Deep Learning-Based Frameworks for Aspect-Based Sentiment Analysis

Ashish Kumar and Aditi Sharan

Abstract Opinions are key influencers of almost all human practices. One can easily find a number of opinions about any product or services in the form of product reviews. These product reviews are available in a tremendous amount. It is not feasible or even impossible to go through each review and make a concise decision about any product. Aspect-based sentiment analysis (ABSA) comes as a solution to this problem. It gives an approach to examine online reviews and provides a summary based on these reviews. Machine learning techniques have been broadly utilized for ABSA. Recently with the evolution of processing power of computers and digitization of the society, deep learning is taking off. Deep learning methods produced state-of-the-art results in various NLP tasks without intensive feature engineering. In this chapter, we present an introduction about ABSA following a comprehensive overview of various deep learning models used in the field of ABSA.

Keywords Aspect-based sentiment analysis · Recurrent neural network · Long short-term memory · Convolution neural network · Natural language processing

1 Introduction

Because of the simple openness of the Web, people are often using Web portals to frame an opinion about a certain product, topic, and service. These online reviews expressed online opinions. These online opinions are valuable resources for decision making. The availability of enormous reviews does not ease our task; in fact, the complications are increased as it is not possible to read each and every review. In

A. Kumar (✉) · A. Sharan
School of Computer and Systems Sciences,
Jawaharlal Nehru University, New Delhi 110067, India
e-mail: ashishkumar2912@gmail.com; ashish29_scs@jnu.ac.in

A. Sharan
e-mail: aditisharan@mail.jnu.ac.in

B. Agarwal et al. (eds.), *Deep Learning-Based Approaches for Sentiment Analysis*, Algorithms for Intelligent Systems,
https://doi.org/10.1007/978-981-15-1216-2_6

order to make an opinion about a product based on its reviews, it is critical to analyze the sentiment associated with reviews.

Sentiment analysis on opinions can be done at various levels, viz. document, sentence, and aspect. Document and sentence levels deal with the identification of overall opinion in the designated document and sentence, respectively. But these levels ignore the fact that a document and sentence may talk about different features (aspects) of an entity. There is a need for extracting these aspects and their corresponding sentiment polarity. This process is called aspect-based sentiment analysis (in short ABSA).

ABSA is a task that involves various subtasks. Following are some of these subtasks [6].

1. Identification of aspect-terms
2. Identification of opinion-terms
3. Extraction of aspect-categories
4. Sentiment (Polarity) identification
5. Sentiment intensity identification
6. Sentiment shifters identification
7. Opinion holder and time identification
8. Generation of opinion tuple
9. Opinion summarization

An opinion can be expressed or understood in different ways. However, according to [16], an objective definition of opinion is given below:

Definition 1 (*Opinion*) An opinion is a quintuple.

$$(e_i, a_{ij}, s_{ijkl}, h_k, t_l) \tag{1}$$

Table 1 Example reviews

Reviews	Entity	Aspect-term	Aspect-category	Opinion-word
1. This laptop is great!	Laptop	–	–	Great
2. It is very overpriced and not very tasty	Restaurant	–	Food, Price	Overpriced, tasty
3. The pizza was pretty good and huge	Restaurant	Pizza	Food	Good, huge
4. it is the best service you will find in even the largest of restaurants	Restaurant	Service	Service	Best
5. It is not worth for that bucks	–	–	Price	Worth

In the problem of sentiment analysis (opinion mining), the relation between the items of tuples is extracted, for example, identifying the different aspects a_{ij} of an entity e_i, determining sentiment s_{ijkl} of an aspect a_{ij}, and finding the opinion holder h_k, who has expressed the opinion at time t_l. Sentiment s_{ijkl} can be positive, negative, or neutral, or can take any discrete value on a certain scale to define sentiment intensity.

Before proceeding to ABSA, it is important to objectively define the notion of aspect-terms and aspect-categories and some other related terms. There are various terms used in ABSA, i.e., entity, aspect-term, aspect-category, opinion-word, etc. Let us try to understand these terms with the help of some examples.

It is clear from Table 1 that reviews are regarding some entities like Laptop, Restaurants, etc. However, they may point out the sentiment about the entity as a whole (in Review 1) or about some features of the entity (in Reviews 2, 3, 4, 5). Different aspect-terms can be used to render an opinion about an aspect-category. Aspect-terms like *pizza*, *bagels*, etc., can be used to put an opinion about aspect-category Food.

Aspect-category is a generic notion/concept/property of an entity. The opinion is generally expressed about an aspect-category. Aspect-categories may be different in different domain. For an instance, let us take an example of Restaurant domain; then, the possible category list includes Service, Food, Price, and Ambience. Different aspect-terms may belong to an aspect-category. Aspect-term and aspect-category are two different things. In some cases, aspect-category may be explicitly mentioned by an aspect-term, (in Reviews 3 and 4). However, such cases are very few. An aspect-category is quite often a hypernym of an aspect-term. For example, Food is hypernym of pizza, but that is not always the case. Aspect-category can be implied implicitly also (in Reviews 2 and 5).

ABSA may be performed either on aspect-terms or on aspect-category. As aspect-terms are explicit, it is easier to identify these terms and relate them with the sentiment expressed. However, when we are performing ABSA on an entity, we may be interested in analyzing opinions about some well-defined aspects of the entity, as discussed earlier. Each of these represents an aspect-category. Thus, ABSA on aspect-categories is more appealing, but at the same time more demanding. When we speak about review sentences, a sentence may talk about a single aspect-category, however, that is not always the case. Some sentences may depict opinion about multiple categories also (in Review 1). This adds further challenges to ABSA.

