information

MDPI

Article

A Review Structure Based Ensemble Model for Deceptive Review Spam

Zhi-Yuan Zeng [1], Jyun-Jie Lin [2], Mu-Sheng Chen [3], Meng-Hui Chen [2,*], Yan-Qi Lan [1] and Jun-Lin Liu [1]

1 School of Software, Nanchang University, Nanchang 33047, China
2 National Taipei University of Business, Jinan Rd., Zhongzheng District, Taipei 10051, Taiwan
3 School of Software, Jiangxi university of Science and Technology, Nanchang 330013, China
* Correspondence: bbb422@hotmail.com

Received: 22 April 2019; Accepted: 31 May 2019; Published: 17 July 2019

Abstract: Consumers' purchase behavior increasingly relies on online reviews. Accordingly, there are more and more deceptive reviews which are harmful to customers. Existing methods to detect spam reviews mainly take the problem as a general text classification task, but they ignore the important features of spam reviews. In this paper, we propose a novel model, which splits a review into three parts: first sentence, middle context, and last sentence, based on the discovery that the first and last sentence express stronger emotion than the middle context. Then, the model uses four independent bidirectional long-short term memory (LSTM) models to encode the beginning, middle, end of a review and the whole review into four document representations. After that, the four representations are integrated into one document representation by a self-attention mechanism layer and an attention mechanism layer. Based on three domain datasets, the results of in-domain and mix-domain experiments show that our proposed method performs better than the compared methods.

Keywords: spam review detection; ensemble learning; bidirectional long-Short term memory; self-attention mechanism; attention mechanism; representation learning

1. Introduction

Consumers' purchase behavior increasingly relies on online reviews. Accordingly, there are more and more deceptive reviews which are written to deceive consumers for commercial purpose. In order to make more profits, some merchants hire writers to write positive reviews to promote their products or write negative reviews to damage the business of their competitors [1]. With the spread and growth of deceptive reviews, more and more research [2–9] is focusing on the detection of deceptive comments.

To identify whether a review is deceptive or not can be regarded as a binary classification problem. The research on spam reviews was first investigated by Jindal and Liu [1]. Early representative works [2–5] generally extract features manually and use machine learning algorithms to solve the problem. As the neural networks model is widely used in natural language processing, more and more research [6,7] builds an end-to-end neural network model to extract the document representation from the review automatically which obtains the better classification results.

It is very difficult to identify deceptive comments. According to the experimental results of Ott et al. [2], the accuracy of three human judges is only 57.3%. But Li et al. [4] built a model using n-grams, part of speech, linguistic inquiry and word count (LIWC) as features, and SVM (Support Vector Machine), Bayes as the classifier, which has a much better performance than humans. Li et al. [6] and Ren et al. [7] built end-to-end neural networks models to extract the representation of the review and gain a better much result than the method carried out by Li et al. [4]. Their works indicate that the representation learned by neural networks can catch more information of a review than manually

extracted features. Compared to the representation extracted by neural networks, manually extracted features are low-dimensional and sparse. According to Ren et al. [6], it's difficult for us to extract features manually that can capture global semantic information over a sentence or discourse.

Although neural networks can learn complex nonlinear relationships from data, they have low bias and high variance, which means that they are sensitive to the statistical noise in the training data. It is easy for neural networks to overfit on small training data. However, the lack of annotated data is a critical problem in deceptive review spam [1], hence, it is important to make full use of the annotated data, and use some methods to improve the generalization performance of neural networks.

We compared the deceptive reviews with truthful reviews carefully and came up with following conclusions: (1) deceptive comments expressed stronger emotions than real comments, which is consistent with the conclusion of Li et al. [4]; (2) the strongest expression of emotion in a comment is at the beginning and the end; and (3) deceptive reviews often start or end with similar sentences, which may due to that deceptive reviews are usually created by dedicated writers, while the same person may create a large number of similar reviews. Table 1 shows some similar beginnings and ends of deceptive reviews.

Table 1. Similar beginnings and endings in deceptive reviews.

Similar Beginnings	My husband and I arrived for a 3-night stay for our 10th wedding anniversary.
	My husband and I stayed there when we went to visit my sister.
	My wife and I checked in to this hotel after a rough flight from Los Angeles.
Similar Endings	I look forward to many visits to Joe's in the future.
	I am looking forward to my next visit to Mike Ditka's—Chicago.
	We definitely will be returning to this restaurant in the near future.

