A Cross-Domain Aspect-Based Sentiment Classification by Masking the Domain-Specific Words

Junhee Lee
Erasmus University Rotterdam
Rotterdam, the Netherlands
ejoone611@gmail.com

Flavius Frasincar
Erasmus University Rotterdam
Rotterdam, the Netherlands
frasincar@ese.eur.nl

Maria Mihaela Trușcă
Bucharest Univ. of Economic Studies
Bucharest, Romania
maria.trusca@csie.ase.ro

ABSTRACT

The Aspect-Based Sentiment Classification (ABSC) models often suffer from a lack of training data in some domains. To exploit the abundant data from another domain, this work extends the original state-of-the-art LCR-Rot-hop++ model that uses a neural network with a rotatory attention mechanism for a cross-domain setting. More specifically, we propose a Domain-Independent Word Selector (DIWS) model that is used in combination with the LCR-Rot-hop++ model (DIWS-LCR-Rot-hop++). It uses attention weights from the domain classification task to determine whether a word is domain-specific or domain-independent, and discards domain-specific words when training and testing the LCR-Rot-hop++ model for cross-domain ABSC. Overall, our results confirm that DIWS-LCR-Rot-hop++ outperforms the original LCR-Rot-hop++ model under a cross-domain setting in case we impose a domain-dependent threshold value for deciding whether a word is domain-specific or not. For a target domain that is highly similar to the source domain, we find that a moderate attention threshold yields the best performance, while a target domain that is dissimilar requires a high attention threshold. Also, we observe information loss when we impose a too strict restriction and classify a small proportion of words as domain-independent.

CCS CONCEPTS

• **Information systems** → **Sentiment analysis**; *Information extraction*; Web mining;

KEYWORDS

ABSC, cross-domain ABSC, domain-specific word masking, attention threshold, DIWS-LCR-Rot-hop++

ACM Reference Format:
Junhee Lee, Flavius Frasincar, and Maria Mihaela Trușcă. 2023. A Cross-Domain Aspect-Based Sentiment Classification by Masking the Domain-Specific Words. In *The 38th ACM/SIGAPP Symposium on Applied Computing (SAC '23), March 27-31, 2023, Tallinn, Estonia.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3555776.3577633

1 INTRODUCTION

The evolution of the Web has changed how people think and make decisions. Furthermore, the recent development of social media and e-commerce platforms has opened the forum for people to exchange their opinions on social events or products and services on the market. As a result, the number of opinionated texts on the Web and the importance of understanding them is rapidly growing. However, it is difficult to summarize this public opinion as an enormous number of opinionated texts and reviews are available. To solve this issue, the current research has proposed a Natural Language Processing (NLP) task so-called Sentiment Analysis (SA) that identifies the overall sentiment of a given text [11].

One of the branches of SA is Aspect-Based Sentiment Analysis (ABSA). It consists of two tasks: Aspect Detection (AD), which identifies the aspect in the text and Aspect-Based Sentiment Classification (ABSC) which determines the sentiment about the previously found aspect [1]. This research focuses on ABSC. In practical applications, it is difficult to train ABSC models to a sufficient extent, because of the limited number of input sentence data. This issue is prominent in certain domains while other domains have a sufficient amount of data. To exploit this data imbalance, a number of approaches have been proposed by various studies. For example, [17] fine-tunes the upper layers of LCR-Rot-hop++ to increase cross-domain adaptability. Also, [19] introduces the BERTMasker algorithm that transforms the input sentences into domain-invariant sentences by masking the domain-related words and training the model using domain-agnostic words. Additionally, [9] extends the LCR-Rot-hop++ model with Domain Adversarial Training (DAT) method to construct a cross-domain DAT-LCR-Rot-hop++ model.

Nevertheless, the first fine-tuning approach partially requires the sentiment-labeled target domain data. Similarly, the BERTMasker model performs the best when it utilizes part of the sentiment-labeled target domain data. Also, BERTMasker cannot process the ABSC task. Thus, these models are not suitable for a cross-domain ABSC where sentiment-labeled data are not available in a domain of our interest. To solve this issue, we propose a Domain-Independent Word Selector (DIWS) model and apply it to the LCR-Rot-hop++ model (DIWS-LCR-Rot-hop++). This model utilizes attention weights from a domain classification task to decide whether a word is domain-specific or domain-independent. It only uses domain-independent words to train and test the LCR-Rot-hop++ model. Unlike fine-tuned LCR-Rot-hop++ [17] and BERTMasker [19], we can train the model only using data from other domains with sufficient sentiment-labeled texts. As DAT-LCR-Rot-hop++ shares the same advantage, we use it as a benchmark model to assess the performance of the proposed DIWS-LCR-Rot-hop++ model.

SAC '23, March 27-31, 2023, Tallinn, Estonia

ACM ISBN 978-1-4503-9517-5/23/03…$15.00
https://doi.org/10.1145/3555776.3577633

In this paper, we apply the DIWS model to the LCR-Rot-hop++ model yielding the DIWS-LCR-Rot-hop++ model. To this end, we aim to verify whether the combination of DIWS and LCR-Rot-hop++ outperforms the naive cross-domain ABSC performance of LCR-Rot-hop++, where we train and test the model with a completely different domain data without any adjustment to LCR-Rot-hop++. Furthermore, to verify the degree of effectiveness of discarding the domain-specific words, we compare the accuracy level under different strictness levels of deciding whether a word is domain-specific or domain-agnostic. More specifically, we incrementally increase the proportion of discarded words by relaxing the discarding threshold and verifying the trade-off between domain adaptability and information loss. Additionally, we compare against the accuracy of the DAT-LCR-Rot-hop++ model and the original LCR-Rot-hop++ model under a cross-domain setting to assess the performance of our proposed model.

