Weakly-Supervised Left-Center-Right Context-Aware Aspect Category and Sentiment Classification

Gonem Lau, Flavius Frasincar[0000−0002−8031−758X](✉), and Finn van der Knaap

Erasmus University Rotterdam, Burgemeester Oudlaan 50, 3062 PA Rotterdam, the Netherlands
gonemlau@gmail.com, frasincar@ese.eur.nl, 573834fk@student.eur.nl

Abstract. Aspect-Based Sentiment Analysis (ABSA) aims to extract all aspects mentioned in a Web review and classify the aspect category and sentiment for each aspect. Most existing methods rely on single-task supervised approaches. However, ABSA tasks are not independent. Furthermore, obtaining labeled data might be difficult or expensive. The Context-aware Aspect category and Sentiment Classification (CASC) model addresses this issue by classifying categories and sentiments simultaneously using a weakly-supervised approach. However, CASC uses a simple neural network on the input text that does not exploit any other information. This paper proposes an extension named Left-Center-Right+CASC (LCR+CASC), where we implement a sophisticated neural model that exploits the location of explicit aspect expressions. Besides aspect categorization and sentiment classification, LCR+CASC also extracts target expressions from a sentence, which goes beyond CASC's abilities. This paper conducts two experiments on restaurant reviews: extracting target expressions and using annotated data that provide targets to evaluate the proposed model. Results show that LCR+CASC outperforms CASC when targets are given, and is able to extract target expressions to some extent.

Keywords: Aspect-based sentiment analysis · Weakly-supervised learning · Neural network

1 Introduction

With virtually everyone having access to the social Web, it is not unthinkable that vast amounts of text could be written every second of every day. It is not feasible to manually analyze everything. Therefore, many methods in the field of Natural Language Processing (NLP) have been proposed to extract information from textual data. One popular subfield, sentiment analysis, is about discovering and understanding opinions from user-generated data [11]. Companies can improve their products or services after analyzing customer sentiments from reviews. Furthermore, reviews are also valuable for consumers, as reviews can help customers with decision-making.

This paper focuses on ABSA, which aims to extract all aspects of a product or service mentioned in a review and classify the sentiment for each aspect [11]. However, in practice, the task is not strictly defined. Some datasets provide aspect categories, whereas others provide targets, in which targets are the explicit aspect expressions found in a sentence. Sentiment classification can also differ per dataset. For example, the SemEval 2015 [13] and SemEval 2016 [12] datasets define sentiment as positive, neutral, or negative, while the Yelp 2014 dataset [14] ranks sentiment from 1 to 5.

In more detail, targets could be described with one or multiple aspect categories, and optionally with the aspect term(s). Sentiments could be described with the sentiment polarity and with the opinion term(s). Therefore, ABSA could be divided into four subtasks: Aspect Term Extraction (ATE), Aspect Category Detection (ACD), Opinion Term Extraction (OTE), and Aspect Sentiment Classification (ASC) [19]. ATE extracts explicit aspect expressions in a text, whereas ACD categorizes the aspects. OTE extracts explicit opinion terms and ASC predicts the sentiment polarity. The main focus of this paper is ACD/ASC.

Even though ABSA is not confined to a single task, most methods address only one task. The LCR-Rot-hop++ model [15] exploits the position of a target for ASC but does not perform ATE or ACD. Some efforts have been made to create a multi-task model by considering a set of interrelated dependencies. An example is the Context-aware Aspect category and Sentiment Classification (CASC) model [9] which is a weakly-supervised approach to ACD and ASC. CASC follows a three-step process. First, class vocabularies for aspect and sentiment categories are constructed using only seed words. Second, unlabeled data is turned into labeled data using weak supervision. Third, a multi-task neural model performs ACD and ASC using a simple layer on top of BERT [2].

In this paper, inspired by the work in [9], we propose a weakly-supervised method for ABSA. The model, called Left-Center-Right+CASC (LCR+CASC), uses a more sophisticated neural model, LCR-Rot-hop++ [15], in the third step. Furthermore, LCR-Rot-hop++ was originally constructed to only perform ASC, whereas we also explore the performance of LCR-Rot-hop++ for ACD. Last, we modify the second step of CASC to also perform ATE. We evaluate the model using the SemEval datasets [12,13]. The implementation is based on code provided in [9] in Python and made freely available at https://github.com/Gogonemnem/LCR-PLUS-CASC (including aspect/sentiment seed words).

The contributions of this work can be summarized as follows. First, we adapt the classification layer of the CASC model. The linear layer is replaced by the sophisticated LCR-Rot-hop++ model. Therefore, positional information is exploited unlike in the simple linear layer. Second, we explore the performance of LCR-Rot-hop++ for the ACD task. The original model is only able to perform ASC. Therefore, inter-dependent information between ACD and ASC is exploited by employing multi-task learning. Third, we extend CASC by extracting aspect terms in the second step. Therefore, the model is able to perform ATE indirectly besides ACD and ASC. Fourth, we analyze the performance of our model on restaurant reviews. While the results are subpar with the state-of-the-art when using the targets detected by the ATE task, when using the gold targets the proposed method beats the state-of-the-art for both ACD and ASC (the main focus of this paper).