According to the above discoveries, we divide a review into three parts: first sentence, middle context, and the last sentence, and propose an ensemble model based on such structure of the review. Firstly, we use bidirectional long short-term memory (BiLSTM) to encode the first sentence, middle context, last sentence, and the whole review into four independent document representations. As the representations obtained by the first sentence, middle context and last sentence only contained one part of the information of a review, we used the self-attention mechanism to integrate three local representations to a global representation which include all information of the review. Since the representation encoded by BiLSTM using the whole review also contains all information of the review, we used the attention mechanism to integrate two global representations into a final representation. Finally, the classification result was obtained through a fully-connected neural network based on the final representation.

We compared the proposed model with the standard benchmark [4] and the state-of-the-art [6] based on the standard dataset [4], which contains three domains (Hotel, Restaurant, Doctor). Results on in-domain and mix-domain experiments show that our model outperforms the compared methods.

The major contributions of the work presented in this paper are as follows:

We split a review into three parts: first sentence, middle context, and last sentence to highlight the first and last sentence, based on the discovery that the first and last sentence express stronger emotion than the middle context.

We used four independent bidirectional LSTM models to encode the first sentence, middle context, last sentence, and the whole review into four document representations. Rather than simply make an average of them, we integrated them using a self-attention mechanism layer and an attention mechanism layer, which can learn a better combination of them through backward propagation.

We verified the effectiveness of our method in three kinds of experiments, we compared it with the baseline method and visualized the weights in the attention mechanism, which showed that the weights of the first sentence and last sentence were significantly higher than middle context, as we expected.

2. Related Work

2.1. Classification of Deceptive Reviews

Research on spam reviews was first investigated by Liu et al. [1], who divide spam reviews into three categories: (1) unreal reviews (deceptive reviews); (2) reviews on brands; and (3) irrelevant reviews. They also conclude that it is easy to identify the spam reviews of the second and third category, but it is difficult to identify the first category, the deceptive review, because of the lack of annotated data. Current research for deceptive reviews is mainly based on the users' behavior and the text of reviews. The approach based on the user's behavior is focused on filtering strategies to withstand faulty or malicious behavior in networks [8–10]. The approach based on the text of reviews is focused on extracting effective features and take this problem as a classification task. In this paper, we mainly introduce the approach based on the text of reviews.

Ott et al. [2] created the first public deceptive review dataset by hiring online writers to write deceptive reviews. Their data included 400 deceptive reviews and 400 truthful reviews about hotels. Based on the data from Ott et al. [2], Feng et al. [11] applied context-free grammar parse trees to extract syntactic features to improve the performance of the model. Li et al. [5] proposed a topic model based on LDA for deceptive review detection. Xu and Zhao [12] exploited generative features to extract text features from the dependency parse tree. While Banerjee and Chua [13] proposed a language framework to analyze the differences between truthful and deceptive reviews in terms of their writing style and readability. In addition, Donato et al. [14] found that the character n-grams are better features than word n-gram features for the detection of opinion spam.

The dataset proposed by Ott is too small, therefore some approaches which use unsupervised or semi-supervised methods are applied to this problem. Donato et al. [15] employed PU-learning to the problem using unlabeled data. Hai et al. [16] developed a multi-task learning method based on logistic regression. Feng et al. [17] studied the distributions of rating scores and introduced strategies to create a dataset with pseudo-standard. Liu and Pang [18] trained multiple tree classifiers to generate labeled samples from unlabeled ones and train a neural network on the extended dataset.

Li et al. [4] collected another deceptive reviews dataset based on the work of Ott et al. [2], which contains three domains: hotel, restaurant, and doctor, and explored a general method to detect deceptive reviews. In this paper, we use the dataset proposed by Li et al. [4], because it is the largest dataset of deceptive review spam to our best knowledge. Based on this dataset, some neural networks models are proposed. Ren et al. [7], Li et al. [6] built hierarchical structure (sentence-document) models and used the attention mechanism to learn the representation of the review, which achieved better results than the baseline model proposed by Li et al. [4]. Sun et al. [19] proposed a convolutional neural network model to integrate the product related review features through a product word composition model.