This research contributes to the current literature by proposing a method named Domain-Independent Word Selector (DIWS) to better train an existing state-of-the-art ABSC model (LCR-Rot-hop++) for cross-domain sentiment analysis tasks. Moreover, this proposed model is not only applicable to ABSC but also to other types of SA, which implies its wide applicability in the field of sentiment analysis. To the best of our knowledge, the idea of discarding the domain-specific words while training a deep learning method for the ABSC task is new. The Python source code and data are available at https://github.com//ejoone/DIWS-ABSC.

The structure of the rest of the paper is as follows. In Section 2, we introduce the methodologies for ABSC and cross-domain SA in the current literature in detail and discuss their relevance to our research. In Section 3, we introduce the used datasets and the cleansing process. In Section 4, we elaborate on the theoretical framework, structure and mathematical formulations of the models that this paper investigates. In Section 5, we display the results and make comparisons between competing methods to answer the research question. Last, Section 6 provides a summary of the findings, introduces the theoretical and practical implications, discusses the limitations of our research, and proposes future research ideas.

2 RELATED WORK

In this section, we present work related to our research. First, in Subsection 2.1, we present approaches for ABSC. After that, in Subsection 2.2, we describe solutions for cross-domain sentiment classification.

2.1 Aspect-Based Sentiment Classification

There exist two major approaches to ABSC: knowledge-based and machine learning-based. This paper focuses on the second approach which uses machine learning algorithms. In detail, the neural network models and attention models are widely used. For example, [3] introduces Recursive Neural Networks (RecNNs) to this field by proposing an Adaptive Recursive Neural Network (AdaRNN). Also, Recurrent Neural Network (RNN) is one of the popular methods in SA [14, 20]. However, the main drawback of an ordinary RNN methodology is the long-term dependency problem, which refers to the tendency that the prior information to be dissolved when the input sequence is too long [7]. To address this issue, [7] suggests a special type of RNN model called Long Short-Term Memory (LSTM). Unlike traditional RNN models, LSTM employs additional gate nodes to control the information transfer between hidden layers. The authors allow LSTM to efficiently learn long-term relationships of data. Nonetheless, LSTM processes the information sequentially, which leads to a tendency that LSTM output converges to the latest input pattern. To address this concern, [5] suggests a bidirectional LSTM (bi-LSTM). It adds a reverse direction LSTM layer to the original LSTM network and uses both forward and backward LSTM layers to obtain a final result.

Furthermore, [23] proposes a Left-Center-Right separated neural network with Rotatory attention (LCR-Rot) that demonstrates high performance when an aspect contains multiple words by capturing the contextual information around the aspect. LCR-Rot is an extension of bi-LSTM and it utilizes three separate bi-LSTM networks, which correspond to the left context, target aspect, and right context, respectively. Also, rotatory attention helps to better model the relationship between the target aspect and left/right context, which allows the model to capture the most important words. [23] has confirmed that LCR-Rot outperforms other LSTM-based models. Additionally, LCR-Rot-hop is an extension of LCR-Rot proposed by [18]. It iterates the rotatory attention mechanism multiple times as it better exploits the interactions between the target aspects and right/left contexts. To even better represent the contextual information, [16] proposes LCR-Rot-hop++ which replaces non-contextual word embeddings of LCR-Rot-hop (GloVe) to a contextual word embeddings (BERT). Also, it adds an extra attention layer to obtain hierarchical attention. [16] has shown that LCR-Rot-hop++ in combination with a domain ontology (HAABSA++) outperforms other models on ABSC. In this research, we focus on LCR-Rot-hop++ and aim to incorporate the DIWS model inspired by the BERTMasker network of [19] to extend LCR-Rot-hop++ to the cross-domain setting.

2.2 Cross-Domain Sentiment Classification

Cross-domain sentiment analysis aims to solve the insufficient training data problem in one domain by leveraging the data from other domains. Unsupervised domain adaptation is one of the approaches to address the training data shortage problem. [24] proposes a representation learning model that selects important domain-independent pivot words. Also, [10] identifies pivots using a hierarchical attention transfer mechanism. Moreover, [6] and [21] extend the domain adaptation to the multi-domain setting.

Another approach to cross-domain sentiment analysis is the shared-private framework [2]. This approach is based on the reasoning that removing the domain-specific tokens would improve the domain-invariance of the input sentence. Hence, the gradient reversal layer is included before the domain classification step and it helps to select the tokens that reduce the performance of the domain classification task and consider corresponding words as domain-agnostic words.

On the other hand, [22] uses an attention mechanism to select the domain-specific information from the shared sentence representation of the input text. This framework is called shared encoder with domain-aware aggregation [19]. To take advantage of the shared-private and shared encoder with domain-aware aggregation paradigms, [19] proposes the BERTMasker model that combines these two frameworks. [19] has demonstrated that the BERTMasker outperforms existing models in both cross-domain and multi-domain settings.

Nevertheless, it is not possible to fully utilize the BERTMasker network in a cross-domain setting. The BERTMasker model consists of two parts: shared and private. The shared part masks the domain-specific tokens and uses the unmasked words to train the sentiment classification model so that it can process the cross-domain sentiment analysis. On the other hand, the private part uses the masked tokens to learn the domain-specific sentiment via an attention mechanism using training data of the target domain. The private part may effectively enhance the performance in the multi-domain setting because it is possible to use the labeled target domain data to train the private part of the model. However, in the cross-domain setting, BERTMasker cannot utilize its private part since the target sentiment labels are unavailable.