The rest of the paper is structured as follows. Section 2 provides an overview of related work. Next, Sect. 3 describes the datasets used for the analysis. Then, Sect. 4 explains the proposed model. In Sect. 5, the performance of the proposed model is compared with a state-of-the-art approach. Last, we present concluding remarks in Sect. 6 together with suggestions for future research.

2 Related Work

This section presents an overview of previous work. First, Subsect. 2.1 gives an overview of methods that solve single tasks in ABSA. The OTE task is not discussed as we do not extract opinion expressions in this paper. Then, Subsect. 2.2 shows recent progress in solving multiple subtasks using multi-task learning.

2.1 Single-task ABSA

Aspect Term Extraction. While supervised approaches yield impressive results, they often require large labeled datasets. The neural models are often not bottlenecked by their simplicity but rather by the available data [6], which motivates the latest trend of un- and semi-supervised models. [3] proposes the Attention-based Aspect Extraction (ABAE) model which de-emphasizes irrelevant words through an attention mechanism to improve the coherence of extracted aspects.

Aspect Category Detection. A big difference between ACD and ATE is that no explicit aspect expressions have to be in a sentence for ACD. For example, “It’s expensive and gross” could be categorized in the categories “price” and “food”, whereas ATE cannot identify what is being reviewed. Semi-supervised machine learning approaches often consist of first applying ATE and, subsequently, ACD on those aspect terms. In other words, sentences without targets cannot be categorized. First, candidate aspect terms are extracted. Then, those candidate terms are mapped or clustered to pre-defined aspect categories. An example of a semi-supervised ACD method is ABAE. One drawback of this model is that the learned aspects need to be mapped manually. Therefore, [7] proposes a teacher-student framework that extends the ABAE model by leveraging seed words, which eliminates the need to manually assign the learned aspects.

Aspect Sentiment Classification. Although many methods have been proposed for supervised ASC [19], unsupervised ASC has not seen much progress. Data can be provided in two different ways (aspect term data or category data). Differences are subtle, but one interesting difference is that positional information can be exploited with aspect term data. For example, LCR-Rot [21] exploits this information by splitting sentences into a left context, a target, and a right context. Many extensions of the LCR-Rot model have been proposed [15,16].

2.2 Multi-task ABSA

Early studies in unsupervised learning that jointly extract aspects and sentiments are mostly based on Latent Dirichlet Allocation (LDA) [1]. [18] proposes the Joint Aspect/Sentiment (JAS) model which adapts LDA by introducing sentiment-related variables and integrating sentiment prior knowledge. [20] further extracts aspect-specific opinions in a generative process.

Recent studies propose weakly-supervised methods for the compound task. [22] introduces the Joint Aspect-Sentiment Analysis (JASA) which extends the ABAE model to learn aspect/sentiment representations. Furthermore, the authors make use of multi-task learning by letting the aspect and sentiment representations interact. Therefore, aspect-specific opinions (such as delicious for the food category) are learned. [6] proposes the Joint Aspect Sentiment (JASen) model. First, the model learns joint topic embeddings. Then, neural layers pre- and self-train through embedding-based predictions, which generalize the word-level discriminative information on unlabeled data.

More recently, [9] proposes the CASC model. The CASC model turns unlabeled data into noisy labeled data. Then, neural models are trained on the noisy labeled data. The process of turning unlabeled data into labeled data is only weakly supervised and requires a small number of seed words per aspect/sentiment category, which turns into a full-fledged vocabulary list.

3 Data

To train the proposed model, we use the Yelp dataset [6], which consists of 17,027 unlabeled review sentences. This dataset is chosen as it resides in the restaurant domain just like our evaluation datasets. Furthermore, the Yelp dataset is much larger compared to the SemEval datasets [12,13].

The datasets used for evaluation are the SemEval 2015 [13] and SemEval 2016 [12] datasets. Specifically, this paper focuses on restaurant reviews. Each review consists of at least one sentence, and each sentence contains the sentiment on at least one aspect. The sentiment can either be positive, neutral, or negative.

First, we remove aspect targets that belong to multiple aspect categories due to limitations in our model. Similar to [6] and [9], neutral sentiment polarities are ignored. This is because it is inherently ambiguous to classify neutral sentiment, making it difficult to bootstrap our algorithm with a set of neutral seed words. Moreover, aspect categories are more general than the original SemEval datasets and only contain the categories: food, place, and service. [9] merged the food and drink SemEval categories into the food category, and the ambiance and location SemEval categories into the place category. Furthermore, the authors removed sentences containing multiple aspects. For our first experiment, where we extract aspect target expressions, we also remove sentences with multiple aspects as our model can only extract one aspect per sentence. However, we keep sentences with multiple aspects for the second experiment, where we do not perform ATE. Another difference compared to [9] is that sentences with implicit targets are not considered in this research due to limitations in our model.

Table 1 shows the full and processed SemEval datasets. After all pre-processing steps, datasets that consider multi-labeled data keep around 60% of the data, whereas singly-labeled data keep less than 30% of the original size.