This paper uses the neural network model to learn the document representation of the review. But to be different from Ren et al. [7] and Li et al. [6], we do not use the sentence-document structure. The structure of our model is based on the review of structures, we divide a review into three parts according to the idea that the beginning and end of a review are more important to detect a deceptive review, and stack LSTM models and attention mechanism to learn the representation of the review.

2.2. Ensemble Learning

The idea of ensemble learning is to build multiple weak models and integrate them together through some strategies to learn a stronger model. There are some popular methods in ensemble learning such as, bagging [20] and boosting [21]. Bagging is to randomly construct several groups of training samples to train several different models. And the independence of the model comes from the independence of the training data. Random forest [22] is a representative model that uses the bagging method. Boosting is to train a group of models iteratively, and change the distribution of the data

according to the results of the classification. AdaBoost [23] is a representative model that uses the boosting method.

Ensemble learning is very popular in tree models, and it is also commonly used in neural network models to improve the generalization ability of models. In addition to general methods such as bagging, there are some useful methods such as using different initialization parameters [24], different hyper-parameters [25] to train a group of models, and the models can be combined through weighted average or stacking models.

In this paper, we use four independent bidirectional LSTM model to encode the beginning, middle, end, and whole article of a review into four document representations based on the discovery that the beginning and end of a review is more important than middle context. To catch the information of four document representation, we use attention mechanism to integrate them into 1 document representation.

3. Materials and Methods

Based on the discovery that the first sentence and last sentence of the review is more important than the middle context, we split a review into three parts, as shown in Figure 1, and propose an ensemble model (RSBE) based on such structure. The model is composed of four Bidirectional LSTM encoders and two layers of attention mechanisms. The four bidirectional LSTM encoders encode the review's first sentence, middle context, last sentence, and whole text into four document representations. While representation 1, representation 2, and representation 3 represent the first sentence, middle context, and the last sentence respectively, and the representation4 represents the entire review. The next two layers of attention mechanism integrate representations 1–4 into a final document representation. In details, the self-attention mechanism integrates representations 1–3 (first sentence, middle context, and last sentence) into representation 5, while the attention mechanism integrates representation 4 and representation 5 to get the final representation. Finally, the classification results are obtained by a feedforward neural network. In the following sections, we will present the details of bidirectional LSTM encoder, self-attention mechanism, and the attention mechanism.

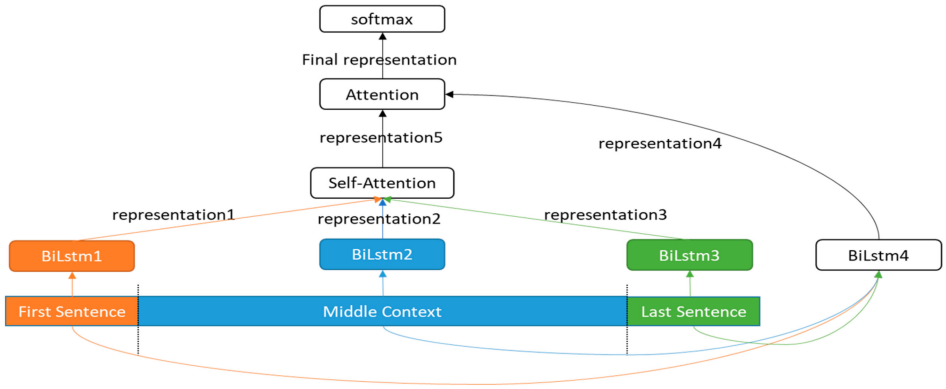


Figure 1. The ensemble model based on the structure of the review.

