ via the squash function [28] as follows:

$$u_j = squash(u_j) = \frac{||u_j||^2}{1 + ||u_j||^2} \frac{u_j}{||u_j|| + e^{-9}} \quad (4)$$

$$u_j = \sum_{i=0}^{t-1} c_{ij} H_{iji} \quad (5)$$

where $c_{ij}$ is the coupling coefficient that can be viewed as the voting weight on the information flow from the $i$ child capsule to the $j$ parent capsule. $H_{iji}$ denotes the information be transferred from $H_i$ into $V_j$.

$$H_{iji} = ReLU(W_{ij}^T H_i) \quad (6)$$

We take a single layer feed forward neural network with a ReLU activation function as the transformation to obtain the information $ReLU(W_{ij}^T H_i)$ from each child capsule $H_i$ and $W_{ij} \in R^{d_i \times (cn \times d_h)}$ is the transformation matrix corresponding to the position $j$ of the parent capsule, where $cn$ is the number of capsules and $d_h$ is the dimension of the neural vector in each capsule.

The dynamic routing process [40], [41] is implemented via an EM iterative process of refining the coupling coefficient $c_{ij}$, which measures proportionally how much information would be transferred from $H_i$ to $V_j$.

At iteration $n$, the coupling coefficient $c_{ij}$ is computed via a softmax function for ensuring that all the information from the child capsule $H_i$ can be transferred to the parent:

$$e_{ij}^n = softmax(b_{ij}^{n-1}) = \frac{exp(b_{ij}^{n-1})}{\sum_k exp(b_{ik}^{n-1})} \quad (7)$$

$$b_{ij}^n = b_{ij}^{n-1} + H_{iji}^n V_j^n \quad (8)$$

where $b_{ij}^0$ is simply a temporary value that will be iteratively updated with the value $b_{ij}^{n-1}$ of the previous iteration and the scalar product of $H_{iji}$ and $V_j^n$, which is essentially the similarity between the input to the capsule and the output from the capsule. Particularly, $b_{ij}^0$ is initialized with 0. The coefficient depends on the location and type of both the child and the parent capsules, which is the iterative refinement of $b_{ij}$. The pseudo-codes for dynamic routing algorithm are presented as follows:

**Algorithm 1 DynamicRouting**

**Input**: Input capsules $H = \{H_1, \ldots, H_t\}$, iteration number $n$  

**Output**: Capsules $V = \{V_1, \ldots, V_{cn}\}$

**Process**:  
1: Initialize $b_{ij}^0 \leftarrow 0$  
2: for $t = 1$ to $n$ do  
3: for $\forall c_{ij}^t$: $c_{ij}^t = softmax(b_{ij}^{t-1})$ /*Formula 7*/  
4: for $\forall H_{iji}^t$: $H_{iji}^t = W_{ij}^T H_i^t$ /*Formula 6*/  
5: for $\forall u_j^t$: $u_j^t = \sum c_{ij}^t H_{iji}^t$ /*Formula 5*/  
6: for $\forall V_j^t$: $V_j^t = squash(u_j^t)$ /*Formula 4*/  
7: for $\forall b_{ij}^t$: $b_{ij}^t = b_{ij}^{t-1} + H_{iji}^t V_j^n$ /*Formula 8*/  
8: end for  
9: return $\{V_1, \ldots, V_{cn}\}$

After the source input is encoded into $cn$ capsules, we concatenate these capsules into a vector:

$$V^* = concat(V_1, V_2, \ldots, V_{dc}) \quad (9)$$

The vector $V^* \in R^{dh}$ can be regard as the representations of the source sequence that contains its part-whole relationship features.

**F. Output Layer**

Before final prediction, we explore several strategies to pool the global aspect-specific representation $H_{[CLS]}$ and local feature representation $V^*$ into an aggregated vector $O \in R^d$:

$$O = merge(H_{[CLS]}, V^*) \quad (10)$$

We use four different pooling strategies, sum, maximum, concatenation and average, which are denoted as add, max, con and avg for short respectively. Let $\theta$ be the set of all trainable parameters for XLNetCN. An additional output layer with a softmax classifier turns the output $O$ into the conditional probability distributions.
4. EXPERIMENTS

In this section, we evaluate our model on the widely-used benchmark dataset SemEval-2014 Task 4; which consists of Restaurant reviews and Laptop reviews [1]. SemEval 2014 Task 4 totally contains four different subtasks and we only consider two of them in the following experiments, including Subtask 2: aspect term polarity and Subtask 4: aspect category polarity. Subtask 2 aims to determine the polarity of each aspect term (e.g. positive, negative, neutral or conflict), given a set of aspect terms within a sentence. Subtask 4 aims to determine the polarity of each aspect category, given a set of pre-identified aspect categories (e.g. {food, service}).