Table 1. Descriptive statistics of the SemEval 2015 and SemEval 2016 datasets

Dataset	Positive		Negative		Food		Place		Service		Total
	Freq.	%	Freq.	%	Freq.	%	Freq.	%	Freq.	%	Freq.
SemEval 2015 - Full	454	54%	346	41%	365	43%	305	36%	175	21%	845
SemEval 2016 - Full	611	71%	204	24%	429	50%	275	32%	155	18%	859
SemEval 2015 Single label - Gold	127	59%	90	41%	106	49%	66	30%	45	21%	217
SemEval 2015 Multi label - Gold	298	64%	166	36%	250	54%	115	25%	99	21%	464
SemEval 2016 Single label - Gold	194	80%	48	20%	115	48%	82	34%	45	19%	242
SemEval 2016 Multi label - Gold	450	81%	104	19%	310	56%	139	25%	105	19%	554

4 Methodology

This section discusses the proposed model. First, Subsect. 4.1 formulates the problem and introduces general notation. Then, Subsect. 4.2 explains the modified CASC model in detail. Last, Subsect. 4.3 explains the training setup.

4.1 Task Formulation

Let the input be a corpus $\mathcal{D} = [X_1, X_2, \ldots, X_n]$ consisting of n unlabeled review sentences from a domain. Given the set of aspect categories A together with a small list of seed words L_a for each aspect category $a \in A$, and sentiment polarities S along with a small list of seed words L_s for each sentiment polarity $s \in S$, the objective is to predict a pair of (a, s) for an unseen review sentence.

Because words are difficult to work with, word embeddings are often used. In this study, Domain Knowledge BERT (DK-BERT) [17] is used. DK-BERT is a post-trained version of BERT [2] that has been trained on domain data, which in this case is the restaurant review data provided by Yelp [6].

4.2 Modified CASC

This section describes the CASC model with our modifications in detail. In the first step, class vocabularies for aspect and sentiment categories are constructed through contextual embeddings using only seed words. Second, unlabeled data is turned into labeled data in a weakly-supervised manner using overlap scores. Third, a simple neural model with a linear layer is replaced with a more sophisticated model to perform ACD and ASC.

Class Vocabulary Construction. Given a set of seed words L_a corresponding to aspect $a \in A$, we find the set of sentences $X_a \subset \mathcal{D}$ that contain any of the seed words. Then, sentence $X_i \in X_a$ is passed through the post-trained DK-BERT Masked Language Model (DK-BERT MLM). DK-BERT MLM outputs token replacement probability scores $P_i \in \mathbb{R}^{|X_i| \times |V|}$, with V being the vocabulary set used by DK-BERT. However, only replacement candidates of tokens that represent seed words are considered. Next, those replacement candidates are passed through a filter to remove stop words and punctuation. Then, the top K words are selected as replacement candidates R_i based on the probability scores computed by DK-BERT MLM. All replacement candidates R_i for all $X_i \in X_a$ are collected and added to a frequency table for all $a \in A$. Subsequently, the top M most frequent words for aspect category a are selected to construct the aspect vocabulary V_a. Last, words that appear in multiple vocabularies are removed in all sets. A similar procedure is used to construct the sentiment vocabularies V_s for all $s \in S$.

The intuition behind these steps is that words within a sentence can be replaced by words that carry a similar contextual meaning. DK-BERT MLM is trained to provide such replacement candidates, such that the words outputted will have a similar meaning as the seed words. Furthermore, the vocabulary sets are disjoint sets to remove ambiguities across aspect and sentiment classes.

Labeled Data Preparation. Following the work from [5], the CASC model exploits the notion of aspect terms to be nouns and opinion terms to be adjectives or adverbs. Therefore, nouns, adjectives, and adverbs are extracted using a Part-of-Speech tagger as *potential-aspects* and *potential-opinions*.

First, we find the set of all sentences $X_q \subset \mathcal{D}$ which contain at least one *potential-aspect* and one *potential-opinion*. Then, sentence $X_j \in X_q$ is passed through the post-trained DK-BERT MLM to find replacement candidates with probability scores for each *potential-aspect*. Then, the list G_{aspect} is created by taking the top K replacement words based on the probability scores. For sentence X_j, we find the overlap score S_a of each aspect $a \in A$ by counting the common words between its corresponding aspect vocabulary V_a and the list G_{aspect}. Overlap scores for all sentences X_q with all aspect categories A are stored in the score matrix $\mathcal{M} \in \mathbb{R}^{|X_q| \times |A|}$.

The vocabularies in the previous step are created with DK-BERT, which is post-trained using a real-world domain-specific dataset. [9] argues that these datasets are often imbalanced, which could imply that certain vocabularies contain a relatively large number of semantically coherent words compared to other vocabularies. This difference could affect the variance of the overlap scores per category. Therefore, the overlap scores per category are standardized.

For sentence X_j, the aspect category is assigned if the standardized score is the largest and above a pre-defined threshold λ. A similar procedure is applied for sentiment categories. Next, a labeled dataset $\mathcal{D}_{\mathcal{L}} \subset \mathcal{D}$ is constructed.