3.1. Bidirectional LSTM Encoder

The long-term short-term memory network (LTSM) [26], is commonly used to model sequences. LSTM is the special architecture of the recurrent neural network (RNN) [27], which is designed to solve the vanishing gradients problem of the RNN. The LSTM introduces the cell memory and gating mechanism based on the common RNN. The memory cell is designed to save memory and gradients of neurons across time. The input, forgetting, and output of the information in the memory cell is controlled by three adaptive gates (g_i, g_f, g_o) which are defined as Equations (1)–(3).

$$g_i = \sigma\left(x_j W^{x_i} + h_{j-1} W^{h_i}\right) \quad (1)$$

$$g_f = \sigma\left(x_j W^{x_f} + h_{j-1} W^{h_f}\right) \tag{2}$$

$$g_o = \sigma\left(x_j W^{x_o} + h_{j-1} W^{h_o}\right) \tag{3}$$

where x_j is the current input at position j in the sequence, and h_{j-1} is the state of the previous cell. g_i, g_f, g_o control the input, forgetting and output of the memory cell. The values of g_i, g_f, g_o are the linear combination of x_j *and* h_{j-1}, passed through a *sigmoid* activation function. The new state is the linear combination of x_j *and* h_{j-1} passed through a *tanh* activation function as shown in Equation (4)

$$z = \tanh\left(x_j W^{xz} + h_{j-1} W^{hz}\right) \tag{4}$$

z is then saved in the memory cell, but it does not replace the old value in the memory cell. The new memory cell is the linear combination of z and the old value. Equation (5) shows the update of the memory cell.

$$c_j = g_f c_{j-1} + g_i z \tag{5}$$

where c_j is the new memory cell, and c_{j-1} is the old value of the memory cell. The forget gate g_f controls how much of old information should be forgotten, and the input gate g_i controls how much of the new information should be saved. The final output of the cell is not z, but the memory cell c_j passed through a *tanh* function and controlled by the output gate g_o. g_o controls how much information of memory cell should be output, as shown in Equation (6).

$$h_j = g_o(\tanh(c_j)) \tag{6}$$

where h_j is the output of LSTM at position j. The memory cell and gate mechanism can effectively alleviate the problem of vanishing gradient and explosion gradient of RNN. Hence, the LSTM can extract the long-distance dependency of sequences. Compared with an ordinary LSTM, the bidirectional LSTM [28] can extract bidirectional information of sequences, which is more effective than a one-directional LSTM. For the convenience of description, we denote the bidirectional LSTM as BiLSTM in this paper. The output of each position in the BiLSTM is the concatenation of the output of forwarding LSTM and the output of backward LSTM, as shown in Equations (7)–(9).

$$\overrightarrow{h_t} = \overrightarrow{LSTM}(e_t, \overrightarrow{h_{t-1}}) \tag{7}$$

$$\overleftarrow{h_t} = \overleftarrow{LSTM}(e_t, \overleftarrow{h_{t-1}}) \tag{8}$$

$$H_t = (\overrightarrow{h_t} : \overleftarrow{h_t}) \tag{9}$$

where $\overrightarrow{LSTM}$ denotes the forwarding LSTM model, $\overleftarrow{LSTM}$ denotes the backward LSTM model. $\overrightarrow{h_t}$, $\overrightarrow{h_{t-1}}$ denote the output of forward LSTM model at position t, t − 1 and $\overleftarrow{h_t}$, $\overleftarrow{h_{t-1}}$ denote the output of backward LSTM at position t, t − 1. e_t denotes the input of sequence at position t. H_t denotes the output of BiLSTM at position t which is the concatenation of $\overrightarrow{h_t}$ *and* $\overleftarrow{h_t}$. $\overrightarrow{h_t} \in R^d$, $\overleftarrow{h_t} \in R^d$, $H_t \in R^{2d}$. Equations (7) and (8) are recursive definitions of the output of forwarding and backward LSTM at position t. They show that the output of forward and backward LSTM is dependent on the current input and the output of the previous position. Equation (9) shows that the output of BiLSTM at each position is the concatenation of forward LSTM and backward LSTM. e_t is the embedding of the word at position t. The word embedding [29] is the continuous real-valued vectors, which can be pre-trained with a large corpus. The word embedding in this paper was pre-trained on Wikipedia corpus using fasttext model [30].

As shown in Figure 2, the output of BiLSTM encoder is the concatenation of the last state of forwarding and backward LSTM, which contains bidirectional information of the whole sequence.