Both original Laptop and Restaurant datasets contain sentences with multiple marked aspect terms or aspect categories that each have a 3-label sentiment polarity. For the sake of comparability with [8], [26], [42], we discard those reviews with the polarity of “conflict”. See detailed statistics about the number of samples for both datasets in Table 2, in which the columns of “Max Sequence Length” and “Max Aspect Length” denote the maximum sequence length and maximum aspect lengths tokenized by SentencePiece respectively.

From Table 2, we can see that both datasets only have a small number of training examples and their sequences are far less than 512 tokens. So, we use their maximum sequence length as the default value to preserve their complete meanings. While for the aspects, since the average aspect lengths in both datasets are about 1.8 for Subtask 2 and 2.4 for Subtask 4, we only take the first five tokens of the aspect sequence in the following experiments.

We use the pre-trained cased XLNetbase as a base for fine-tuning for the sake of computation costs and compar-

### TABLE 2. Statistics about sample numbers of SemEval 2014.

| Task   | Dataset    | Negative Train | Negative Test | Neutral Train | Neutral Test | Positive Train | Positive Test | Average Length | Max Sequence Length | Average Aspect Length | Max Aspect Length |
|--------|------------|----------------|---------------|---------------|--------------|----------------|---------------|-----------------|---------------------|---------------------|-------------------|
| Subtask 2 | Laptop     | 866            | 128           | 460           | 169          | 987            | 341           | 22             | 88                  | 1.8                 | 11                |
|         | Restaurant | 805            | 196           | 633           | 196          | 2164          | 728           | 22             | 85                  | 1.8                 | 24                |
| Subtask 4 | Restaurant | 839            | 222           | 500           | 94           | 2179          | 657           | 17             | 85                  | 2.4                 | 5                 |

\[ P(y_i | O, \theta) \] over the category set \( y = \{ y_1, \ldots, y_m \} \) or \( y = \{0, 1\} \) in the following ways:

\[
P(y_i | O, \theta) = \text{softmax}(OW^T) = \frac{\exp(P(y_i | O, \theta))}{\sum_{j=1}^{m} \exp(P(y_j | O, \theta))}
\]

where \( W \in \mathbb{R}^{m \times d} \) is the task-specific parameter matrix and \( m \) is the number of labels.

The label with the largest \( y_s = \text{argmax}(P(y_i | O, \theta)) \) value would be selected as the predicted result and a standard calculation loss \( J(x, \theta) \) is achieved based on the canonical cross-entropy function.

We fine-tune all parameters from XLNetCN as well as \( W \) jointly by maximizing the log-probability of the correct label.

### IV. EXPERIMENTS

In this section, we study how the iteration number \( n \) and the number of capsules affect the performance of capsule network on Subtask 2. We only consider XLNetCN-NS-A-Add and XLNetCN-AS-AP models in this experiment. Figure 2 and 3 shows the accuracy comparisons of 2-6 iterations in the dynamic routing process and the pair denotes \((cn, de)\) that satisfies \(cn \times de = dh\).

From the above results, we found that the performance of these two XLNetCN models is unsteady on the conditions of different iteration numbers and capsule number settings. But generally, they both reach the best when iteration is set to 3 and the pair \((cn, de)\) is \((24, 32)\). So, in the following experiments we will take 3 as the default value for the iteration.

- [A. EXPERIMENT 1: EFFECTS OF ITERATIVE ROUTING ANALYSIS](#a-experiment-1-effects-of-iterative-routing-analysis)

---

1. https://alt.qcri.org/semeval2014/task4
2. https://storage.googleapis.com/xlnet/released_models/cased_L-12_H-768_A-12.zip
TABLE 3. Results of XLNetCN on Subtask 2 with different auxiliary sentences and pooling strategies.