With the accessibility of an enormous volume of data and increase in computational power of computers (GPUs) in the last decade, deep learning has become the first choice of the research community. Unlike traditional machine learning techniques that require intensive feature engineering and a separate model for classification, deep learning performs both tasks. It does feature engineering and classification with the help of input data only. Apart from good performance in image encouraging tasks, deep learning shows promising outcomes in various NLP tasks like named-entity recognition, text summarization, machine translation, sentiment analysis, etc.

This chapter highlights the diverse deep learning strategies for aspect-terms extraction, aspect-category detection, and sentiment polarity detection methods. This chapter shows the contribution and challenges of deep learning in ABSA. Chapter association is done in an accompanying manner. Section 2 describes the problem formulation in ABSA. Section 3 lighten up some useful observation/assumption by researchers in the field of ABSA. Section 4 talks about input representation for deep learning. In Sect. 5, introduction about some basic methods of deep learning is given. Section 6 highlights the different deep learning architectures used in ABSA. Finally, Sect. 7 finishes up the chapter.

2 Problem Formulation

As there is a lot of subjectivity involve in opinion mining and ABSA, it is important to formulate the problem of aspect-term extraction and aspect-category detection. This section formulates these two problems.

2.1 Aspect-Term Extraction

This task aims to extract explicit aspect expression presented in an online review. In most of the cases, ATE task can be examined as a sequence labeling problem where each review token is labeled to represent whether it is a piece of aspect-term or not. For this tagging process, popular BIO tagging scheme [22] is used generally, where B represents the beginning of aspect-term, I represents inside of aspect-term, and O is used for others that are not part of aspect-term.

For a given review sentence $x = (w^{<1>}, w^{<2>}, \dots, w^{<T>})$, the output is a sequence of labels $y = (y^{<1>}, y^{<2>}, \dots, y^{<T>})$, where $w^{<i>}$ represents word position in review sentence and each individual label $y^{<i>} \in \Sigma$; $\Sigma = \{B, I, O\}$. The problem can be considered as a multi-class classification problem with $|\Sigma|^T$ different classes.

2.2 Aspect-Category Detection

For a given predefined aspect-category set $C = \{c_1, c_2, c_3, \dots, c_k\}$, where C denotes the category label space with k possible categories and a review dataset $R = \{r_1, r_2, r_3, \dots, r_n\}$ containing n review sentences. The task of aspect-category detection can be formulated as a learning function $h : R \to 2^C$ from a multi-category training set $D = (r_i, Y_i) \mid 1 \leq i \leq n, Y_i \subseteq C$ is a set of category labels associated with r_i. For each unseen review $r \in R$, the aspect-category prediction function $h(\cdot)$ predicts $h(r) \subseteq C$ as the set of proper category label for r.

3 Observation/Assumption in ABSA

By conducting a survey on various articles regarding ABSA, we came up with the following observations/assumptions that are made by researchers. By considering these assumptions in mind while designing a model for ABSA, these assumptions can help to tackle the problem of ABSA in a very efficient way.

1. In ABSA, the context of words plays a very significant role. Words' location and how the words are interacting with each other matter a lot. A basic bag-of-words model is never again adequate, since all context information lost in the bag-of-words model [29]. To represent negative sentiment, generally negation of positive sentiment words is used. If we use bag-of-word, then it will be difficult to capture sentiment orientation in that case.
2. To construe the sentiment of a specific aspect in the review sentence, only some subsets of context words are required. Rather than focusing on full context, it is always beneficial to give attention to that subset of context words [28]. In the given review "*The price is reasonable although the service is poor.*" The context word *reasonable* is more important as compared to *poor* for aspect "price." Oppositely *poor* is more important as compared to *reasonable* for aspect "service."
3. Aspect-term should co-occur with some opinion-words that helps in determining the sentiment polarity on it [15]. For example, Given review "*I've eaten at many different Indian restaurants.*", contains no opinion-words. Hence, the word *restaurants* should not be extracted as aspect-term.
4. In addition to the word-context association, handling the connection between sentiment words and aspects can likewise be valuable for ABSA [34]. For example, many sentiment words are aspect-specific like 'delicious' and 'tasty' are used for aspect *food* only while 'cheap' and 'costly' are used only for *price*. Dependency parsing will be helpful in capturing the connection between aspect-terms and sentiment (opinion) words.
5. In sequential labeling, the predictions at the previous time-steps are useful clues for reducing the error space of the current prediction. For example, in the B-I-O tagging, if the previous prediction is O, then the current prediction cannot be I [15].
6. In a sentiment classification task, while using deep neural networks like LSTM, casting sentence portrayal alone does not perform well. Fusing target information into LSTM helps in improvement of sentiment classification accuracy.
7. For aspect-term extraction task, if we are using CNN then do not apply any max-pooling layer after convolution layers because a sequence labeling model needs good representations for every position and max-pooling operation mixes the representations of different positions, which is undesirable [32].

4 Input Representation

Inputs for the neural networks should be represented appropriately for the desired outputs. One should consider the good representation of the input, so that designated neural network learn good features. Word-embedding forms the basic building block for representing text as an input to a deep neural network in the field of NLP. Following are some of the ways to represent a sentence/words/aspect-terms.