Moreover, BERTMasker is not designed for the ABSC task. Several attempts have been made to perform ABSC under the multi-domain and cross-domain settings. For example, [17] applies cross-domain fine-tuning to LCR-Rot-hop++, which is an ABSC model. More specifically, the authors fine-tune the upper layers of LCR-Rot-hop++, because the upper layers contain more domain-specific information while the lower layers represent general language characteristics [17]. However, the fine-tuning procedure requires the training data with sentiment labels from a target domain. Thus, it is a multi-domain ABSC model and we cannot directly compare this model to the cross-domain DIWS-LCR-Rot-hop++ model.

Additionally, [9] suggests the DAT-LCR-Rot-hop++ model that combines Domain Adversarial Training (DAT) [4] with LCR-Rot-hop++ so that it can perform cross-domain ABSC. Unlike [17], it does not require training data from a target domain. It replaces the final Multi-Layer Perceptron (MLP) layer with a domain adversarial component, which consists of two feed-forward MLPs. One is a domain discriminator with a gradient reversal layer. It allocates higher importance to the domain-agnostic words that cannot classify domain well. The other one is a class discriminator that aims to predict the sentiment label of an aspect in the sentence.

To continue exploring cross-domain aspect-based sentiment analysis, this research exploits the attention mechanism to select the domain-independent words from the input sequence. The DIWS-LCR-Rot-hop++ and DAT-LCR-Rot-hop++ models are based on the same reasoning that paying less or even no attention to the domain-specific words that are crucial for a domain classification task would improve the cross-domain performance of ABSC. However, there are some differences between the two models. First, the DIWS-LCR-Rot-hop++ model sequentially trains the DIWS component and LCR-Rot-hop++ component while DAT-LCR-Rot-hop++ jointly optimize the LCR-Rot-hop++ and domain class discriminator by letting parameters from LCR-Rot-hop++ affect the discriminator loss. Unlike DAT-LCR-Rot-hop++, in DIWS-LCR-Rot-hop++, DIWS parameters and LCR-Rot-hop++ parameters do not affect each other. Second, DIWS-LCR-Rot-hop++ discretely excludes domain-specific words that pass a certain attention threshold but DAT-LCR-Rot-hop++ allocates less importance to the domain-specific words rather than discarding them.

3 DATA

This research uses review data in five different domains to execute our proposed sentiment analysis. We summarize the used domains, datasets, sample size, and the distribution of sentiments in Table 1.

Table 1: Distribution of sentiment polarities.

Data	Size	Negative		Neutral		Positive	
		Freq.	%	Freq.	%	Freq.	%
Hotel [12]	264	55	21	10	4	199	75
DVD Player [8]	313	172	55	0	0	141	45
Digital Camera [8]	395	74	19	0	0	321	81
MP3 Player [8]	676	262	39	0	0	414	61
Cell Phone [8]	284	70	25	0	0	214	75

Note that the MP3 player review data has the maximum sample size while other domain data such as hotel, DVD player, digital camera, and cell phone contains a relatively small amount of samples. Hence, we use MP3 player review data to train our proposed cross-domain DIWS-LCR-Rot-hop++ model. Such domain is called the source domain. On the other hand, data in other domains are used to test the performance of trained cross-domain DIWS-LCR-Rot-hop++. Such domains are called target domains. Note that DIWS-LCR-Rot-hop++ requires a pair of one source domain and one target domain to train the model and assess its performance. We fix the MP3 player as a source domain for every pair. Hence, this research examines the performance of DIWS-LCR-Rot-hop++ for the following domain combinations: MP3 Player-Hotel, MP3 Player-DVD Player, MP3 Player-Digital Camera, and MP3 Player-Cell Phone.

For robustness of the experiment, we remove some samples in case of the presence of implicit aspects. An implicit aspect refers to the situation where the aspects appear as non-noun words and are implied in the sentence. We removed the samples with such characteristics as the machine learning algorithm cannot process such data [17]. Table 2 shows the results of the cleansing process.

Table 2: Cleansed datasets.

Domain	Removed: implicit aspects (%)
Hotel	22.1
DVD Player	27.4
Digital Camera	19.2
MP3 Player	20.3
Cell Phone	15.9

4 FRAMEWORK

Our proposed DIWS-LCR-Rot-hop++ model uses DIWS module to identify and discard domain-dependent words from the original input text and process the transformed text using LCR-Rot-hop++. First, in Subsection 4.1, we explain the overall structure of the DIWS-LCR-Rot-hop++. Second, in Subsection 4.2, we elaborate on the DIWS model for the cross-domain sentiment analysis task. Last, in Subsection 4.3, we introduce the LCR-Rot-hop++ model.

4.1 DIWS-LCR-Rot-hop++ Structure

The overall structure of DIWS-LCR-Rot-hop++ is as follows. First, DIWS models the input sentence using pre-trained BERT and obtains corresponding word embeddings. Second, it processes domain classification tasks for both source and target domains via the feed-forward attention layer. In this process, attention weights are computed by a softmax function and we take a linear combination of the attention weights and hidden representation of sentence words as sentence representation. This linear combination is fed into the sigmoid activation function and we obtain the model's prediction for the domain of the input sentence. Note that the optimal attention weights of each word in an input sentence are determined by gradient descent and back-propagation algorithm. Afterward, we select the domain-independent words by discarding the words that have attention weights higher than a certain attention threshold. We classify such words as domain-specific words. This step is based on the reasoning that the word with high attention weight has a high contribution to the domain classification task, and such words are domain-specific words that specifically appear in a certain domain.

After we identify domain-independent and domain-related words, we move on to the LCR-Rot-hop++ part of the model and compute the final sentiment prediction probabilities. Figure 1 visualizes the algorithm of the model and Figure 2 displays the graphical overview.