| Auxiliary Sentence | Pooling | Laptop | Restaurant |
|-------------------|---------|--------|------------|
|                   |         | Acc    | MF1        | Acc    | MF1        |
| single            | add     | 0.7912 | 0.7536    | 0.8143 | 0.6924    |
|                   | con     | 0.7884 | 0.7485    | 0.8179 | 0.7005    |
|                   | max     | 0.7727 | 0.7234    | 0.8161 | 0.7028    |
|                   | avg     | 0.7774 | 0.7252    | 0.7991 | 0.6625    |
| QA                | add     | 0.7931 | 0.7539    | 0.8135 | 0.7001    |
|                   | con     | 0.7868 | 0.7359    | 0.8134 | 0.6963    |
|                   | max     | 0.7799 | 0.7343    | 0.8062 | 0.6839    |
|                   | avg     | 0.7799 | 0.7382    | 0.817  | 0.7071    |
| QA-A              | add     | 0.7958 | 0.7551    | 0.8491 | 0.7663    |
|                   | con     | 0.8072 | 0.768    | 0.8446 | 0.7561    |
|                   | max     | 0.8009 | 0.7615    | 0.8554 | 0.7699    |
|                   | avg     | 0.7994 | 0.7561    | 0.8661 | 0.7595    |
| NS-A              | add     | 0.7962 | 0.7564    | 0.8634 | 0.7883    |
|                   | con     | 0.7937 | 0.7477    | 0.8607 | 0.7786    |
|                   | max     | 0.8056 | 0.768    | 0.8554 | 0.7669    |
|                   | avg     | 0.7978 | 0.7617    | 0.858  | 0.7767    |
| AS-AP             | add     | 0.8741 | 0.8577    | 0.9045 | 0.8922    |
|                   | con     | 0.8673 | 0.8508    | 0.8943 | 0.8811    |
|                   | max     | 0.8704 | 0.8534    | 0.8938 | 0.8802    |
|                   | avg     | 0.873  | 0.8571    | 0.8938 | 0.8802    |
| AS-P              | add     | 0.8592 | 0.8433    | 0.8766 | 0.8685    |
|                   | con     | 0.8621 | 0.8448    | 0.8738 | 0.8752    |
|                   | max     | 0.8532 | 0.8358    | 0.8711 | 0.8537    |
|                   | avg     | 0.8511 | 0.8307    | 0.8744 | 0.8576    |
| AW-AP             | add     | 0.8636 | 0.8454    | 0.8938 | 0.8801    |
|                   | con     | 0.8662 | 0.8499    | 0.8982 | 0.8851    |
|                   | max     | 0.8589 | 0.84    | 0.8896 | 0.8756    |
|                   | avg     | 0.8657 | 0.849    | 0.8935 | 0.8799    |
| AW-P              | add     | 0.8584 | 0.8401    | 0.8765 | 0.8607    |
|                   | con     | 0.8495 | 0.8305    | 0.8747 | 0.8582    |
|                   | max     | 0.8511 | 0.8324    | 0.8738 | 0.8567    |
|                   | avg     | 0.8547 | 0.8336    | 0.867  | 0.8501    |

number $n$, and (24, 32) for $(cn, dc)$ without further explicit mention.

B. EXPERIMENT 2: EFFECTS OF AUXILIARY SENTENCE AND POOLING STRATEGY ANALYSIS

In this section, we explore the performance of various XLNetCN models under different situations of auxiliary sentences and pooling strategies so as to confirm the effectiveness of XLNetCN. We summarize the experimental results in Table 3 and boldface the best scores. For the sake of comparisons, we use “single” to denote that the XLNetCN model only uses the text sequence without auxiliary sentence.

From the results in Table 3, we can see that:

1. The performance of various XLNetCN-QA models is comparable to the corresponding XLNetCN-single models. Although the former consists of a sequence-pair in its inputs, there doesn’t contain any aspect or polarity messages in the auxiliary sentence, which as a result can’t provide any help for the training of XLNetCN. For all XLNetCN-QA-A models, they outperform both XLNetCN-single and XLNetCN-QA due to the including of the aspect tokens, which allows XLNet to learn the sequence-aspect relationship and yield aspect-specific representations. We can see similar results from XLNetCN-NS-A whose auxiliary sentence only includes the aspect tokens. XLNetCN-NS-A gains better improvements on Restaurant over Laptop due to the larger number of training samples, which ensures more fully pre-training of XLNet.

2. When using other auxiliary sentences except for QA, the performance of XLNetCN is improved significantly, which suggests the effectiveness of auxiliary sentences for improving the aspect-aware and task-agnostic abilities. From the comparisons between NS-A and QA, we can see that the Acc and MF1 values of NS-A are improved obviously due to the including of the aspect, which demonstrates that XLNet can better learn the aspect-specific representations for the sequence and confirms the importance of catching the relationships between the sentence and the aspect.

3. From the results of AS-P, AW-P, NS-A and QA, the performance of XLNetCN, especially MF1 score, is improved significantly after converting the task from original 3-label to 3-label. According to our analysis, there might include at least two reasons. Firstly, although AS-P and AW-P don’t contain aspect tokens in their auxiliary sentence, the polarity information and the converted task allow the model to learn the relations between the aspect and sequence even without NSP training objective in XLNet, since it concatenates them together and all possible variants of masking are fully exploited. Secondly, the increasing number of training data allows the model to be more fully trained and learn more task-agnostic knowledge. The converting task into 2-label is also helpful for decreasing the uncertainties of some noising labels in multi-label task, e.g. neural label that is often confused with positive or negative label, which can be confirmed from the greatly increasing MF1 values of AS-P and AW-P over NS-A. Similar conclusions can also be drawn from the experimental results of BERT-pair in [27]. As a results, we see that constructing suitable auxiliary sentence and converting the task into a binary classification one can provide an effective and feasible way for improving the aspect-specific and task-agnostic learning abilities.