1. Each word of review sentence is represented as an embedding vector. Word-embeddings [17, 19] are distributed representation of words in a vector space. Words and phrases are encapsulated to vectors of real numbers. Word-embedding represents word meaning from its surrounding context which is learned from large corpora.
2. To represent any sentence, one can take the bag-of-words approach by averaging the word vectors of the input sentence.
3. Each aspect-term can be represented using word-embeddings. For the aspect-term that consists only single word can be represented using the word-embedding of that word only. But for multi-word aspect-term, averaging of each word can be a way to represent multi-word aspect-term [28].
4. Some time it is better to represent words as a consolidation of word-embeddings and character-embeddings [10] to illustrate the effect of word morphology.
5. However, it has been observed that aspect-terms are generally nouns or noun phrases. So passing the POS information along with word information can be useful. If there are six pos-tags (noun, verb, adverb, adjective, preposition, and conjunction), these can be encoded as a six-dimensional vector. If the word-embedding dimension is 300, then word + POS features dimension will be 306 [20].
6. Since aspect plays a key role in ABSA, aspect information can be taken into account by concatenating aspect vector into sentence hidden representations and by additionally appending aspect vector into the input word vectors [30].

5 Concepts Related to Deep Learning

5.1 Word-Embeddings

Different ways of generating semantical associations are Linked Statistical Data (LSD), WordNet, Word-embeddings, etc. WordNet is an ontology representation of relationships of words which is constructed manually. WordNet is a symbolic representation, computing the similarity between words is limited to its hierarchical representation, whereas word-embeddings represent word meaning from its surrounding context words which are learned from large corpora. Word2vec (word-embeddings model) represents words in multi-dimensional vector space, this enables similarity calculation in terms of vector distance. Hence in terms of similarity calculation, word2vec is more effective than WordNet. In this technique, words and phrases are encapsulated to vectors of real numbers. Various strategies have been proposed to obtain word-embeddings [17–19]. In this architecture, neural network language model first learns word vectors, and then, n-grams neural network language model is trained on top of these distributed representations of words. Out of two models, skip-gram and continuous bag-of-words (CBOW) proposed by [18], Skip-gram model anticipates the context based on the current word. The following mathematical formulation needs to be maximized as an objective function for a given sequence of words $w_1, w_2, w_3, \ldots, w_T$.

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c, j\neq 0}\log p(w_{t+j} \mid w_t) \tag{2}$$

where w_t is the focused word and c is context window size. A larger value of c provides more samples for training which leads to high accuracy at the cost of training time. softmax function is used to calculate the probability $p(w_{t+j} \mid w_t)$.

$$p(w_o \mid w_I) = \frac{\exp({v'_{w_O}}^T v_{w_I})}{\sum_{w=1}^{W}\exp({v'_w}^T v_{w_I})} \tag{3}$$

where v_w and v'_w are the "input" and "output" vector representations of w, and W is vocabulary size. The cost of computing $\triangledown \log p(w_O \mid w_I)$ is in proportion to W, which is often large (10^5 to 10^7 terms). So, one of the following two approximations are used, hierarchical softmax and negative sampling, to solve it.

5.2 Long Short-Term Memory (LSTM)

LSTMs are an extraordinary sort of RNNs [12], which have been devised to capture long-term dependencies that are difficult to be handled by simple RNNs. LSTM resolves the problem of vanishing gradient by offering the concept of gates into their state dynamics. The fundamental architecture of RNN and LSTM is same, but hidden state activation computation function is different in LSTM. The memory of LSTM is called cell and is treated as a black box whose inputs are the past state $a^{<t-1>}$ and present input $x^{<t>}$.

LSTM has the ability to add new information, update, and remove previous information stored in the cell states. It uses the concept of gates to regulate the information flow at each time-step. A standard LSTM consists many gates like input (i), forget (f), and output (o) gates. The input gate decides what new information from the input need to be updated in the cell state. While the forget gate determines which information is not needed anymore. So that it can be erased from the cell state. Output gate chooses what to output conditioned on input and the content of the memory cell.

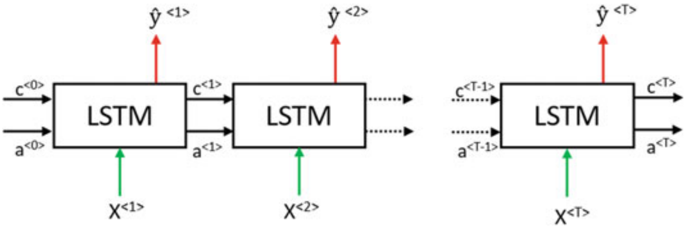

Fig. 1 LSTM Network

The LSTM cell at time-step t is formulated as below:

$$\begin{aligned}\hat{c}^{<t>} &= \tanh(W_c[a^{<t-1>}, x^{<t>}] + b_c)\\ i^{<t>} &= \sigma(W_i[a^{<t-1>}, x^{<t>}] + b_i)\\ f^{<t>} &= \sigma(W_f[a^{<t-1>}, x^{<t>}] + b_f)\\ o^{<t>} &= \sigma(W_o[a^{<t-1>}, x^{<t>}] + b_o)\\ c^{<t>} &= i^{<t>} * \hat{c}^{<t>} + f^{<t>} * c^{<t-1>}\\ a^{<t>} &= o^{<t>} * \tanh(c^{<t>})\end{aligned}$$

where σ is the logistic sigmoid function, $*$ is element-wise multiplication function, $\hat{c}$, c are memory cell states, a is activation (hidden) state, W_c, W_i, W_f, W_o are weight matrices, and b_c, b_i, b_f, b_o are bias vectors of different gates for input x. Like RNN, LSTM cells' combination is used to represent LSTM architecture (illustrated in Fig. 1).