In contrast to the original CASC model, we also extract aspect terms in this step. During the aspect labeling process, each aspect label a is assigned an aspect term for each sentence X_j. When the aspect label is decided, the corresponding aspect term is assigned to the sentence. There are many methods for assigning aspect target expressions. An approach could be to identify noun chunks [4] and select chunks with the highest scores. Noun chunk scores can be computed in various ways. One example is to take the average of the overlap score among all nouns in the noun chunk. CASC uses a so-called average score labeler, whereas our proposed model uses a maximum score labeler. An average score labeler computes the average overlap score for all *potential-aspects* in a sentence, per aspect category. A maximum score labeler labels sentences based on the aspect target expression. Therefore, the labeling is done by using the *potential-aspect* (which becomes the aspect target expression) that has the highest overlap score in a sentence.

Joint Neural Network for ACD and ASC. In the original CASC model, the authors used a simple yet effective neural model to classify aspects and sentiments. The authors make use of DK-BERT embeddings. We refer to [9] for additional information about the original joint neural network.

However, aspect term positions are not exploited in this neural model. Therefore, we suggest using the LCR-Rot-hop++ model [15]. We propose the double task variant of the model to exploit inter-dependent information between ACD and ASC. This sophisticated model exploits positional information by using three Bidirectional Long Short Term Memory (BiLSTM) layers. Furthermore, two types of attention mechanisms are implemented in an iterative manner to exploit local and global contexts. The LCR-Rot model [21] was originally built for ASC. Therefore, a slight modification is made to also perform ACD.

We apply DK-BERT to generate embeddings. Then, sentence X_l is split into a left context, a target, and a right context of lengths L, T, and R, respectively. Moreover, the contexts are represented as $X_l^l = \left(w_1^l, \ldots, w_L^l\right)$, $X_l^t = (w_1^t, \ldots, w_T^t)$, and $X_l^r = (w_1^r, \ldots, w_R^r)$, where w_i^j represents the ith word in context j. To ensure that each context will have the same number of tokens, padding tokens $[PAD]$ are added if the context lacks tokens. Furthermore, contexts are truncated if there are too many tokens. Then, $[CLS]$ token is added at the beginning and separation tokens $[SEP]$ are added between the contexts and target to easily find the token length of each context. Note that the special tokens $[CLS]$ and $[SEP]$ are ignored after embedding. Therefore, the embeddings of a sentence are expressed as $H^l \in \mathbb{R}^{L\times d}$, $H^t \in \mathbb{R}^{T\times d}$, and $H^r \in \mathbb{R}^{R\times d}$. Next, each embedding part feeds separate BiLSTM layers which produce hidden states $B^l = \left(b_1^l, \ldots, b_L^l\right)$, $B^t = (b_1^t, \ldots, b_T^t)$, and $B^r = (b_1^r, \ldots, b_R^r)$.

Then, a three-step attention mechanism is iteratively applied over the three hidden states. The first and second attention mechanisms are rotary attention mechanisms that exploit local information, whereas the third is a hierarchical attention mechanism that exploits global information. First, new left and right context representations are generated by using old target information. Second, two new target representations are generated by using the context representations produced in the first step. Third, the representations are updated based on relative importance separately for context and target. The mathematical details are presented below.

First, an attention mechanism is applied. The new context representations are weighted sums of the hidden states of each context:

$$
\begin{aligned}
r^c &= B^{c\top} \times \alpha^c, && (1)\\
\alpha^c &= \operatorname{softmax}\left(f\left(B^c, r^{t_c}\right)\right), && (2)\\
f\left(B^c, r^{t_c}\right) &= \tanh\left(B^{c\top} \times W^c \times r^{t_c} + b^c \times \mathbf{1}\right), && (3)
\end{aligned}
$$

where c denotes the left or right context $\{l, r\}$ consisting of C words. Note that C equals either L or R. Furthermore, $B^c \in \mathbb{R}^{C \times 2d}$ corresponds to the hidden states for context c, and $\alpha^c \in \mathbb{R}^C$ denotes the attention weights assigned to each hidden state. Then, t_c denotes the left or right target $\{t_l, t_r\}$ consisting of T words. Furthermore, $r^{t_c} \in \mathbb{R}^{2d}$ corresponds to the target representation of target side c, and $\mathbf{1} \in \mathbb{R}^C$ denotes a vector of ones. Then, the trainable parameters are the weight matrix $W^c \in \mathbb{R}^{2d \times 2d}$ and the bias scalar $b^c \in \mathbb{R}$. Unfortunately, the first iteration has no old target information. Therefore, $r^{t_c} \in \mathbb{R}^{2d}$ is extracted using the average pooling operator. In other iterations, target representations r^{t_c}, which are computed in step two, are used.

Second, another attention mechanism is applied to generate the left and right target representations. Two target representations are generated to exploit context representations separately. Thus, the left target representation uses the left context, and the right target representation the right context. The computations are similar to Equation 1, 2, and 3. However, the c's and t_c's are swapped in this step, thus changing dimensions containing size C to T, and vice versa.