4. For each XLNetCN that contains auxiliary sentence and is converted into 2-label task, e.g. AS-AP, AS-P, AW-AP and AS-P, it achieves significant improvements over XLNetCN-single and other XLNetCN models that only include auxiliary sentence, which obviously confirms the importance of converting the task. From the results of AS-P and AS-AP, as well as AW-P and AW-AP, XLNetCN achieves better performance obviously when the auxiliary sentence contains both the aspect and polarity information, rather than simply the polarity information, which confirms again the effectiveness and importance of explicitly modeling the
relations between the aspect and the text sequence. From their great improved performance over NS-A, we can see that converting the task from 3-label into 2-label one is critical. Generally, the performance of AW-P and AW-AP is not so good as the corresponding AS-P and AS-AP, but the differences are minor. According to our analysis, although the auxiliary sentence structures of AW-P and AW-AP are similar with those of AS-P and AS-AP for having polarity label or both the aspect and polarity information, the auxiliary sentence in AS-P or AS-AP has more complete syntactic structures such that XLNet can learn more accurate contextualized meanings. We can see similar results in [27].

(5) Among four different pooling strategies, the adding strategy achieves best results in AS-AP on both two datasets. The main reasons we think might lie in the facts that there usually exists some informative words that have important impacts on the final prediction of the sentiment classification task. The adding strategy allows the model to enhance the impacts of these informative words, e.g. sentiment-related words, with the help of capsule network that extracts local n-gram information from the text sequence.

From the above results, we can see that XLNetCN-AS-AP-add gains best performance on both Laptop and Restaurant datasets. So, in the following experiments we mainly use it as our representative model for comparisons.

C. EXPERIMENT 3: EFFECTS OF CAPSULE NETWORK
In this section, we firstly conduct experiments on Subtask2 to explore the effectiveness of capsule network in learning the local features and then demonstrate its advantages over other neural network. Let XLNetbase denote the canonical model that doesn’t use capsule network and directly takes $H[CLS]$ for downstream task. For fair comparisons, most of hyper-parameters for XLNetbase and XLNetCN are remained consistently except for the settings about capsule networks. The experimental results are illustrated in Table 4.

From the results in Table 4, we can see that capsule network can slightly improve the performance of the XLNetbase model either on the canonical situation with one sequence as input or on the situations with auxiliary sentence, which demonstrates that capsule network can improve the performance of XLNet by further exploiting the local and spatial features.

In the following experiment, we continue to compare capsule network with other classical neural encoders, e.g. LSTM and CNN, by simply replacing the capsule network in the CapsNet layer with LSTM or CNN respectively. For LSTM, we only consider the forward direction and take the last hidden state vector as the output with the dimension of 768. For CNN, we consider two different encoders with the convolutional kernel of [2] and [2, 3, 4] respectively. See Figure 4 for their accuracy results on Subtask 2.

From the results in Figure 4, we can see that capsule network outperforms both LSTM and CNN obviously almost all the time.

D. EXPERIMENT 4: COMPARISONS WITH BERT
In this section, we compare XLNetCN with the corresponding BERT-based works, denoted as BERTCN, by simply replacing the XLNet with BERT in the XLNet layer on the Subtask 2. See Table 5 for the experimental results.

From the results in Table 5, we can observe that both XLNet and BERT demonstrate their great potentials on the ABSA task. Generally, XLNet outperform BERT obviously on the Laptop dataset. Since XLNetbase can better tackle the pretrain-finetune discrepancy problem and is pre-trained over a much larger corpus than BERTbase, it outperforms than BERTbase on the Laptop data when training data is limited such that full fine-tuning can’t be ensured. While for the Restaurant dataset, they perform differently and sometimes BERTCN outperforms XLNet. The main reasons we think might be mainly due to the larger number of training data in the Restaurant dataset than Laptop, which as a result enables both BERT and XLNet to be more fully trained for learning task-specific knowledge. After constructing auxiliary sentence and converting the 3-label task into the 2-label one, BERTCN and XLNetCN achieve competitive results on AS-AP with minor difference, which suggests that both of them have their own advantages as long as enough training

| Model          | Laptop Acc | Laptop MF1 | Restaurant Acc | Restaurant MF1 |
|----------------|------------|------------|----------------|-----------------|
| XLNetbase      | 0.7915     | 0.7533     | 0.8170         | 0.6996          |
| XLNetCN-single-add | 0.7929     | 0.7562     | 0.8191         | 0.7021          |
| QA             | 0.7900     | 0.7506     | 0.8125         | 0.6961          |
| QA-A           | 0.7994     | 0.7568     | 0.8545         | 0.7762          |
| NS-A           | 0.7868     | 0.7412     | 0.8985         | 0.8586          |
| AS-AP          | 0.8715     | 0.8542     | 0.8952         | 0.882           |
| AS-P           | 0.8568     | 0.8394     | 0.8714         | 0.8546          |
| AW-AP          | 0.8662     | 0.8492     | 0.8896         | 0.8756          |
| AW-P           | 0.8521     | 0.8328     | 0.8705         | 0.8535          |
| QA-add         | 0.7931     | 0.7539     | 0.8135         | 0.7001          |
| QA-A-add       | 0.7958     | 0.7551     | 0.8491         | 0.7663          |
| NS-A-add       | 0.7962     | 0.7564     | 0.8634         | 0.7883          |
| AS-AP-add      | 0.8741     | 0.8577     | 0.9045         | 0.8922          |
| AS-P-add       | 0.8592     | 0.8433     | 0.8766         | 0.8685          |
| AW-AP-add      | 0.8636     | 0.8454     | 0.8938         | 0.8801          |
| AW-P-add       | 0.8584     | 0.8401     | 0.8765         | 0.8607          |
TABLE 5. Comparisons of XLNetCN and BERTCN on Subtask 2.