For sequence tagging problems like aspect-term extraction if the input sequence is $x^{<1>}, x^{<2>}, \ldots, x^{<T>}$ and output sequence is $y^{<1>}, y^{<2>}, \ldots, y^{<T>}$, then while making a prediction for $y^{<3>}$, LSTM uses the information not only from $x^{<3>}$ but from $x^{<2>}$ and $x^{<1>}$ also. However, one weakness of LSTM is that it only uses the information that is earlier in the sequence to make a prediction. In order to decide whether or not the word is part of an aspect-term, it would be really useful to know not just information from the preceding words but to know information from the later words in the sentence.

5.3 Bi-directional Long Short-Term Memory (Bi-LSTM)

The idea behind the concept of Bi-LSTMs is to use the information associated with previous and future elements while generating output for the current element [23]. To capture all available information, two different networks are used (one for each direction) and results from both networks are combined to predict the final output. In simple words, bidirectional LSTMs are just two LSTMs assembled side by side [8]. The Bi-LSTM architecture is depicted in Fig. 2.

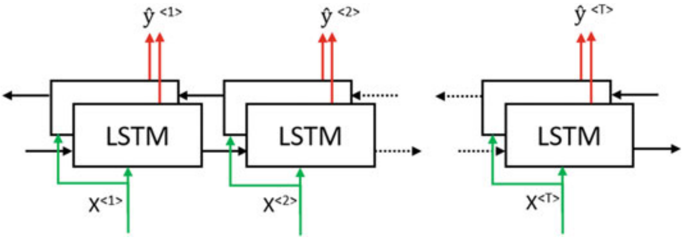

Fig. 2 Bi-LSTM network

5.4 RNN with Attention

Given a continuous input vector sequence $x^{<1>}, x^{<2>}, \ldots, x^{<T>}$, a standard RNN estimates the sequence of output of same length $y^{<1>}, y^{<2>}, \ldots, y^{<T>}$ using following equations [25].

$$\begin{aligned} h^{<t>} &= \tanh(W_a[a^{<t-1>}, x^{<t>}] + b_a) \\ \hat{y}^{<t>} &= \text{softmax}(W_y[a^{<t>}] + b_y) \end{aligned}$$

But this structure fails, for the problems when the input sequence and output sequence have a different length (in machine translation). A simple way is to map the inputs to a vector of fixed length and adopt this vector in an output sequence generation (encoder–decoder architecture).

First RNN that computes this fixed-size context vector c is called encoder and second RNN that computes output from the encoded fixed-size vector is called decoder. This overall architecture is called encoder–decoder. It computes the following conditional probability.

$$p(y^{<1>}, \ldots, y^{<T_y>}|x^{<1>}, \ldots, x^{<T_x>}) = \prod_{t=1}^{T_y} p(y^{<t>}|c, y^{<1>}, \ldots, y^{<t-1>}) \tag{4}$$

All the information of input sequence (sentence) is converted into a single vector. It must fully capture all the information (meaning) from the input sequence. This encoding process is senseless with a potentially very long input sequence. So the performance deteriorates when input sequence length increases.

To tackle this issue, attention mechanism was proposed by [2]. An attention mechanism was used to selective focus on sentence part while doing language translation. Using RNN, Eq. 4 can be modeled as

$$p(y^{<t>}|v, y^{<1>}, \ldots, y^{<t-1>}) = g(y^{<t-1>}, s^{<t>}, c^{<t>}) \tag{5}$$

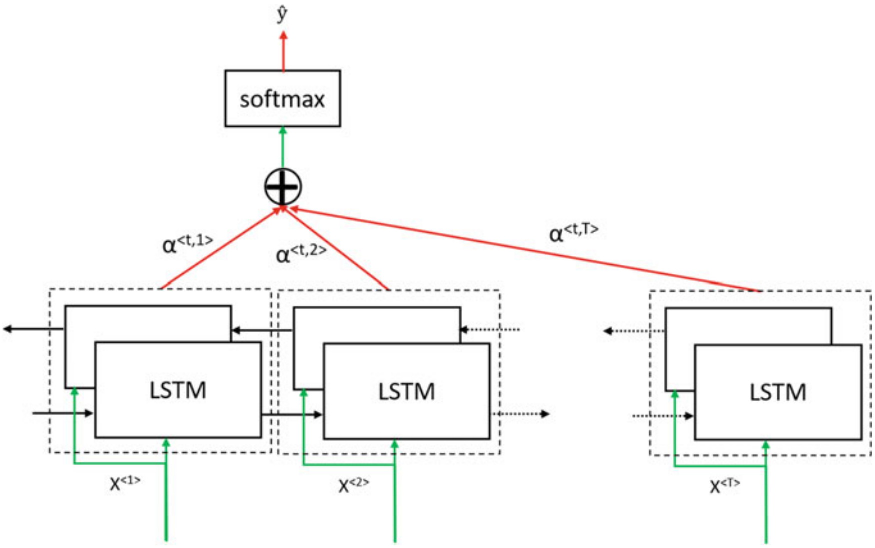

Fig. 3 Bi-LSTM with attention network

where g is a nonlinear function that outputs the probability of $y^{<t>}$ using a single directional RNN with state $s^{<t>}$. Context vector $c^{<t>}$ for each time-step will depend on features at different time-step $h^{<1>}, h^{<2>}, \ldots, h^{<T_x>}$ and attention parameter α that tells how much attention should pay on different features. Context vector $c^{<t>}$ is computed as weighted sum of features at different time-step $h^{<i>}$.

$$c^{<t>} = \sum_{t'=1}^{T_x} \alpha^{<t,t'>} h^{<t'>} \tag{6}$$

Attention weight should satisfy this

$$\sum_{t'=1}^{T_x} \alpha^{<t,t'>} = 1 \tag{7}$$

The weight $\alpha^{<t,t'>}$ of each annotation $h^{<t'>}$ is computed by

$$\alpha^{<t,t'>} = \frac{\exp(e^{<t,t'>})}{\sum_{t'=1}^{T_x} \exp(e^{<t,t'>})} \tag{8}$$

where $e^{<t,t'>} = a(s^{<t-1>}, h^{<t'>})$.
A simple way is to utilize a small neural network, to parameterize the alignment model a. Instead of generating a common fixed-length vector for each step of output. At

each step of output generation, decoder focuses on various sections of the sentence. Notably, the model grasps "what to attend" depending on the current input and previously generated outputs. Figure 3 shows a graphical illusion of attention-based network.