Third, a hierarchical attention mechanism is applied to exploit global information. Intuitively, the mechanism decides whether the left or right context provides more relevant information about the target. The left and right contexts are scaled with respect to each other. Furthermore, the left and right targets are scaled as well, but separately from the contexts. The computations are as follows:

$$
\begin{aligned}
\hat{r}^p &= \alpha^{p\top} \times I_2 \times r^p, && (4)\\
\alpha^p &= \operatorname{softmax}\left(f\left(r^p\right)\right), && (5)\\
f\left(r^p\right) &= \tanh\left(r^p \times W^p + b^p \times \mathbf{1}\right), && (6)
\end{aligned}
$$

where $r^p \in \mathbb{R}^{2 \times 2d}$ denotes the vertically concatenated contexts $[r^l; r^r]$ or targets $[r^{t_l}; r^{t_r}]$ decided by p. $I_2 \in \mathbb{R}^{2 \times 2}$ denotes the identity matrix. Then, $\alpha^p \in \mathbb{R}^2$ denotes the attention weight vector assigned to each representation. The trainable parameters are the weight vector $W^p \in \mathbb{R}^{2d}$ and the bias scalar $b^p \in \mathbb{R}$.

Last, all four representation vectors are horizontally concatenated and feed a dense layer with a softmax function and bias vector for sentiment prediction. However, the original model does not classify aspect categories. Therefore, we extend the LCR-Rot-hop++ model. Modifying this neural model is inspired by the CASC model. Instead of passing the sequence representation through a single layer, two dense layers are instantiated for the ACD and ASC tasks, respectively. The ASC branch is the same as before. The difference for the ACD task is that its layer has its own weight matrix and bias vector.

4.3 Training Setup

This section discusses the training setup of the neural model. First, we present the loss function. Then, we discuss the hyperparameters and how we optimize them.

Training Procedure. The model minimizes a General Cross Entropy (GCE) function, which exploits both the Categorical Cross Entropy (CCE) and Mean Absolute Error (MAE). CCE converges quickly but overfits to noise, while MAE is robust to noise but converges slowly. It has been shown that GCE performs better than CCE [9], as the dataset is semi-automatically labeled which introduces noise while annotating. In short, GCE de-emphasizes difficult samples compared to CCE but accentuates them more than MAE during training. The unregularized overall loss $\mathcal{L}$ is the sum of the aspect category classification loss $\mathcal{L}_a$ and sentiment polarity classification loss $\mathcal{L}_s$, which are defined as follows:

$$\mathcal{L}_a = (1 - \hat{a}^q_{y_a})/q, \tag{7}$$

$$\mathcal{L}_s = (1 - \hat{s}^q_{y_s})/q, \tag{8}$$

where $\hat{a}_{y_a}$ and $\hat{s}_{y_s}$ denote the predicted probabilities against the true aspect and sentiment labels, respectively, and $q \in (0, 1)$ is a hyperparameter. The loss function becomes the MAE function when $q = 1$, and the CCE function when $q = 0$. Differently than CASC, L_1 and L_2 regularizations are applied to avoid overfitting. The regularization is deemed necessary, as the neural model is more sophisticated.

For loss minimization, weight matrices described in Subsect. 4.2 are randomly initialized using a uniform distribution $U(-0.1, 0.1)$. Furthermore, biases are initialized to zero. The rest of the parameters are set using the default settings provided by TensorFlow. The algorithm used to minimize the loss is Adam [8].

Hyperparameter Optimization. The hyperparameters are optimized by the Hyperband algorithm [10]. Furthermore, hyperparameters that are not involved in the neural model and not discussed in this section are taken from CASC [9].

We optimize eight hyperparameters. Table 2 shows the hyperparameters for LCR+CASC, LCR+CASC-CON, and LCR+CASC+ASYNC (the last two models are discussed in the next section). Most hyperparameters vary across different models and do not display interesting results. However, we notice that the number of hops is always higher than the three hops given in [15]. Furthermore, the values for q are relatively low, meaning that the loss function is closer to CCE than MAE.

Table 2. Optimized hyperparameters of various models

Hyperparameter	LCR+CASC	LCR+CASC-CON	LCR+CASC+ASYNC
L_1	10^-7	0.0001	10^-5
L_2	10^-7	0.001	10^-7
Learning Rate	0.001	0.001	0.01
Hops	6	8	8
q	0.1	0.3	0.3
BiLSTM units	550	650	700
Dropout rate 1	0.6	0.3	0.5
Dropout rate 2	0.3	0.2	0.3

5 Results

We discuss the results of the proposed model in this section. First, Subsect. 5.1 discusses the evaluation measures and gives the baseline models that are used for comparison. Then, we present the datasets in Subsect. 5.2. Last, Subsect. 5.3 compares the results of the proposed model and baseline models.