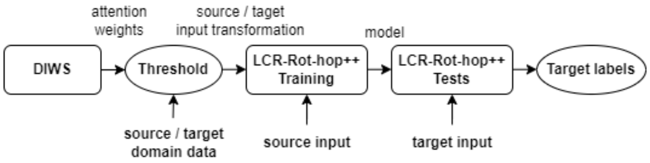

Figure 1: Overall representation of the DIWS-LCR-Rot-hop++ model.

4.2 Domain-Independent Word Selector

This section explains the mathematical formulations of Domain-Independent Word Selector (DIWS). Consider a sequence of BERT word embedding $\{h_1, h_2, ..., h_N\}$, transformed from input sentence $X = \{x_1, x_2, ..., x_N\}$. The preliminary attention score for the i^{th} word is:

$$\underset{1\times1}{\alpha_i} = \underset{1\times d}{h_i}^T \underset{d\times1}{V}, \tag{1}$$

where the $V \in \mathbb{R}^d$ is a context vector that is used as a query vector to find a word that is more important and informative for an accurate domain classification. We process these attention scores with the softmax function to obtain corresponding attention weights for every $i = 1, ..., N$:

$$\underset{1\times1}{\alpha_i} = \frac{exp(\underset{1\times1}{\alpha_i})}{\sum_{i=1}^{N} exp(\underset{1\times1}{\alpha_j}).} \tag{2}$$

We use this softmax operation again in LCR-Rot-hop++ and refer to it as the $softmax(\cdot)$ function later. Next, we represent the input sentence with length N as a weighted average of word embeddings h_i by their corresponding attention weights a_i where $i = 1, ..., N$:

$$\underset{d\times1}{h} = \sum_{i=1}^{N} \underset{1\times1}{\alpha_i} \times \underset{d\times1}{h_i}. \tag{3}$$

This process refers to the linear combination layer in Figure 2. Finally, h is fed to the fully connected layer in Equation 4 to produce polarity score s and it is fed into a sigmoid function in Equation 5 to produce prediction probability p:

$$\underset{1\times1}{s} = \underset{1\times d}{h}^T \underset{d\times1}{W} + \underset{1\times1}{d}, \tag{4}$$

$$\underset{1\times1}{p} = \frac{1}{1 + exp(-\underset{1\times1}{s})}, \tag{5}$$

where $W \in \mathbb{R}^d$ is a weight vector and b is a bias term. We train the values of the parameters such as weight vector and bias term during the training phase to minimize the loss. The loss function is the binary cross-entropy:

$$\underset{1\times1}{L_{DIWS}} = -\sum_{j=1}^{M+P} (\underset{1\times1}{y_j}\, log(\underset{1\times1}{p^{(j)}}) + (1 - \underset{1\times1}{y_j}) log(1 - \underset{1\times1}{p^{(j)}})), \tag{6}$$

where y_j is an actual binary domain label corresponding to j^{th} input sentence and $p^{(j)}$ refers to j^{th} domain prediction probability when there exists M and P (number of) sentences in the source and target domain, respectively.

The attention weights capture the relative importance of words in its input sentence for predicting the correct domain [15]. Based on this interpretation, we assume that the word with higher attention weights within the sentence is more domain-related.

In this research, we test different threshold values between domain-specific words with high attention and domain-agnostic words with low attention. Let us define the set of threshold percentiles as $K = \{10, 20, 30, 40, 50, 60, 70, 80, 90, 100\}$. For every input sentence,

$$d_i = \begin{cases} \text{domain-specific}, & \text{if } \alpha_i \geq Q_K, \\ \text{domain-independent}, & \text{otherwise}, \end{cases} \tag{7}$$

where d_i is a domain-relatedness label associated with every x_i, and Q_K refers to a K^{th} percentile value of the attention weights for every word in an input sentence. We discard domain-specific words to construct domain-invariant input sentence representations for both source and target domains, so that we can perform domain-independent training and domain-independent testing using LCR-Rot-hop++. Last, we fed the transformed input sentences to the LCR-Rot-hop++ model.

4.3 LCR-Rot-hop++

The LCR-Rot-hop++ model uses three bi-LSTM networks and a rotatory, hierarchical attention mechanism to classify the sentiment of a given aspect. This section describes the LCR-Rot-hop++ model and its mathematical formulations.

LCR-Rot-hop++ uses a sentence X as its input, where $x_{target} = \{x_1^t, ..., x_T^t\}$ represents the set of T words describing an aspect of the sentence X. Then it splits X into three separate components,

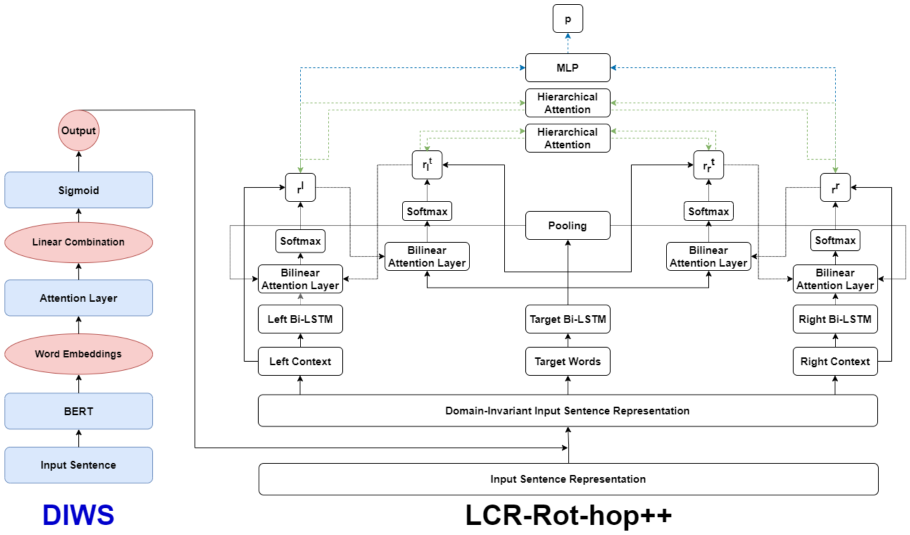

Figure 2: Detailed representation of the DIWS-LCR-Rot-hop++ model.

namely a left context, a target, and a right context. The left context is a set of words that appear before the target x_{target}, and the right context is a set of words that appear after the target x_{target}.