5.5 Convolution Neutral Network (CNN)

CNN has been used for the image classification task. More recently, with the development of word-embeddings, CNN produces state-of-the-art results in various NLP tasks also [14, 33]. Each sentence can be converted into a sentence matrix with the help of word-embeddings (word2vec, glove, etc.). If sentence length is s and word-embeddings dimension is d, then sentence matrix S dimension will become $s \times d$. Now sentence can be treated like an image a CNN can be easily applied over it. In CNN, a filter is convolved over the image an produce a feature map. If the image dimension is $n \times n$, filter size is $f \times f$ and feature map will be $(n - f + 1) \times (n - f + 1)$. Same in the case of text, convolution procedure comprises a filter $w \in \mathbb{R}^{hd}$, which is imposed on a window of h words to produce a new feature.

Following output sequence will be obtained by applying convolution operator over the sentence matrix S.

$$o_i = \mathbf{w} \cdot \mathbf{S}[i : i + h - 1]; \tag{9}$$

where $(\cdot)$ denotes dot-product and i has interval $[1, s - h + 1]$. After that feature map $c \in \mathbb{R}^{s-h+1}$ is introduced by adding a bias b and a nonlinear activation function f (e.g tanh).

$$c_i = f(o_i + b) \tag{10}$$

Finally, each feature map goes through a pooling function to generate a fixed-length vector. That vector can be used as an input in other neural network architecture. An illusion of sentence classification using CNN architecture is depicted in Fig. 4.

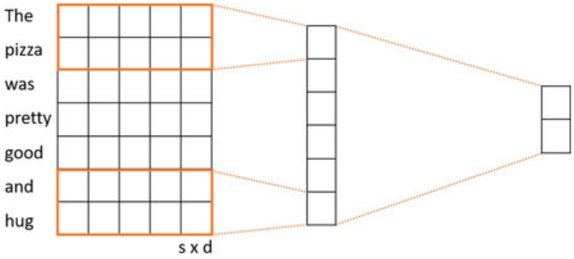

Fig. 4 CNN architecture for text

6 Deep Learning Architectures Used in ABSA

In this section summarizes the applications of deep learning approaches on different tasks of ABSA.

6.1 Sentiment Analysis

For sentence-level sentiment classification, [29] used basic CNN architecture in their sentiment model. The CNN model performs a convolution operation, using a filter, over the words of a sentence within a window size of h. This operation results in a feature map for the whole sentence. In next step, maximum value is extracted from the feature map using max-over-time pooling. The drawback of this approach is that it is aspect-agnostic. This approach works well only for sentences containing uni-sentiment.

In the approach by [4], authors also have done sentence-level sentiment analysis using CNN. In their approach, sentences were categorized into various groups depending on the total count of aspect-terms presented in the review sentence. Then, different CNN classifiers were trained separately on each sentence groups. Authors observed a significant increase in performance by separating sentences into different groups.

6.2 Aspect-Term Extraction

Chen et al. [4] used a neural network model consisting of bi-directional LSTM with CRF (Bi-LSTM-CRF) for extraction of aspect-terms presented in the reviews sentences.

Similar to the aforesaid architecture, to extract aspect-terms, a Bi-LSTM-CRF model was developed by Al-Smadi et al. [1]. This model used character-level features along with word-level features in the form of embeddings while predicting aspect-terms. Use of character-level embeddings benefited in analyzing of affixes without morphology analysis.

Giannakopoulos et al. [7] employed a Bi-LSTM architecture to extract features from the inputs. Randomly initialized character-embeddings and word-embeddings were passed as input to a Bi-LSTM layer. These character-embeddings were learned during the training process. Indirectly, a feature vector was created as a combination of pre-trained word-embeddings provided by fastText and character-based word-embeddings of each token of a sentence. This feature vector acts as input to main Bi-LSTM layer, which is utilized to extract features for the CRF layer. These features were generated by exploiting word morphology and sentence structure.

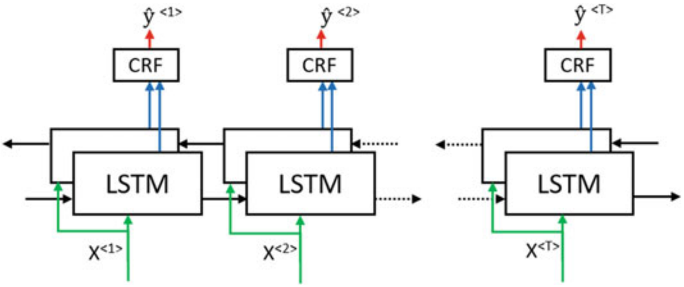

Fig. 5 Bi-LSTM-CRF network

Jebbara and Cimiano [13] used a hybrid neural architecture consisting of CNN and GRU for extraction of aspect-terms and opinion-terms. Deep CNN was able to capture local dependencies near a word of interest. Using the extracted features by CNN, GRU preserved the information over a long distance.