In this paper, we use the BiLSTM to encode the first, middle and last part of the review into the three vectors s_1, s_2, s_3 with the same dimensions ($s_1, s_2, s_3 \in R^{d_m}$). Since that s_1, s_2, s_3 can only represent a part of the review, we use BiLSTM to encode the whole review into a vector s_c to catch the information of the whole review. Though s_1, s_2, s_3, s_c come from the same architecture, each encoder is independent with others and takes a different sequence as the input, hence, the outputs of them are totally different.

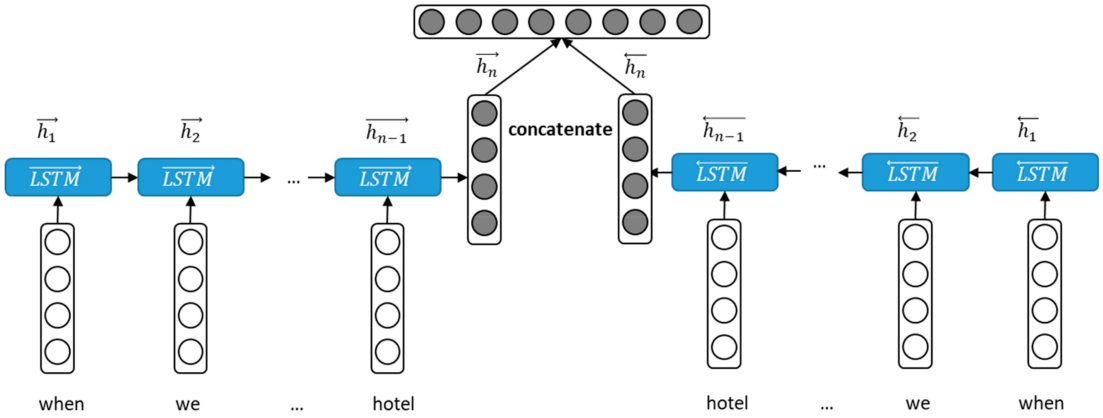


Figure 2. Bidirectional long-short term memory network (LSTM) encoder.

3.2. Self-Attention Mechanism

s_1, s_2, s_3, which are encoded by BiLSTM contain the information of the first, middle and last part of the review respectively. Since that $[s_1, s_2, s_3]$ can be regarded as a sequence with a length of three, a sequence model is a better way to integrate them than a weighted average. As shown in Figure 3, we use the self-attention mechanism to encode the sequence composed of s_1, s_2, s_3.

Self-attention [31] is a special kind of attention mechanism, which can effectively extract the dependencies of different positions like common sequence models such as RNN and CNN. Compared with RNN and CNN, it has fewer parameters and lower computational complexity. The output of the self-attention mechanism is the weighted average of different positions of the input sequence, and the weights are obtained by a function of the input sequence. We denote the weights and the function of *Attention* and *Adp*. In our model, the input sequence is a matrix composed of s_1, s_2, s_3. We denote the input sequence as S, $S = [s_1 : s_2 : s_3], S \in R^{3 \times d_m}$.

We use a multilayer perceptron (MLP) as the function *Adp* and use softmax to normalize the *Attention* because MLP can fit any continuous function and adjust parameters adaptively through backward propagation. The *Adp* function and *Attention* are defined as follows:

$$Adp = \tanh(W \cdot S^T + b) \tag{10}$$

$$Attention = softmax(Adp(S)) \tag{11}$$

$$Attention_i = \frac{\exp^{Adp(S_i)}}{\sum_{i=1}^{3} \exp^{Adp(S_i)}} \tag{12}$$

$Attention \in R^{3 \times 3}$, $W \in R^{3 \times d_m}$, $b \in R^{3 \times 3}$. The output of the self-attention mechanism is the weighted average of S, while the weights matrix is *Attention*. We denote the output as $Z, Z \in R^{3 \times d_m}$. Z can be represented as $[z_1 : z_2 : z_3]$ and z_i is obtained by the weighted average of S.

In fact, the output of the self-attention mechanism is still a sequence, and each element of the sequence can be viewed as a document representation. But compared with the input sequence, the output sequence has no information about the order of sequence. To keep the positional information

of the sequence, we add the positional encoding into $[z_1 : z_2 : z_3]$. In this paper, we use the sine and cosine function to encode the position, which is proposed by Vaswani et al. [31]:

$$PE_{pos,2i} = \sin(pos/10000^{\frac{2i}{d}}) \tag{13}$$

$$PE_{pos,2i+1} = \cos(pos/10000^{\frac{2i}{d}}) \tag{14}$$

PE is the positional encoding of sequence at position pos. $PE_{pos,2i}$ and $PE_{pos,2i+1}$ is the value of vector PE at position $2i$ and $2i + 1$. According to Vaswani et al. [31], compared with other encoding methods, this method can extract the relative positional information without adding any parameter to the model.