A rotatory attention mechanism aims to distinguish the most important words in the left context, target, and the right context for determining the sentiment using a two-step procedure. First, it calculates the target-aware left and right context representation r_l^t and r_r^t by considering the information in the target representation r^t. The initial value of r^t is determined by the pooling operation of the hidden states of the target, which is the output of the target bi-LSTM module:

$$\underset{2d\times1}{r^t} = pooling([\underset{2d\times1}{h_1^t}, ..., \underset{2d\times1}{h_T^t}]) \tag{8}$$

r^t is then fed in the bilinear attention layer together with $[h_1^l, ..., h_L^l]$ and $[h_1^r, ..., h_R^r]$ separately. We illustrate the mathematical formulation for the left context, but the same logic applies to the right context. The output of the bilinear attention layer is obtained by multiplying the transposed h_i^l, weights (W_c^l), and r^t, and adding the bias term (b_c^l), and input the result to the $tanh$ activation function for every $i = 1, ..., L$:

$$\underset{1\times1}{f(h_i^l, r^t)} = tanh(\underset{1\times2d}{h_i^{lT}} \times \underset{2d\times2d}{W_c^l} \times \underset{2d\times1}{r^t} + \underset{1\times1}{b_c^l}) \tag{9}$$

Then we process the obtained score with the softmax function to obtain the attention score α_i^l, and get the target-aware left context representation r^l by computing a weighted average of the left context hidden states in terms of the attention scores:

$$\underset{1\times1}{\alpha_i^l} = softmax(\underset{1\times1}{f(h_i^l, r^t)}) \tag{10}$$

$$\underset{2d\times1}{r^l} = \sum_{i=1}^{L} \underset{1\times1}{\alpha_i^l} \times \underset{2d\times1}{h_i^l} \tag{11}$$

Unlike the first step, the second step uses r^l and r^r to construct the left and right context-aware target representations r_l^t and r_r^t, respectively. The logic is the same as in the first procedure, while we no longer need to use the pooling operation as we already have r^l and r^r from the first step.

We obtain four representations r^l, r^r, r_l^t, and r_r^t as outputs of the rotatory attention mechanism. Two context representations (r^l, r^r) and two target representations (r_l^t, r_r^t) are then separately weighted by a hierarchical attention mechanism and updated to encode global information around the input sentence, not only the local, left, target, or right contextual information. The logic is similar to the previous process. For example, the context representations are updated as follows:

$$\underset{1\times1}{f(r^l, r^r)} = tanh(\underset{1\times2d}{r^{lT}} \times \underset{2d\times2d}{W_h^c} \times \underset{2d\times1}{r^r} + \underset{1\times1}{b_h^c}) \tag{12}$$

$$\alpha_i^{l'} = softmax(f(r^l, r^r)) \tag{13}$$

$$\underset{2d\times1}{r_i^{l'}} = \sum_{i=1}^{L} \underset{1\times1}{\alpha_i^l} \times \underset{2d\times1}{r_i^l} \tag{14}$$

Using the same logic, we obtain the updated representations $r^{l'}$, $r^{r'}$, $r_l^{t'}$, and $r_r^{t'}$. [18] argues that it is optimal to repeat this procedure three times. We inherit this idea and repeat this mechanism for three hops. The final four representations are concatenated and processed by a Multi-Layer Perceptron (MLP). The mathematical notation $\oplus$ in Equation 15 denotes vector concatenation. Last, we take softmax to calculate the final prediction probability for each sentiment polarity (p), which is a 3-dimensional vector as we consider three sentiment polarities, i.e., positive, neutral, and negative:

$$\underset{8d\times1}{r} = \underset{2d\times1}{r^{l'}} \oplus \underset{2d\times1}{r^{r'}} \oplus \underset{2d\times1}{r_l^{t'}} \oplus \underset{2d\times1}{r_r^{t'}} \tag{15}$$

$$\underset{3\times1}{p} = softmax(MLP(r)) \tag{16}$$

We calculate the final loss function for LCR-Rot-hop++ sentiment classification by taking the cross-entropy of the predicted sentiment and actual sentiment label of the j^{th} sentence denoted as a_j over

M input sentences:

$$L_{sc} = -\sum_{j=1}^{M} \underset{3\times1}{a_j}\ log(\underset{3\times1}{p^{(j)}}) + \lambda||\theta_2||^2 \tag{17}$$

where $p^{(j)}$ is the prediction probablity of the j^{th} sentence, $||\theta_2||^2$ is a $L2$-norm regularization term, which determines the penalty of having a certain parameter set, and λ is a weight for this term.

5 EVALUATION

In this section, we present the result of our evaluation. First, in Subsection 5.1, we evaluate the performance of the domain classification. Then, in Subsection 5.2, we present the performance of the aspect-based sentiment classification on the target domain. Next, in Subsection 5.3, we compare the results of our proposed model with the ones of DAT-LCR-Rot-hop++. Last, in Subsection 5.4, we give insights in the obtained results.

5.1 Domain Classification Performance

Table 3 shows the domain classification accuracy of the DIWS model. The training sample consists of 80% of the randomly mixed source and target domain data, and the testing sample consists of the remaining 20% of the data.