5.1 Performance Measures & Baseline Models

We evaluate the predictive performance of the models using the out-of-sample accuracy and macro-F1 scores. This paper compares the novel model against versions of the CASC model. The considered baseline models are:
CASC [9]: The model framework presented in [9] using the post-trained DK-BERT MLM and a small set of seed words to prepare labeled data.
CASC+MAX: We use the score that corresponds to the word that has the maximum score amongst all *potential-aspects* and *potential-opinions* in a sentence. Sentences are labeled according to the maximum score, whereas the CASC model uses the average.
CASC+MAX+ATE: An extension of CASC+MAX, where we also extract the aspect target expressions. Similar to the LCR+CASC model, this model introduces [SEP] tokens before and after target expressions.
LCR+CASC: Our model framework builds on the CASC+MAX+ATE model. We replace the simple linear neural model with the sophisticated LCR-Rot-hop++ [15] neural model.
LCR+CASC-DL: We omit the neural model altogether and use the labeler described in labeled data preparation. We name this method LCR+CASC-DL but it is equivalent to CASC+MAX+ATE-DL.

LCR+CASC-CON: We remove the left- and right-context-aware BiLSTM output in the last prediction layer. We only use the concatenated target representations $\{r_l^t, r_r^t\}$.
LCR+CASC+ASYNC: We create separate neural layers for each subtask. Therefore, each task is asynchronously solved, whereas the LCR+CASC model shares all neural layers (besides the classification head) for the aspect and sentiment classification tasks.
[MODEL]+ATE (GOLD): These types of models use the gold aspect target annotations provided by the SemEval datasets, instead of extracting the aspect target expressions. Note that the noisy training data is still generated using a labeler.

5.2 Processed Data

The Yelp dataset is not labeled and the resulting datasets may differ per score calculator. Furthermore, the SemEval datasets are also smaller as the aspect target cannot always be found by the labeler. Table 3 shows the datasets that are used for training (Yelp) and evaluation (SemEval). Note that the Yelp dataset differs between the average and maximum score labeler.

Table 3. Descriptive statistics of the aspect classes and the polarities of the Yelp and SemEval datasets

Dataset	Postive		Negative		Food		Place		Service		Total
	Freq.	%	Freq.	%	Freq.	%	Freq.	%	Freq.	%	Freq.
Yelp CASC	1543	69%	706	31%	881	39%	781	35%	587	26%	2249
Yelp Max	1744	74%	601	26%	689	29%	920	39%	736	31%	2345
SemEval 2015	82	46%	96	54%	83	47%	55	31%	40	22%	178
SemEval 2016	159	78%	46	22%	88	43%	73	36%	44	21%	205

5.3 Performance Results

In this section, we compare the different models described in Subsect. 5.1. First, we use the data with no gold target annotations. Then, we use test data where sentences have been split into the correct contexts using gold target annotations.

From Table 4 we notice that CASC outperforms all methods when it comes to ACD. Furthermore, from Table 5 we observe that it performs relatively well for ASC. CASC beats LCR+CASC by a significant amount at ACD, whereas the performance comparison for ASC is more ambiguous. CASC performs worse at ASC for the 2016 dataset, while LCR+CASC performs worse than all neural models for the 2015 dataset for ASC. To understand the effectiveness of different novel components, we perform an ablation study. We remove or add components as is explained in Subsect. 5.1. The results are shown in Table 4 for aspect classification and in Table 5 for sentiment detection.

Table 4. Performance of various methods on aspect classification

Aspect	2015		2016	
Model	Acc.	Macro-F1	Acc.	Macro-F1
CASC	**85.71**	**85.43**	**86.78**	**86.93**
CASC+MAX	64.52	63.51	65.70	64.46
CASC+MAX+ATE	83.71	83.14	83.90	83.87
LCR+CASC-DL	65.90	65.41	69.42	68.64
LCR+CASC-CON	79.78	79.58	83.41	83.43
LCR+CASC+ASYNC	80.90	80.43	82.93	83.12
LCR+CASC	80.90	80.44	82.44	82.52

Table 5. Performance of various methods on sentiment classification

Sentiment	2015		2016	
Model	Acc.	Macro-F1	Acc.	Macro-F1
CASC	90.32	90.20	87.19	82.92
CASC+MAX	**92.17**	**92.01**	88.84	84.41
CASC+MAX+ATE	89.89	89.84	89.76	86.56
LCR+CASC-DL	68.66	68.37	57.44	55.29
LCR+CASC-CON	88.76	88.65	93.17	90.61
LCR+CASC+ASYNC	89.89	89.78	91.22	87.93
LCR+CASC	88.20	88.07	**94.15**	**91.84**

First, the maximum score labeler drops CASC's performance for ACD. Furthermore, CASC+MAX loses over 20 percentage points for ACD. Interestingly, ASC is more robust than ACD as the neural network is able to find sentiments well. Second, CASC+MAX+ATE produces good results, even though the maximum score labeler produces poor results. However, it performs worse than CASC in most situations since it only beats CASC in the 2016 dataset for ASC. Combined with the previous results, it indicates that the neural model is able to pick up some patterns if the location of the aspect is provided. Even with imperfect target location information, the performance is close to the performance of CASC. Third, LCR+CASC-CON beats LCR+CASC in some cases. One possible reason could be that the two target representations capture the most relevant information of the contexts. Fourth, LCR+CASC+ASYNC produces similar results compared to LCR+CASC for ACD, whereas this is not the case for ASC for the 2016 dataset.