| Models                | Laptop | Restaurant |
|-----------------------|--------|------------|
|                       | Acc    | MF1        | Acc    | MF1        |
| BERTbase              | 0.7571 | 0.6965     | 0.8071 | 0.7054     |
| XLNetbase             | 0.7915 | 0.7533     | 0.8170 | 0.6996     |
| BERTCN-single-add     | 0.7633 | 0.7076     | 0.8116 | 0.7087     |
| XLNetCN-single-add    | 0.7929 | 0.7562     | 0.8191 | 0.7021     |
| BERTCN, Add           |        |            |        |            |
| QA                    | 0.7649 | 0.7198     | 0.8080 | 0.7041     |
| QA-A                  | 0.7759 | 0.7305     | 0.8554 | 0.7838     |
| NS-A                  | 0.7853 | 0.7452     | 0.8509 | 0.7771     |
| AS-AP                 | 0.8576 | 0.8387     | 0.904  | 0.8916     |
| AS-P                  | 0.8375 | 0.8167     | 0.8801 | 0.8642     |
| AW-AP                 | 0.8548 | 0.8359     | 0.8969 | 0.8901     |
| AW-P                  | 0.8297 | 0.8093     | 0.8646 | 0.847      |
| XLNetCN, Add          |        |            |        |            |
| QA                    | 0.7931 | 0.7539     | 0.8135 | 0.7001     |
| QA-A                  | 0.7958 | 0.7551     | 0.8491 | 0.7663     |
| NS-A                  | 0.7962 | 0.7564     | 0.8634 | 0.7883     |
| AS-AP                 | 0.8741 | 0.8577     | 0.9045 | 0.8922     |
| AS-P                  | 0.8592 | 0.8433     | 0.8766 | 0.8685     |
| AW-AP                 | 0.8636 | 0.8454     | 0.8938 | 0.8801     |
| AW-P                  | 0.8584 | 0.8401     | 0.8765 | 0.8607     |

data is available. Additionally, we can see that the removing of the NSP training objective in XLNet doesn’t produce too much effects on the final performance about the ABSA task.

E. EXPERIMENT 5: COMPARISONS WITH RELATED WORKS

In this section, we first compare the performance of our model with some recently typical works on the Subtask 2. The selected typical baseline models are briefly introduced as follows:

- **MemNet** [29]. A deep memory network that explicitly captures the importance of each context word when inferring the sentiment polarity of an aspect, and such importance degree and text representation are calculated with multiple computational layers. Each computational layer is a neural attention model over an external memory.

- **RAM** [8]. A multiple-attention based model that non-linearly combines the results of multiple attentions with a recurrent neural network, which provides a tailor-made memory for different opinion targets of a sentence.

- **IAN** [30]. An LSTM-based model that uses the interactive attention networks (IAN) to interactively learn attentions in the contexts and targets, and generate the representations for targets and contexts separately, which finally yields context representation for sentiment classification.

- **DSAN-mean** [6]. A dependency subtree attention network (DSAN) that takes the syntax distances between the aspect and context words into consideration and utilizes a bidirectional GRU (Gated Recurrent Unit) network and dot-product attention mechanism to generate aspect-related representations for the sentence.

- **DTDA-mean** [7]. A bidirectional GRU model that takes dependency tree and distance attention (including syntactic distance and relative distance) to better capture the relations between sentences and aspects.

- **BERTADA-base**. The BERTbase model in [42] that is further trained on a domain-specific dataset and evaluated on the test set from the same domain.

- **XLNetADA-base**. The XLNetbase model in [42] like BERTADA-base except for using XLNet.

- **AEN-BERT** [31]. Attentional encoder network that uses pre-trained BERT to generate word embeddings of the sequence and the aspect respectively, which are then fed to an attentional encoder layer that consists of multi-head attention and point-wise convolution transformation for learning their relationships.

- **TD-GAT-BERT(5)** [17]. A target-dependent graph attention network (TD-GAT) that explicitly utilizes the dependency relationship among words and BERT representations with five layers.

- **LCF-ATEPC** [43]. A multi-task learning model for ABSA based on domain-adapted BERT by equipping the capability of extracting aspect term and inferring aspect term polarity synchronously.