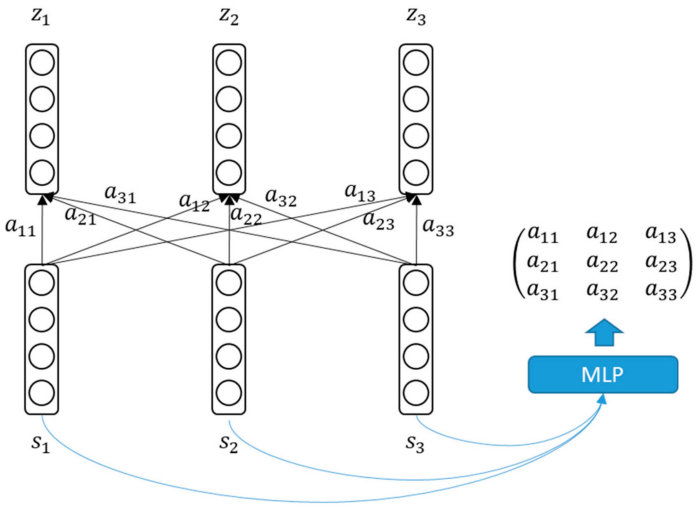


Figure 3. Self-attention mechanism.

3.3. Attention Mechanism

The output of the self-attention mechanism can be regarded as a sequence, and each element of the sequence includes the information of the whole review. While s_c encoded by BiLSTM using the whole review also contains the information of the whole review. Since Z and s_c are encoded by different models, we can integrate them to obtain a better document representation. However, the dimension of Z and s_c is different, $Z \in R^{3\times d_m}, s_c \in R^{d_m}$. Z can be represented as $[z_1 : z_2 : z_3]$, we can view Z as the concatenation of z_1, z_2, z_3. Hence, we are actually integrating four representations: z_1, z_2, z_3, s_c. We take $[z_1 : z_2 : z_3]$ as a sequence and use attention mechanism to encode the sequence. The reason to use the attention mechanism is that we cannot add s_c to the sequence $[z_1 : z_2 : z_3]$, because of the difference between s_c and z_i. But we can take s_c as the *query*, and Z as the key-value pair, which is natural in Attention mechanism (Figure 4).

The idea behind Attention mechanism is to compare each element of a sequence with a *query* vector. While the higher the similarity is, the larger weight the element can get. In our model, the *query* vector is s_c, and the sequence is $[z_1 : z_2 : z_3]^T$. The weight of z_i is denoted as a_i, and the matrix $[a_1 : a_2 : a_3]$ which is concatenated by a_1, a_2, a_3 is denoted as $Attention$. a_i is obtained by a similarity function of s_i and z_i. We use multilayer perceptron (MLP) to compute the similarity of s_i and z_i. And the softmax is applied to normalize the similarity. The Sim function and $Attention$ are defined as follows:

$$Sim = (tanh(s_c W_q + W_z Z^{\mathrm{T}} + b)) \tag{15}$$

$$Attention = softmax(Sim(s_c, Z)) \tag{16}$$

$$a_i = \frac{\exp^{Sim(s_c, z_i)}}{\sum_{i=1}^{3} \exp^{Sim(s_c, z_i)}} \tag{17}$$

$Attention \in R^{1\times 3}$, $Z \in R^{3\times d_m}$, $s_c \in R^{1\times d_m}$, $W_q \in R^{d_m\times 3}$, $W_z \in R^{1\times d_m}$, $b \in R^{1\times 3}$. We take *Attention* as the weights and make a weighted combination of Z to get the output O, as shown in Equation (18). O is the integration of s_1, s_2, s_3, s_c, which is the final representation of the review, $O \in R^{1\times d_m}$.