We now consider data with gold annotations. Table 6 and Table 7 show the results for ACD and ASC, respectively. We investigate this to understand if studying more complex models is worth it. The maximum score labeler might cause poor results for LCR+CASC, indicating that ATE in this model must improve. We investigate the potential of complex models by not performing ATE for the evaluation data. Note that sentences in the training data are still annotated using the maximum score labeler and remain unchanged. Because we use gold annotations, sentences in the test data can contain multiple aspects. Therefore, we also investigate the performance of sentences with multiple aspects.

Table 6. Performance of various methods on aspect classification with gold data

Aspect	Single Aspect				Multiple Aspects			
	2015		2016		2015		2016	
Model	Acc.	Macro-F1	Acc.	Macro-F1	Acc.	Macro-F1	Acc.	Macro-F1
CASC+ATE (GOLD)	86.18	86.18	86.36	85.96	75.00	73.93	79.06	77.00
CASC+MAX+ATE (GOLD)	84.33	83.73	89.67	88.90	85.56	85.03	85.56	84.86
LCR+CASC-CON+ATE (GOLD)	90.78	91.27	94.21	94.80	92.03	91.80	93.14	92.97
LCR+CASC+ASYNC+ATE (GOLD)	**94.47**	**94.71**	**95.87**	**95.17**	**94.83**	**94.52**	**94.77**	94.05
LCR+CASC+ATE (GOLD)	92.63	93.28	94.63	94.96	93.53	93.36	94.22	**94.10**

Table 7. Performance of various methods on sentiment classification with gold data

Sentiment	Single Aspect				Multiple Aspects			
	2015		2016		2015		2016	
Model	Acc.	Macro-F1	Acc.	Macro-F1	Acc.	Macro-F1	Acc.	Macro-F1
CASC+ATE (GOLD)	**92.63**	**92.53**	85.95	81.37	87.28	**86.36**	88.81	83.61
CASC+MAX+ATE (GOLD)	89.40	88.99	88.43	83.31	87.50	86.13	90.07	84.24
LCR+CASC-CON+ATE (GOLD)	86.64	86.17	91.74	87.58	84.70	83.14	92.06	87.44
LCR+CASC+ASYNC+ATE (GOLD)	89.40	89.07	91.32	87.40	85.99	84.61	91.88	87.20
LCR+CASC+ATE (GOLD)	89.86	89.36	**93.80**	**90.33**	**87.93**	86.35	**92.96**	**88.33**

First, CASC drops in performance when including multiple aspects while LCR+CASC does not. It seems that more sophisticated models perform better at ACD. Second, by using the maximum score labeler, performance is on par with singly-labeled sentences. Furthermore, the performance decrease from single- to multi-labeled data for ACD is larger compared to ASC. Third, unlike before, LCR+CASC-CON performs worse than LCR+CASC, indicating that the left and right contexts are useful only when the target expression is correct. The performance loss is less for ACD than for ASC as ACD likely depends less on the context. Sentiments, however, depend more on context. Fourth, LCR+CASC+ASYNC performs the best in all but one situation for ACD, indicating that separate neural layers benefit the model. However, this model performs worse compared to LCR+CASC for ASC.

In short, LCR+CASC outperforms CASC in most situations when the aspect target expression is given for test data, even with the noisy training labels generated by the maximum score labeler. Compared to CASC, multi-labeled data see much improvement as the sophisticated neural model performs better with multi-labeled data.

6 Conclusion

In this paper, we introduced a novel model by combining the state-of-the-art weakly-supervised CASC [9] and supervised LCR-Rot-hop++ [15] models to classify aspect categories and sentiment polarities. The models were modified to work with each other. First, a new scoring method had to be constructed to extract the aspect target expression. Second, the sophisticated LCR-Rot-hop++ had to be altered to also solve the ACD task.

We found that the LCR+CASC model performed subpar compared to the CASC model when ATE was also performed. Results showed that the LCR-Rot-hop++ model can be extended to a joint neural model to solve the ACD and ASC tasks as the LCR+CASC model is able to detect which aspect category belongs to a target. The weakness of the model is ATE. Labeling and extracting a word with the highest aspect score, rather than labeling the sentence based on the average score, produced modest results. However, the LCR+CASC outperformed CASC in most situations when target locations were correctly provided in the test data. Specifically, LCR+CASC outperformed CASC significantly when it comes to multi-labeled data. Moreover, the components of the neural model seemed to have a positive effect on the performance. Thus, our proposed model is able to exploit target location information as it outperformed CASC when using high-quality target data.

This research focused on the restaurant domain. Although CASC has seen great results in different domains, LCR+CASC's performance in different domains is not yet known. This is left as a suggestion for future research. Furthermore, results showed that the ATE part of the labeled data preparation step produced weak performance. Rather than extracting words that have the highest score in a sentence, future research could examine the extraction of noun phrases or more advanced scoring methods. Noun phrases seem promising as it is not as broad as taking the average over the whole sentence, but not too fine-grained as extracting single words.