- **SDGCN-BERT** [44]. A model that uses GCN to capture the sentiment dependencies between multi-aspects in one sentence and also use BERT as feature representations.

We report the Acc and MF1 metrics and boldface the score with the best performance across all models. Table 6 illustrates their results, where “-” means not reported and the results of the baselines come from their reported works respectively.

From the results in Table 6, we can see:

1. On both Laptop and Restaurant, XLNetbase and XLNetCN-single-add significantly outperform those feature-based methods that use typical LSTM or GRU models, such as TD-LSTM, MemNet, RAM, ATAE-LSTM, IAN, DTDA-sum and DTDA-mean. Without employing any additional attention mechanisms or dependency tree, XLNetbase achieves satisfactory results by simply using a softmax classifier followed by a XLNet model, which as a result demonstrates the obvious advantages of pre-trained language model over traditional word vectors, i.e. word2vec, Glove 1.8M and GloVe 2.2M, et al.

2. Compared with BERTADA-base and XLNetADA-base that both are trained on a domain-specific dataset and evaluated on the test set from the same domain, XLNetbase and XLNetCN-single-add achieve underperformance due to the lack of pre-training on domain-specific dataset. AEN-BERT, TD-GAT-BERT(5) and LCF-ATEPC all use BERT as the feature representations and the improvements of performance than XLNetbase and XLNetCN-single-add suggests the importance of explicit modeling the relationship between the sequence and the aspect.
TABLE 6. Experimental results on Subtask 2.

| Model            | Laptop Acc | Laptop MF1 | Restaurant Acc | Restaurant MF1 |
|------------------|------------|------------|----------------|----------------|
| MemNet           | 0.7033     | 0.6409     | 0.7816         | 0.6583         |
| RAM              | 0.7449     | 0.7135     | 0.8023         | 0.7080         |
| IAN              | 0.7210     | -          | 0.7810         | -              |
| DSAN-mean        | 0.7461     | 0.7052     | 0.7964         | 0.7136         |
| DTDA-mean        | 0.7471     | 0.7089     | 0.8008         | 0.7281         |
| BERTADA-base     | 0.7769     | 0.7260     | 0.8492         | 0.7693         |
| XLNetADA-base    | 0.7989     | 0.7778     | 0.8584         | 0.7835         |
| ASR-BERT         | 0.7993     | 0.7631     | 0.8312         | 0.7376         |
| TD-GAT-BERT(5)   | 0.8010     | -          | 0.8280         | -              |
| LCF-ATEPC        | 0.8150     | 0.7775     | 0.8625         | 0.8026         |
| SDGCN-BERT       | 0.8135     | 0.7834     | 0.8357         | 0.7467         |
| XLNetbase        | 0.7915     | 0.7533     | 0.8170         | 0.6996         |
| XLNetCN-single-add | 0.7929   | 0.7562     | 0.8191         | 0.7021         |
| XLNetCN-AS-P-add | 0.8592     | 0.8433     | 0.8766         | 0.8685         |
| XLNetCN-AP-P-add | 0.8584     | 0.8401     | 0.8765         | 0.8607         |
| XLNetCN-AP-ADD   | 0.8636     | 0.8454     | 0.8938         | 0.8801         |
| XLNetCN-AS-ADD   | 0.8741     | 0.8577     | 0.9045         | 0.8922         |

TABLE 7. Accuracies of BERT-pair and XLNetCN On Subtask 4.

| Model                | 4-way Acc | 3-way Acc | binary Acc |
|----------------------|-----------|-----------|------------|
| BERT-pair-QA-M       | 0.8520    | 0.8930    | 0.9540     |
| XLNetBase-QA-A       | 0.8638    | 0.8942    | 0.9587     |
| XLNetCN-QA-A-add     | 0.8668    | 0.8998    | 0.9598     |
| BERT-pair-NLI-M      | 0.8510    | 0.8870    | 0.9440     |
| XLNetbase-NS-A       | 0.8589    | 0.8911    | 0.9499     |
| XLNetCN-NS-A-ADD     | 0.8620    | 0.8972    | 0.9522     |
| BERT-pair-NS-A-B     | 0.8590    | 0.8990    | 0.9560     |
| XLNetbase-AS-AP      | 0.8625    | 0.9068    | 0.9594     |
| XLNetCN-AS-AP-add    | 0.8705    | 0.9131    | 0.9621     |
| BERT-pair-NL-I-B     | 0.8460    | 0.8870    | 0.9510     |
| XLNetbase-AW-AP      | 0.8508    | 0.8910    | 0.9567     |
| XLNetCN-AW-AP-add    | 0.8527    | 0.8912    | 0.9574     |

(3) Compared with AEN-BERT, TD-GAT-BERT(5), LCF-ATEPC and SDGCN-BERT, all sequence-pair XLNetCN models achieve significant improvements by simply converting the task into 2-label one. LCF-ATEPC also contains the aspect in the auxiliary sentence as a sentence-pair to yield aspect-specific representations, but it doesn’t convert the task. So, we believe that the converting of the task is critical.