Table 3: Domain classification accuracy of DIWS.

Source-target domain	Domain classification test accuracy
MP3 Player - DVD Player	0.824
MP3 Player - Digital Camera	0.884
MP3 Player - Hotel	0.979
MP3 Player - Cell Phone	0.824
Average	0.878

On average, the DIWS model can well classify the source domain and target domain with an average accuracy of 0.878. It implies the robustness of the attention weights from the DIWS model. In particular, the accuracy for the hotel domain is relatively high compared to the other domains. It signals that the difference between the target domain and the source MP3 player domain is greater for the hotel domain.

5.2 Aspect-Based Sentiment Classification Performance

We measure a test accuracy for different values of percentile threshold $K = \{10, 20, 30, 40, 50, 60, 70, 80, 90, 100\}$. Here, percentile threshold K implies that the there exists K% of the words with lower attention weight than the corresponding attention threshold. Figure 3 displays the change in test accuracy level as the threshold percentile increases. Note that $K = 100$ refers to the case that every word is classified as domain-agnostic regardless of their DIWS attention weights. Hence, it represents the original LCR-Rot-hop++ model that is purely trained by MP3 player domain data and tested on the DVD player data. This interpretation of $K = 100$ applies to all target domains.

For the DVD player domain, the DIWS-LCR-Rot-hop++ test accuracy varies from 55% to 71%. The lowest accuracy is colored gray and the highest accuracy is colored light gray. The model attains the lowest accuracy when $K = 10$ and attains the highest accuracy when $K = 70$. Also, there exists a general trend that the accuracy increases as we reduce the proportion of domain-specific discarded words, and achieves maximum accuracy when $K = 70$. The accuracy drops at $K = 80$ but bounces again as we reduce the discarded words to the extreme.

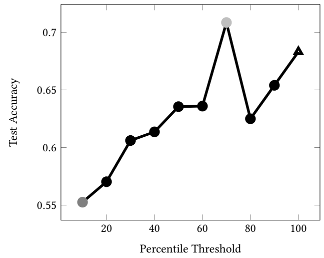

Figure 3: Test accuracy of DVD player domain data under DIWS-LCR-Rot-hop++ model trained with MP3 player domain data.

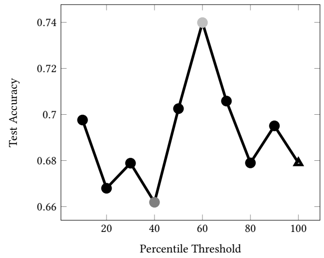

Figure 4: Test accuracy of digital camera domain data under DIWS-LCR-Rot-hop++ model trained with MP3 player domain data.

For the digital camera domain, the DIWS-LCR-Rot-hop++ test accuracy varies from 66% to 74%. The model attains the lowest accuracy when $K = 40$ and attains the highest accuracy when $K = 60$. Unlike the DVD player domain, DIWS-LCR-Rot-hop++ has a decent performance when we discard a large proportion of words. For instance, the accuracy gap between the maximum accuracy at $K = 60$ and accuracy for the low threshold values $K = 10, 20, 30$ is not as large as the DVD player domain. Additionally, after having the highest accuracy at $K = 60$, the accuracy diminishes as K increases to the extreme.

For the hotel domain, the DIWS-LCR-Rot-hop++ test accuracy varies from 63% to 73%. The model attains the lowest accuracy at $K = 100$ and attains the highest accuracy at $K = 20$. Unlike the other domains, DIWS-LCR-Rot-hop++ performs the best for the small K value ($K = 20$), and the accuracy decreases until the model attains the lowest accuracy at $K = 100$, although there are some local peaks at $K = 60$ and $K = 80$.

For the cell phone domain, the test accuracy varies from 64% to 77%. The model attains the minimum accuracy at $K = 80$ and attains the highest accuracy at $K = 90$, without a clear trend. DIWS-LCR-Rot-hop++ has a sudden dip and spike at $K = 80$ and $K = 90$,

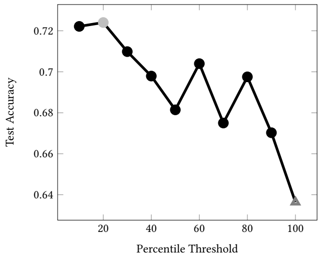

Figure 5: Test accuracy of hotel domain data under DIWS-LCR-Rot-hop++ model trained with MP3 player domain data.

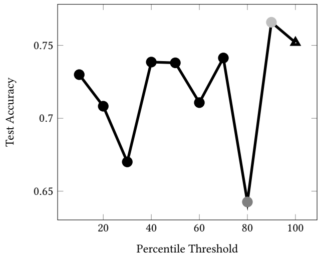

Figure 6: Test accuracy of cell phone domain data under DIWS-LCR-Rot-hop++ model trained with MP3 player domain data.

respectively. After the global maximum accuracy, the accuracy decreases again at $K = 100$.

5.3 Comparison with DAT-LCR-Rot-hop++

To assess the overall performance of the DIWS-LCR-Rot-hop++, we compare the results to the DAT-LCR-Rot-hop++ model [9]. For each target domain, we choose the threshold value K that attains maximum accuracy. Table 4 displays the test accuracy of the two models.

Table 4: The accuracy comparison between DIWS-LCR-Rot-hop++ (DIWS++) and DAT-LCR-Rot-hop++ (DAT++).

Source-target domain	DIWS++ accuracy	DAT++ accuracy
MP3 Player - DVD Player	**0.708**	0.470
MP3 Player - Digital Camera	**0.740**	0.693
MP3 Player - Hotel	0.724	**0.751**
MP3 Player - Cell Phone	**0.766**	0.742

Note: We train the DAT-LCR-Rot-hop++ with our selection of datasets using its source code as the original paper does not use our datasets.