$$O = Attention \cdot Z \tag{18}$$

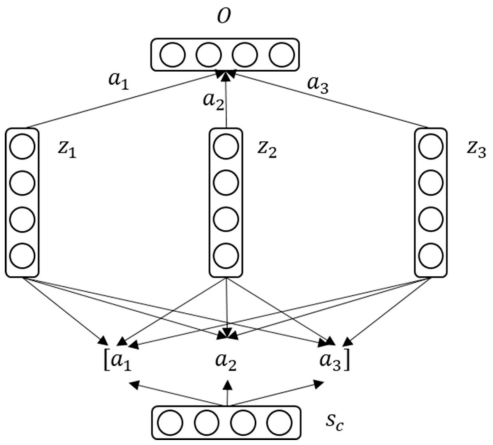


Figure 4. Attention mechanism.

3.4. Classifier

The classifier is a shallow fully-connected neural network based on the final document representation. Note that we can use this classifier to make a classification based on other representations such as s_1, z_1, s_c, but we only use O for classification because it combines the information of all other representations.

The fully-connected neural network is used to map multi-dimensional vectors to a 2-dimensional vector y, $y = [y_0, y_1]$. y_0, y_1 are scores of the review on two categories predicted by the model. The softmax is to normalize y_0 and y_1, the result of normalization which is denoted as p can be viewed as the probability distribution of model on two categories, $p \in R^2$. y and p are defined as follows:

$$\mathrm{y} = tanh\left(WO^T + b\right) \tag{19}$$

$$p = \frac{\exp(y_i)}{\sum_{i=1}^{2} \exp(y_i)} \tag{20}$$

4. Results

We evaluated our model in three experiments (in-domain, mix-domain, and cross-domain) based on three domain datasets (*Hotel*, *Restaurant*, *Doctor*). Compared with the baseline model of Li et al. [4] and Li et al. [6], the results of in-domain and mix-domain experiment showed that our model gets a better result than the compared methods.

4.1. Datasets and Evaluation Metrics

We evaluated the proposed model using the standard dataset proposed by Li et al. [4], which is the largest dataset of deceptive reviews to our best knowledge. The dataset contains three domains (Hotel,

Restaurant, and Doctor). The Table 2 shows the distribution of the data. There are three types of data in each domain: "Turker", "Expert" and "Customer". The review of type "Turker" and "Expert" belongs to deceptive reviews, while the reviews of type "Customer" are truthful reviews written by customers with high credibility. The review of type "Turker" are collected by Li et al. [4] and Ott et al. [2] through the Amazon online crowdsourcing market. The reviews of type "Expert" are written by experts with domain knowledge. However, the reviews of "Experts" are much fewer than the "Turker" and the "User", hence, we don't use them in the experiment.

Table 2. Statistics of the three-domain dataset.

Domain	Turker	Expert	User
Hotel	800	280	800
Restaurant	200	0	200
Doctor	356	0	200

We compared the proposed model with the baseline method [4] and the state-of-the-art method [6]. Li et al. [4] and Li et al. [6] evaluate their model in three kinds of experiments: in-domain experiments, cross-domain experiments, and mix-domain experiments. To make a comparison with them, we also tested our model in these three experiments. In order to make the results of experiments more reliable, we used five-fold cross-validation. The data was split into five equal folds, and four folds were taken as training data, the remaining fold is for testing. Li et al. [4] and Li et al. [6] used the F1 score, precision, recall, and accuracy to evaluate the performance of the model. To make a comparison with them, we also used these four metrics to evaluate our model. As shown in Tables 3–5, we compared our model (RSBE) with Li et al. [4]'s model (SAGE), and the Li et al. [6]'s model (SWNN) in three experiments. The SAGE model proposed by Li et al. [4] used n-grams features and the SAGE model [4], and the SWNN model proposed by Li et al. [6] is a hierarchical model based on convolution neural networks and hard attention mechanism.

4.2. In-Domain Experiments

Table 3 shows the results of in-domain experiments. In the hotel domain as well as doctor domain, our proposed model (RSBE) performed significantly better than SAGE and SWNN. In the restaurant domain experiment, the method of SWNN got the best result, but RSBE gained the highest recall and performed much better than SAGE. Although SWNN performed best in the restaurant domain, its performances on another two domains were much worse than the restaurant domain. The performance of RBME was stable on three domains, which is about 85% in every metric, although the sample size of restaurant dataset and doctor dataset was much smaller than the hotel dataset. However, the performance of SWNN and SAGE in doctor domain was much worse than their performances on the restaurant domain and hotel domain.