References

1. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. The Journal of Machine Learning Research **3**, 993–1022 (2003)
2. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: Pre-training of deep bidirectional transformers for language understanding. In: 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2019). pp. 4171–4186. ACL (2019)
3. He, R., Lee, W.S., Ng, H.T., Dahlmeier, D.: An unsupervised neural attention model for aspect extraction. In: 55th Annual Meeting of the Association for Computational Linguistics (ACL 2017). pp. 388–397. ACL (2017)
4. Honnibal, M., Montani, I., Van Landeghem, S., Boyd, A.: spaCy: Industrial-strength natural language processing in Python (2020)
5. Hu, M., Liu, B.: Mining and summarizing customer reviews. In: 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2004). pp. 168–177. ACM (2004)
6. Huang, J., Meng, Y., Guo, F., Ji, H., Han, J.: Weakly-supervised aspect-based sentiment analysis via joint aspect-sentiment topic embedding. In: 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP). pp. 6989–6999. ACL (2020)
7. Karamanolakis, G., Hsu, D., Gravano, L.: Leveraging just a few keywords for fine-grained aspect detection through weakly supervised co-training. In: 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP 2019). pp. 4611–4621. ACL (2019)
8. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: 3rd International Conference on Learning Representations (ICLR 2015) (2015)

9. Kumar, A., Gupta, P., Balan, R., Neti, L.B.M., Malapati, A.: BERT based semi-supervised hybrid approach for aspect and sentiment classification. Neural Processing Letters **53**(6), 4207–4224 (2021)
10. Li, L., Jamieson, K., DeSalvo, G., Rostamizadeh, A., Talwalkar, A.: Hyperband: A novel bandit-based approach to hyperparameter optimization. Journal of Machine Learning Research **18**(185), 1–52 (2018)
11. Liu, B.: Sentiment analysis and opinion mining. Synthesis Lectures on Human Language Technologies **5**(1), 1–167 (2012)
12. Pontiki, M., Galanis, D., Papageorgiou, H., Androutsopoulos, I., Manandhar, S., AL-Smadi, M., Al-Ayyoub, M., Zhao, Y., Qin, B., De Clercq, O., Hoste, V., Apidianaki, M., Tannier, X., Loukachevitch, N., Kotelnikov, E., Bel, N., Jiménez-Zafra, S.M., Eryiğit, G.: SemEval-2016 task 5: Aspect based sentiment analysis. In: 10th International Workshop on Semantic Evaluation (SemEval-2016). pp. 19–30. ACL (2016)
13. Pontiki, M., Galanis, D., Papageorgiou, H., Manandhar, S., Androutsopoulos, I.: SemEval-2015 task 12: Aspect based sentiment analysis. In: 9th International Workshop on Semantic Evaluation (SemEval 2015). pp. 486–495. ACL (2015)
14. Tang, D., Qin, B., Liu, T.: Aspect level sentiment classification with deep memory network. In: 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP 2016). pp. 214–224. ACL (2016)
15. Truşcǎ, M.M., Wassenberg, D., Frasincar, F., Dekker, R.: A hybrid approach for aspect-based sentiment analysis using deep contextual word embeddings and hierarchical attention. In: 20th International Conference on Web Engineering (ICWE 2020). LNCS, vol. 12128, pp. 365–380. Springer (2020)
16. Wallaart, O., Frasincar, F.: A hybrid approach for aspect-based sentiment analysis using a lexicalized domain ontology and attentional neural models. In: 16th Extended Semantic Web Conference (ESWC 2019). LNCS, vol. 11503, pp. 363–378. Springer (2019)
17. Xu, H., Liu, B., Shu, L., Yu, P.: BERT post-training for review reading comprehension and aspect-based sentiment analysis. In: 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2019). pp. 2324–2335. ACL (2019)
18. Xu, X., Tan, S., Liu, Y., Cheng, X., Lin, Z.: Towards jointly extracting aspects and aspect-specific sentiment knowledge. In: 21st ACM International Conference on Information and Knowledge Management (CIKM 2012). pp. 1895–1899. ACM (2012)
19. Zhang, W., Li, X., Deng, Y., Bing, L., Lam, W.: A survey on aspect-based sentiment analysis: Tasks, methods, and challenges. IEEE Transactions on Knowledge and Data Engineering **35**(11), 11019–11038 (2023)
20. Zhao, X., Jiang, J., Yan, H., Li, X.: Jointly modeling aspects and opinions with a MaxEnt-LDA hybrid. In: 2010 Conference on Empirical Methods in Natural Language Processing (EMNLP 2010). pp. 56–65. ACL (2010)
21. Zheng, S., Xia, R.: Left-center-right separated neural network for aspect-based sentiment analysis with rotatory attention. arXiv preprint arXiv:1802.00892 (2018)
22. Zhuang, H., Guo, F., Zhang, C., Liu, L., Han, J.: Joint aspect-sentiment analysis with minimal user guidance. In: 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (ACM 2020), pp. 1241–1250. ACM (2020)