On average, DIWS-LCR-Rot-hop++ outperforms DAT-LCR-Rot-hop++ in three out of four target domains. The hotel domain is the only target domain that DAT-LCR-Rot-hop++ outperforms DIWS-LCR-Rot-hop++. For the camera, hotel, and cell phone domains, the performance difference is small, as it ranges from 0.024 to 0.047. On the other hand, the DVD player domain experience performance enhancement to a great extent (0.24). Overall, we conclude that the DAT-LCR-Rot-hop++ model improves the cross-domain ABSC performance compared to the DAT-LCR-Rot-hop++ model for the source and target domains that we use.

5.4 Insights

In general, DIWS-LCR-Rot-hop++ performs better than the random guessing baseline (0.33 if there are three sentiment classes; 0.5 if there are two sentiment classes) in all domains. For the DVD player and cell phone domains, it even outperforms the majority guessing baseline (see Table 1 for the distribution of the sentiment) without having the information about the sentiment distribution. Furthermore, it improves the cross-domain performance of the original LCR-Rot-hop++ model ($K = 100$) and the DAT-LCR-Rot-hop++ model if we choose the optimal threshold K for each target domain. Next, let us investigate the overall pattern of accuracy level as K increases and the reasoning behind the results. First, the results show that the model performs the best when we discard 10% to 40% of the domain-specific words for three out of the four domains (DVD player, camera, and cell phone). If we discard too many words which correspond to the low values of K, the accuracy is below average for most of the data except for the hotel domain. This is due to the excessive information loss. Accordingly, discarding most of the domain-specific words that implies keeping the articles (a/an/the) or linking verbs (be/is/are) would enhance domain-invariance of the input texts, but it also makes the remaining sentence useless for ABSC.

On the other hand, the hotel domain obtains its maximum accuracy at $K = 20$, where we discard 80% of the words. This difference is due to the difference in closeness between the source domain and the target domain. Note that the hotel domain is even more distinct from the source MP3 player domain compared to the other target domains such as DVD player, digital camera, and cell phone. The high domain-classification accuracy for the hotel domain (0.979) in Table 3 supports this claim, because it would be easier to classify the domain if there exists a large difference between the target and source domains. Thus, for the hotel domain, the accuracy gained from discarding the domain-specific words outweighs the accuracy drop due to the information loss.

To conclude, the results show that discarding domain-specific words leads DIWS-LCR-Rot-hop++ to perform better than the original LCR-Rot-hop++ model under the cross-domain ABSC task, while the optimal proportion of remaining words after the dropout depends on the degree of closeness between the source domain and the target domain. In general, discarding an excessive proportion of words even further worsens the performance of DIWS-LCR-Rot-hop++ compared to the original LCR-Rot-hop++ where we do not discard any of the words. These findings answer the research question. Discarding domain-specific words indeed improves the performance of cross-domain aspect-based sentiment analysis when we discard 10% to 40% of the words if the target domain and source domain are not very different. If we drop too many words, the model experiences a performance drop due to the information loss. However, if we recognize that the source and target domains are distinct from each other, we should discard a large percentage of words (80% for the hotel domain) as the accuracy gained from discarding the domain-specific words outweighs the information loss effect. The domain classification accuracy from DIWS would be a good indication of whether a target domain is very different from a source domain or not.

6 CONCLUSION

To apply the state-of-the-art LCR-Rot-hop++ model to the cross-domain setting, this work proposes the DIWS model to select and discard the domain-specific words. The proposed model for cross-domain ABSC is the DIWS-LCR-Rot-hop++ model. It utilizes a domain classification architecture with a feed-forward attention layer to filter out the domain-specific words with attention weights higher than a certain threshold. Then we analyze the performance of our proposed model for 10 different threshold values. Based on the experiments on 5 datasets, we conclude that without any sentiment label of the target domain data, our model effectively enhances the accuracy by discarding the domain-specific words from source and target domain data.

Furthermore, we have found that there is a danger of information loss and thus we should select the threshold between domain-specific words and domain-agnostic words carefully. In addition, the results imply that the degree of difference between the source domain and target domain affects the performance of the DIWS-LCR-Rot-hop++ model for different threshold values.

Nevertheless, there exist limitations to our research. First, due to the lack of computational power of the testing PC environment, we could not apply DIWS-LCR-Rot-hop++ to the large popular datasets in the field of ABSC. For example, such data includes restaurant domain data and laptop domain data from SemEval 2014 [13]. Second, the performance gain from DIWS-LCR-Rot-hop++ is not always positive compared to the original LCR-Rot-hop++. If we do not use the optimal threshold value, the accuracy of our model can be even less than the original model. Therefore, we advise users to run the DIWS-LCR-Rot-hop++ using different threshold values and select the optimal one for the final prediction for every source-target domain combination. Last, the DIWS-LCR-Rot-hop++ model sequentially trains the DIWS component and LCR-Rot-hop++ component. This sequential training process may prevent the model to find the global optimal parameter values during optimization.

Several further research directions are available on this topic. First, the sequential training processes can be merged into a simultaneous optimization in which the final loss function is a sum of the DIWS loss function and LCR-Rot-hop++ loss function. Second, it is possible to directly utilize the optimal domain-classification attention weights by allocating lower importance to words in an input sentence by their attention weights. For example, words with high attention weights are considered less during the LCR-Rot-hop++ training because they are likely to be domain-specific. Finally, it is possible to extend the model to the multi-domain setting where the sentiment-labeled target domain data is partially available. In this case, one can exploit the shared-private framework and thus expect even higher performance results.