Wu et al. [31] have used a set of dependency relation-based rules for extraction of NP chunks. These NP chunks were treated as candidate aspect-terms and used to generate the noisy labeled dataset. Finally, a deep GRU network was trained to predict aspect-terms.

The basic Bi-LSTM-CRF model architecture is presented in Fig. 5.

6.3 Aspect-Category Extraction

In aspect model proposed by Wang and Liu [29], input was generated by averaging of word vectors of sentence and output was a probability distribution over aspects (where aspects were entity and attribute pair, E#A pair). They used a basic two-layer fully connected neural network for detection of aspects (aspect-categories).

Zhou et al. [34] also represented input sentence same as [29]. Then two separate neural networks were trained to learn shared features and aspect-specific feature. The first two-layer neural network was trained to classify the aspect-categories, and a hidden layer of this network was used as shared feature. The second neural network was trained for one specific category only (different neural networks were trained for each category). The hidden layer of this network was described as an aspect-specific feature. To form hybrid features, aspect-specific and shared features were concatenated. Finally, aspect-categories ware predicted using the hybrid features by a logistic regression classifier.

6.4 Aspect-Based Sentiment Detection

Wang and Liu [29] re-scaled word-embeddings in proportion to the aspect distribution in the review sentence. The idea behind this is to encode the relation between word and aspect associated with it. For this process, top n-aspects were filtered out based on the aspect distribution. Then filter the probabilistic mass with respect to each aspect. Aspect-specific probabilistic mass was propagated based on the parse tree. Finally, each word vector was re-scaled (weight distribution) and new word vectors were used in the CNN-based sentiment model.

Tang et al. [26, 27] claimed that standard LSTM is incapable to produce good results by incorporating only sentence representation. Along with sentence representation, if target information is used then the performance of sentiment classification will boost significantly. So they developed two models, target-dependent LSTM (TD-LSTM) and target-connection LSTM (TC-LSTM). In TD-LSTM, the selection of the relevant part of the context is based on the relatedness of it with the target word. They used two LSTMs, one for preceding context plus target word and another for target word plus following context. Last hidden vectors of both LSTMs after concatenation were passed to a softmax layer. Finally, target-specific sentiment polarity was inferred by the softmax layer. In this way, they include the context in both directions. An extension of the TD-LSTM was also introduced, named TC-LSTM. To expressly use the relationship between target word and context word, a target-connection component was used. Target vector was acquired by averaging the vectors of words it contains.

For first LSTM, they used word-embeddings of preceding context words concatenated with target vector, and for second LSTM, they used word-embeddings of following context words concatenated with target vector. This architecture works best for positive and negative examples but misclassifies examples belonging to the neutral category.

Neural network methods presented by [27] cannot efficiently identify which word in the sentence is more important. Existing work ignores or does not explicitly model the position information of the aspect-term in a sentence, which has been studied for improving performance in information retrieval (IR). Fortunately, attention mechanisms are an effective way to solve this problem. When an aspect-term occurs in a sentence, its neighboring words in the sentence should be given more attention than other words with long distance. Gu et al. [9] proposed a position-aware bidirectional attention network (PBAN) based on bidirectional Gated Recurrent Units (Bi-GRU). This model consists of three components: (1) Obtain position information of each word based on current aspect-term, then convert position information into position embedding. (2) The PBAN consists of two Bi-GRU for extracting aspect-level and sentence-level features respectively. (3) Use the bi-directional attention mechanism to model the mutual relation between aspect-terms and its corresponding sentence.

Wang et al. [30] explored the interrelation between an aspect and a sentence. To emphasize the crucial part of a sentence provided the aspect, aspect-to-sentence attention mechanism was used. There were two approaches by which this can be achieved.

One was the concatenation of aspect vector with sentence hidden representation. Another was to appending the aspect vector with the input vectors. So they proposed two models, attention-based LSTM (AT-LSTM) and attention-based LSTM with aspect-embedding (ATAE-LSTM). In AT-LSTM, aspect-embeddings were concatenated with the hidden representation of sentence to calculate the attention of different context words with respect to an aspect. On another hand, aspect-embedding was also appended into each input word vector in AT-LSTM to built ATAE-LSTM model. These models were designed to discover the aspect-based sentiment polarity. These models have the capabilities to deals with different parts of a sentence based on the current aspects.

Similar to aforementioned ATAE-LSTM model, Al-Smadi et al. [1] also developed an attention-based model named (AB-LSTM-PC) for aspect-based sentiment polarity classification. In this model, attention weights were calculated using word and aspect-embeddings. This model was trained on Arabic review dataset.

A sentence could contain multiple sentiment-target pairs. To isolate diverse opinion circumstances for different targets, He et al. [11] proposed an approach for improving the effectiveness of attention. To acquire the semantic significance of the opinion target, a better target representation method was proposed by authors. The computed attention weights rely entirely on the semantic associations between context words and the target representation. However, this may not be sufficient for differentiating opinions words for different targets. Apart from semantic information, syntactic information was also incorporated into the attention mechanism. Their syntax-based attention mechanism selectively focuses on a small subset of context words that are close to the target on the syntactic path which was obtained by applying a dependency parser on the review sentence.

Li et al. [15] model contains two key components, namely Truncated History-Attention (THA) and Selective Transformation Network (STN), for capturing aspect detection history and opinion summary, respectively. THA and STN were built on two LSTMs that generate the initial word representations for the primary ATE task and the auxiliary opinion detection task, respectively. THA was designed to integrate the information of aspect detection history into the current aspect feature to generate a new history-aware aspect representation. STN first calculates a new opinion representation conditioned on the current aspect candidate. Then, a bi-linear attention network was used to calculate the opinion summary as the weighted sum of the new opinion representations, according to their associations with the current aspect representation. Finally, the history-aware aspect representation and the opinion summary were concatenated as features for aspect prediction of the current time-step.