In the following experiment, we continue to compare our XLNetCN with BERT-pair [27] on Subtask 4, which is closed to our work for constructing auxiliary sentence. Similar like BERT-pair, we replace aspect tokens with both aspect and category tokens in the auxiliary sentence for Subtask 4. To facilitate comparisons, we put each XLNetCN with the corresponding similar BERT-pair model together in Table 7 and illustrate the experiment results, in which “-” means not reported.

From the results in Table 7, we can see that all XLNetbase models outperform their corresponding BERT-pair models with three sub-task settings, which confirms the superiority of XLNet over BERT with respect to ABSA task. Additionally, from the better performance of XLNetCN-AW-AP-add over XLNetbase-AW-AP, we can see that capsule network is also useful for improving the performance of XLNetCN on Subtask 4.

V. CONCLUSION

In this paper, we propose a XLNet and capsule network-based model for ABSA aiming to address aspect-specific, local feature-agnostic and task-agnostic challenges. The main contributions are summarized as follows. (1) Constructing auxiliary sentence to model the sequence-aspect relation and generate global aspect-specific representations for enhancing aspect-specific ability and pre-training of XLNet so as to improve task-agnostic capability. (2) Employing a capsule network with the dynamic routing algorithm to extract the local and spatial hierarchical relations of the text sequence, and yield its local feature-aware representations, so as to address the local feature-aware challenge. (3) Conducting extensive experiments to demonstrate that XLNetCN outperforms significantly than the classical BERT, XLNet and traditional feature-based approaches on the two benchmark datasets of SemEval 2014 and achieve new state-of-the-art results. The results also show that adding additional neural network (e.g. capsule network) might be helpful for further improving the performance of pre-training language models on ABSA task.

In the future, we would like to further explore how to incorporate domain-related knowledge into XLNet with in-domain and cross-domain pre-training. We also want to extend our work with multi-task learning and evaluate the performance on other ABSA tasks.

REFERENCES

[1] M. Pontiki, D. Galanis, J. Pavlopoulos, H. Papageorgiou, I. Androutsopoulos, and S. Manandhar, “SemEval-2014 task 4: Aspect based sentiment analysis,” in Proc. 8th Int. Workshop Semantic Eval. (SemEval), 2014, pp. 27–35.
[2] K. Schouten and F. Frasincar, “Survey on aspect-level sentiment analysis,” IEEE Trans. Knowl. Data Eng., vol. 28, no. 3, pp. 813–830, Mar. 2016.
[3] B. Liang, Q. Liu, J. Xu, Q. Zhou, and P. Zhang, “Aspect-based sentiment analysis based on multi-attention CNN,” J. Comput. Res. Develop. Chin., vol. 54, no. 8, pp. 1724–1735, 2017.
[4] Y. Wang, M. Huang, X. Zhu, and L. Zhao, “Attention-based LSTM for aspect-level sentiment classification,” in Proc. Conf. Empirical Methods Natural Lang. Process., 2016, pp. 606–615.
[5] X. Wang and G. Chen, “Dependency-attention-based LSTM for target-dependent sentiment analysis,” in Proc. Chin. Nat. Conf. Social Media Process., 2017, pp. 206–217.
[6] Z. Ouyang, “Research on aspect-level sentiment classification based on dependency tree and attention network,” M.S. thesis, South China Univ. Technol., Guangzhou, China, 2018.
[7] J. Su, Z. Ouyang, and S. Yu, “Aspect-level sentiment classification for sentences based on dependency tree and distance attention,” J. Comput. Res. Develop., vol. 56, no. 8, pp. 1731–1745, 2019.
[8] P. Chen, Z. Sun, L. Bing, and W. Yang, “Recurrent attention network on memory for aspect sentiment analysis,” in Proc. Conf. Empirical Methods Natural Lang. Process., 2017, pp. 452–461.
[9] M. Zhang, Y. Zhang, and D. T. Vo, “Gated neural networks for targeted sentiment analysis,” in Proc. AAAI. Palo Alto, CA, USA: AAAI Press, 2016, pp. 3087–3093.
[10] J. Liu and Y. Zhang, “Attention modeling for targeted sentiment,” in Proc. 15th Conf. Eur. Chapter Assoc. Comput. Linguistics, Short Papers, 2017, pp. 572–577.
B. Huang and K. M. Carley, “Syntax-aware aspect level sentiment classification,” in Proc. Conf. Empirical Methods Natural Lang. Process., 2018, pp. 1091–1096.

W. Xue and T. Li, “Aspect based sentiment analysis with gated convolutional networks,” in Proc. 56th Annu. Meeting Assoc. Comput. Linguistics, Long Papers, 2018, pp. 2514–2523.