7 BIBLIOGRAPHY

[1] Gianni Brauwers and Flavius Frasincar. 2022. A Survey on Aspect-Based Sentiment Classification. *Comput. Surveys* (2022). https://doi.org/10.1145/3503044

[2] Xilun Chen and Claire Cardie. 2018. Multinomial Adversarial Networks for Multi-Domain Text Classification. In *2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2018)*. ACL, 1226–1240.

[3] Li Dong, Furu Wei, Chuanqi Tan, Duyu Tang, Ming Zhou, and Ke Xu. 2014. Adaptive recursive neural network for target-dependent Twitter sentiment classification. In *52nd Annual Meeting of the Association for Computational Linguistics (ACL 2014)*. ACL, 49–54.

[4] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor S. Lempitsky. 2017. Domain-Adversarial Training of Neural Networks. In *Domain Adaptation in Computer Vision Applications*. Springer, 189–209.

[5] Alex Graves and Jürgen Schmidhuber. 2005. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Networks* 18, 5-6 (2005), 602–610.

[6] Jiang Guo, Darsh J. Shah, and Regina Barzilay. 2018. Multi-Source Domain Adaptation with Mixture of Experts. In *2018 Conference on Empirical Methods in Natural Language Processing (EMNLP 2018)*. ACL, 4694–4703.

[7] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation* 9, 8 (1997), 1735–1780.

[8] Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2004)*. ACM, 168–177.

[9] Joris Knoester, Flavius Frasincar, and Maria Truşcă. 2022. Domain Adversarial Training for Aspect-Based Sentiment Analysis. In *23rd International Conference on Web Information Systems (WISE 2022) (LNCS)*. Springer.

[10] Zheng Li, Ying Wei, Yu Zhang, and Qiang Yang. 2018. Hierarchical Attention Transfer Network for Cross-Domain Sentiment Classification. In *32nd AAAI Conference on Artificial Intelligence (AAAI 2018)*. AAAI Press, 5852–5859.

[11] Bing Liu. 2020. *Sentiment Analysis - Mining Opinions, Sentiments, and Emotions* (second ed.). Cambridge University Press.

[12] Stephan Oepen, Marco Kuhlmann, Yusuke Miyao, Daniel Zeman, Silvie Cinková, Dan Flickinger, Jan Hajič, and Zdeňka Urešová. 2015. SemEval 2015 Task 18: Broad-Coverage Semantic Dependency Parsing. In *9th International Workshop on Semantic Evaluation (SemEval 2015)*. ACL, 915–926.

[13] Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014. SemEval-2014 Task 4: Aspect Based Sentiment Analysis. In *8th International Workshop on Semantic Evaluation (SemEval 2014)*. ACL, 27–35.

[14] Sebastian Ruder, Parsa Ghaffari, and John G. Breslin. 2016. A Hierarchical Model of Reviews for Aspect-based Sentiment Analysis. In *2016 Conference on Empirical Methods in Natural Language Processing (EMNLP 2016)*. ACL, 999–1005.

[15] Xiaobing Sun and Wei Lu. 2020. Understanding Attention for Text Classification. In *58th Annual Meeting of the Association for Computational Linguistics, (ACL 2020)*. ACL, 3418–3428.

[16] Maria Mihaela Truşcă, Daan Wassenberg, Flavius Frasincar, and Rommert Dekker. 2020. A hybrid approach for aspect-based sentiment analysis using deep contextual word embeddings and hierarchical attention. In *20th International Conference of Web Engineering (ICWE 2020) (LNCS)*, Vol. 12128. Springer, 365–380.

[17] Stefan van Berkum, Sophia van Megen, Max Savelkoul, Pim Weterman, and Flavius Frasincar. 2021. Fine-Tuning for Cross-Domain Aspect-Based Sentiment Classification. In *20th IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT 2021)*. ACM, 524–531.

[18] Olaf Wallaart and Flavius Frasincar. 2019. A Hybrid Approach for Aspect-Based Sentiment Analysis Using a Lexicalized Domain Ontology and Attentional Neural Models. In *16th Extended Sematic Web Conference (ESWC 2019) (LNCS)*, Vol. 11503. Springer, 363–378.

[19] Jianhua Yuan, Yanyan Zhao, and Bing Qin. 2022. Learning to share by masking the non-shared for multi-domain sentiment classification. *International Journal of Machine Learning and Cybernetics* (2022), 1–14.

[20] Meishan Zhang, Yue Zhang, and Duy-Tin Vo. 2016. Gated Neural Networks for Targeted Sentiment Analysis. In *30th AAAI Conference on Artificial Intelligence (AAAI 2016)*. AAAI Press, 3087–3093.

[21] Han Zhao, Shanghang Zhang, Guanhang Wu, José M. F. Moura, João Paulo Costeira, and Geoffrey J. Gordon. 2018. Adversarial Multiple Source Domain Adaptation. In *32nd International Conference on Neural Information Processing Systems (NIPS 2018)*. Curran Associates, 8568–8579.

[22] Renjie Zheng, Junkun Chen, and Xipeng Qiu. 2018. Same Representation, Different Attentions: Shareable Sentence Representation Learning from Multiple Tasks. In *27th International Joint Conference on Artificial Intelligence (IJCAI 2018)*. International Joint Conferences on Artificial Intelligence Organization, 4616–4622.

[23] Shiliang Zheng and Rui Xia. 2018. Left-center-right separated neural network for aspect-based sentiment analysis with rotatory attention. *arXiv preprint arXiv:1802.00892* (2018).

[24] Yftah Ziser and Roi Reichart. 2018. Pivot Based Language Modeling for Improved Neural Domain Adaptation. In *2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2018)*. ACL, 1241–1251.