Inspired by the work of [24], Tang et al. [28] uses memory networks to capture the importance of context words for aspect-level sentiment classification. An external memory was created by stacking the context word vectors. A computational layer called hop took aspect vector as an input and focused on memory using attention mechanism. Each hop summed up the linearly transformed aspect vector and attention layer output, which was passed as an input to next hop. Last hop output was treated as sentence representation with respect to aspect and passed to softmax classifier to sentiment classification. Apart from performance enhancement, they observed

Table 2 Summary of state-of-the-art ABSA methods

Reference	Dataset	Domain	Task	Embeddings	Performance	Method
[1]	SemEval-16 Arabic	Hotel	ATE	Word2vec + FastText	F1-score: 69.98	Bi-LSTM-CRF
			AC-PD		Accuracy: 82.70	LSTM + Attention
[4]	SemEval-16 Task-5	Restaurant	ATE	Word2vec	F1-score: 72.44	Bi-LSTM-CRF
[7]	SemEval-14 Task-4	Restaurant	ATE	FastText	F1-score: 84.12	2 layer Bi-LSTM-CRF
		Laptop			F1-score: 77.96	2 layer Bi-LSTM-CRF
[34]	SemEval-14 Task-4	Restaurant	ACD	–	F1-score: 90.10	2-layer NN + Logistic R
[26]	Dong et al. 2014 [5]	Twitter	AT-PD	SSWE	F1-score: 69.00	TD-LSTM
					F1-score: 69.50	TC-LSTM
[30]	SemEval-14 Task-4	Restaurant	AT-PD	Glove	Accuracy: 76.60	AE-LSTM
					Accuracy: 77.20	ATAE-LSTM
		Laptop			Accuracy: 68.90	AE-LSTM
					Accuracy: 68.70	ATAE-LSTM
		Restaurant	AC-PD		Accuracy: 82.50	AE-LSTM
					Accuracy: 83.10	AT-LSTM
					Accuracy: 84.10	ATAE-LSTM
[9]	SemEval-14	Restaurant	AT-PD	Glove	Accuracy: 81.16	Bi-GRU + Attention
		Laptop			Accuracy: 74.12	
[11]	SemEval-14 Task-4	Restaurant	AC-PD	Glove	Accuracy: 80.36	LSTM + Attention
	SemEval-14 Task-4	Laptop			Accuracy: 71.94	
	SemEval-15 Task-12	Restaurant			Accuracy: 81.67	
	SemEval-16 Task-5	Restaurant			Accuracy: 84.64	
[29]	SemEval-15 Task-12	Laptop	ACD	Word2vec	F1-score: 51.30	2-layer NN
		Laptop	AC-PD		Accuracy: 78.30	Parse Tree + CNN

(continued)

Table 2 (continued)

Reference	Dataset	Domain	Task	Embeddings	Performance	Method
[20]	SemEval-14 Task-4 + Dataset by Qiu et al. [21]	Restaurant	ATE	Word2vec + Amazon Reviews Embeddings	F1-score: 87.17	7-layer CNN + Linguistic Pattern
		Laptop			F1-score: 82.32	
[28]	SemEval-14 Task-4	Restaurant	AT-PD	–	F1-score: 80.95	Memory Network + Attention
		Laptop			F1-score: 72.37	
[3]	SemEval-14 Task-4	Restaurant	AT-PD	GloVe	F1-score: 70.80	Memory Network + Attention + GRU
		Laptop			F1-score: 71.35	
	Dong et al. 2014	Twitter			F1-score: 67.30	
[13]	SemEval-15 Task-12	Restaurant	ATE	Amazon Reviews Embeddings	F1-score: 65.90	CNN + GRU
[31]	SemEval-14 Task-4	Restaurant	ATE	Word2vec	F1-score: 76.15	Bi-GRU + POS
		Laptop			F1-score: 60.75	

that deep memory network with 9 layers is 15 times faster than LSTM with CPU implementation.

Similarly, with described architecture Chen et al. [3] utilized multiple attention mechanism but differently, it uses Bi-LSTM to generate memory. The further relative position of words with respect to aspect target was used to generate location weighted memory. A recurrent network (GRU) was used to capture multiple attention on memory. Finally, softmax classifier was used to sentiment classification.

A summary of current state-of-the-art methods of ABSA is presented in Table 2.

7 Conclusion

In this chapter, we have presented some introduction about deep learning methodology used in Natural Language Processing especially for ABSA. This chapter discussed various tasks of ABSA and deals with the approaches used for these tasks. For sequence labeling task like aspect-term extraction, deep learning models consisting the Bi-LSTM-CRF are generally used. For the sentiment classification task, vanilla neural networks and CNN have shown state-of-the-art performance. For aspect sentiment detection task, exploiting the relationship between aspect and opinion, aspect and context words is beneficial. Researchers tried different approaches to extract this relationship and used in the deep neural networks. There is also a need to combine approaches to jointly perform two tasks, i.e., aspect detection and sentiment analysis. Overall, this chapter provides a starting point to dive into ABSA using deep learning approaches.