S. Wang, S. Mazumder, B. Liu, M. Zhou, and Y. Chang, “Target-sensitive memory networks for aspect sentiment classification,” in Proc. 56th Annu. Meeting Assoc. Comput. Linguistics, Long Papers, 2018, pp. 957–967.

C. Zhang, Q. Li, and D. Song, “Aspect-based sentiment classification with aspect-specific graph convolutional networks,” 2019, arXiv:1909.03477. [Online]. Available: http://arxiv.org/abs/1909.03477

B. Huang and K. M. Carley, “Syntactic-aware aspect level sentiment classification with graph attention networks,” 2019, arXiv:1906.02606. [Online]. Available: http://arxiv.org/abs/1906.02606

L. Zeyang, Y. Yujie, Y. Min, Z. Wei, G. Jun, and L. Yi, “A human-like semantic cognition network for aspect-level sentiment classification,” in Proc. AAAI, 2019, pp. 6650–6657.

X. Li, L. Bing, W. Zhang, and W. Lam, “Exploiting BERT for end-to-end aspect-based sentiment analysis,” 2019, arXiv:1910.00883. [Online]. Available: http://arxiv.org/abs/1910.00883

B. McCann, J. Bradbury, C. Xiong, and R. Socher, “Learned in translation: Contextualized word vectors,” in Proc. Adv. Neural Inf. Process. Syst., 2017, pp. 6294–6305.

M. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, “Deep contextualized word representations,” in Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol., Long Papers. Stroudsburg, PA, USA: Association for Computational Linguistics, 2018, pp. 2227–2237.

J. Howard and S. Ruder, “Universal language model fine-tuning for text classification,” 2018, arXiv:1801.06146. [Online]. Available: https://arxiv.org/abs/1801.06146

R. Aloe, N. Karrhiik, S. Tim, and S. Ilya, “Improving language understanding with unsupervised learning,” Tech. Rep., 2018.

D. Jacob, C. Ming-Wei, L. Kenton, and T. Kristina, “BERT: Pre-training of deep bidirectional transformers for language understanding,” 2018, arXiv:1810.04805. [Online]. Available: https://arxiv.org/abs/1810.04805

Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. Salakhutdinov, and Q. V. Le, “XLNet: Generalized autoregressive pretraining for language understanding,” 2019, arXiv:1906.08237. [Online]. Available: http://arxiv.org/abs/1906.08237

X. Hu, L. Bing, S. Lei, and Y. S. Philip, “BERT post-training for review reading comprehension and aspect-based sentiment analysis,” in Proc. NAACL, 2019, pp. 2324–2335.

D. Shi, C. Hai, L. Biao, and Q. Xipeng, “Utilizing bert for aspect-based sentiment analysis via constructing auxiliary sentence,” in Proc. NAACL, 2019, pp. 380–385.

S. Sera, F. Nicholas, and H. E. Geoffrey, “Dynamic routing between capsules,” in Proc. Adv. Neural Inf. Process. Syst., 2017, pp. 3856–3866.

D. Tang, B. Qin, and T. Liu, “Aspect level sentiment classification with deep memory network,” in Proc. Conf. Empirical Methods Natural Lang. Process., 2016, pp. 214–224.

D. Ma, S. Li, X. Zhang, and H. Wang, “Interactive attention networks for aspect-level sentiment classification,” in Proc. 26th Int. Joint Conf. Artif. Intell., Aug. 2017, pp. 4068–4074.

Y. Song, J. Wang, T. Jiang, Z. Liu, and Y. Rao, “Attentional encoder network for targeted sentiment classification,” 2019, arXiv:1902.09314. [Online]. Available: http://arxiv.org/abs/1902.09314

B. Huang and K. M. Carley, “Syntactic-aware aspect level sentiment classification with graph attention networks,” 2019, arXiv:1906.02606. [Online]. Available: http://arxiv.org/abs/1906.02606

K. Jungu and L. Jong-Hyeok, “Modeling inter-speaker relationship in XLMR for contextual spoken language understanding,” 2019, arXiv:1910.10479. [Online]. Available: http://arxiv.org/abs/1910.10479

S. Longpre, Y. Lu, Z. Tu, and C. DuBois, “An exploration of data augmentation and sampling techniques for domain-agnostic question answering,” 2019, arXiv:1912.02145. [Online]. Available: http://arxiv.org/abs/1912.02145

Y. Dong, Y. Fu, L. Wang, Y. Chen, Y. Dong, and J. Li, “A sentiment analysis method of capsule network based on BiLSTM,” IEEE Access, vol. 8, no. 1, pp. 37014–37020, 2020.

K. Zhang, M. Jiao, X. Chen, Z. Wang, B. Liu, and L. Liu, “SC-BiCapsNet: A sentiment classification model based on Bi-channel capsule network,” IEEE Access, vol. 7, pp. 171801–171813, 2019.

J. Su et al.: Enhancing ABSA With Capsule Network