References

1. Al-Smadi, Mohammad, Bashar Talafha, Mahmoud Al-Ayyoub, and Yaser Jararweh. 2018. Using long short-term memory deep neural networks for aspect-based sentiment analysis of Arabic reviews. *International Journal of Machine Learning and Cybernetics*, pp. 1–13
2. Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473
3. Chen, Peng, Zhongqian Sun, Lidong Bing, and Wei Yang. 2017. Recurrent attention network on memory for aspect sentiment analysis. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 452–461
4. Chen, Tao, Ruifeng Xu, Yulan He, Xuan Wang. 2017. Improving sentiment analysis via sentence type classification using BiLSTM-CRF and CNN. *Expert Systems with Applications* **72**, 221–230
5. Dong, Li, Furu Wei, Chuanqi Tan, Duyu Tang, Ming Zhou, and Ke Xu. 2014. Adaptive recursive neural network for target-dependent twitter sentiment classification. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (volume 2: Short papers)*, vol. 2, 49–54
6. Feng, Jinzhan, Shuqin Cai, and Xiaomeng Ma. 2018. Enhanced sentiment labeling and implicit aspect identification by integration of deep convolution neural network and sequential algorithm. *Cluster Computing*, 1–19

7. Giannakopoulos, Athanasios, Claudiu Musat, Andreea Hossmann, and Michael Baeriswyl. 2017. Unsupervised aspect term extraction with b-lstm & crf using automatically labelled datasets. arXiv preprint arXiv:1709.05094
8. Graves, Alex, Schmidhuber, Jürgen: Framewise phoneme classification with bidirectional LSTM and other neural network architectures. Neural Networks **18**(5–6), 602–610 (2005)
9. Gu, Shuqin, Lipeng Zhang, Yuexian Hou, and Yin Song. 2018. A position-aware bidirectional attention network for aspect-level sentiment analysis. In *Proceedings of the 27th International Conference on Computational Linguistics*, 774–784
10. Güngör, Onur, Güngör, Tunga, Üsküdarli, Suzan: The effect of morphology in named entity recognition with sequence tagging. Natural Language Engineering **25**(1), 147–169 (2019)
11. He, Ruidan, Wee Sun Lee, Hwee Tou Ng, and Daniel Dahlmeier. 2018. Effective attention modeling for aspect-level sentiment classification. In *Proceedings of the 27th International Conference on Computational Linguistics*, 1121–1131
12. Hochreiter, Sepp, Schmidhuber, Jürgen: Long short-term memory. Neural Computation **9**(8), 1735–1780 (1997)
13. Jebbara, Soufian, and Philipp Cimiano. 2016. Aspect-based relational sentiment analysis using a stacked neural network architecture. In *Proceedings of the Twenty-second European Conference on Artificial Intelligence*, 1123–1131. IOS Press
14. Kim, Yoon. 2014. Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882
15. Li, Xin, Lidong Bing, Piji Li, Wai Lam, and Zhimou Yang. 2018. Aspect term extraction with history attention and selective transformation. arXiv preprint arXiv:1805.00760
16. Liu, Bing: Sentiment analysis and opinion mining. Synthesis Lectures on Human Language Technologies **5**(1), 1–167 (2012)
17. Mikolov, Tomas, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781
18. Mikolov, Tomas, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. *Advances in Neural Information Processing Systems*, 3111–3119
19. Pennington, Jeffrey, Richard Socher, and Christopher D. Manning. 2014. Glove: global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 1532–1543. ACL
20. Poria, Soujanya, Cambria, Erik, Gelbukh, Alexander: Aspect extraction for opinion mining with a deep convolutional neural network. Knowledge-Based Systems **108**, 42–49 (2016)
21. Qiu, Guang, Liu, Bing, Jiajun, Bu, Chen, Chun: Opinion word expansion and target extraction through double propagation. Computational linguistics **37**(1), 9–27 (2011)
22. Ramshaw, Lance A., and Mitchell P. Marcus. 1999. Text chunking using transformation-based learning. In *Natural Language Processing Using Very Large Corpora*, 157–176. Berlin: Springer
23. Schuster, Mike, and Kuldip K. Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11), 2673–2681
24. Sukhbaatar, Sainbayar, Jason Weston, Rob Fergus, et al. 2015. End-to-end memory networks. In *Advances in Neural Information Processing Systems*, 2440–2448
25. Sutskever, Ilya, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems*, 3104–3112
26. Tang, Duyu, Bing Qin, Xiaocheng Feng, and Ting Liu. 2015. Target-dependent sentiment classification with long short term memory. *CoRR*, abs/1512.01100
27. Tang, Duyu, Bing Qin, Xiaocheng Feng, and Ting Liu. Effective LSTMs for target-dependent sentiment classification. [9], 3298–3307
28. Tang, Duyu, Bing Qin, and Ting Liu. 2016. Aspect level sentiment classification with deep memory network. arXiv preprint arXiv:1605.08900
29. Wang, Bo, and Min Liu. 2015. Deep learning for aspect-based sentiment analysis. *Stanford University report*

30. Wang, Yequan, Minlie Huang, Li Zhao, et al. 2016. Attention-based LSTM for aspect-level sentiment classification. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 606–615
31. Wu, Chuhan, Fangzhao Wu, Sixing Wu, Zhigang Yuan, Yongfeng Huang. 2018. A hybrid unsupervised method for aspect term and opinion target extraction. *Knowledge-Based Systems* 148, 66–73
32. Xu, Hu, Bing Liu, Lei Shu, and Philip S. Yu. 2018. Double embeddings and cnn-based sequence labeling for aspect extraction. arXiv preprint arXiv:1805.04601
33. Zhang, Ye, and Byron Wallace. 2015. A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification. arXiv preprint arXiv:1510.03820
34. Zhou, Xinjie, Xiaojun Wan, and Jianguo Xiao. 2015. Representation learning for aspect category detection in online reviews. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 417–424. AAAI Press