Table 3. In-domain results.

Domain	Method	Accuracy	Precision	Recall	F1
Hotel	SAGE	81.8%	81.2%	84.0%	82.6%
	SWNN	-	84.1%	83.3%	83.7%
	RSBE	**85.7%**	**85.5%**	**86.1%**	**85.7%**
Restaurant	SAGE	81.7%	84.2%	81.6%	82.8%
	SWNN	-	**87.0%**	88.2%	**87.6%**
	RSBE	**85.5%**	84.1%	**88.5%**	85.8%
Doctor	SAGE	74.5%	77.2%	70.1%	73.5%
	SWNN	-	85.0%	81.0%	82.9%
	RSBE	**84.7%**	83.6%	**86.5%**	**85.0%**

As mentioned in this paper, the first and last sentence of the review is more important than middle context. Therefore, there will be more information from detecting deceptive reviews given from the first and last sentence than middle context. Based on the point, RSBE extracts more information of detecting deceptive reviews than SWNN, which can enhance the sensitivity of detecting spam reviews. This view is proved by the experimental results in Table 3. The recall of RSBE was the best in three domains of all. In general, a classifier with high sensitivity gets a good performance in recall but might reduce the precision. That is the reason that F1 measurement was adopted to determine the classifier good or not.

The average F1 score of RBME (85.5%) was significantly higher than SAGE (79.6%) and SWNN (84.7%). In general, our method performs better than SWNN and SAGE in the in-domain experiment.

4.3. Mix-Domain Experiments

Table 4 shows the results of the mix-domain experiment. In this experiment, we gathered all domain data into a mix-domain dataset and verify our method with SWNN and several neural networks models. The results of Basic LSTM, Hier-LSTM, and Basic CNN were from Li et al. [6]'s paper. We have not compared our model with SAGE because there was no mix-domain result in Li et al. [4]'s paper.

Table 4. Mix-domain results.

Method	Accuracy	Precision	Recall	F1
SWNN	80.1%	80.0%	87.3%	**83.4%**
Basic LSTM	55%	59%	72%	72%
Hier-LSTM	62%	61%	**95%**	74%
Basic CNN	71%	69%	88%	78%
RSBE	**83.4%**	**82.5%**	82.1%	82.3%

All of the methods in Table 4 are learning a document representation using neural networks models. The basic LSTM method uses LSTM to extract document representation, and Hier-LSTM uses LSTM to extract sentence representations and combine them into a document representation. Basic CNN uses convolutional neural networks to learn document representation. SWNN is the modification of the Basic CNN model.

Table 4 shows the result that the RSBE and SWNN model performs significantly better than other neural networks models. Though Hier-LSTM gained a very high recall value, its accuracy and precision were very low, which means that the model fails to fit the data. The RSBE model gained the highest value in accuracy and precision, which are important metrics for classification, while SWNN gained the best results in recall and F1 score. In general, our method performs comparably with SWNN and better than other neural networks in the mix-domain experiment.

4.4. Cross-Domain Experiments

The cross-domain experiment is designed to test the robustness of the model. In the experiment, we trained a model on a dataset and evaluated the model on other datasets. Since the sample size of hotel dataset was the largest (1600), compared with the hotel dataset (400), and the restaurant dataset (400), we trained the model on the hotel dataset and test it on restaurant and doctor dataset.

Table 5 shows the results of the cross-domain experiment. In the test experiment on restaurant dataset, Li et al. [4]'s method gains the best results, while the performance of RSBE was better than SWNN on accuracy and precision metrics. In the doctor domain, Li et al. [6]'s method gained the best result because of the high recall, but it failed to get a good result in accuracy and precision which are important metrics to evaluate a model. In general, the performances of all three methods in the

cross-domain experiment is worse than in-domain and mix-domain experiments. All models trained on *Hotel* reviews performed better on *Doctor* reviews than on *Restaurant* reviews, which is reasonably due to the vocabulary of *Hotel* domain being more similar to the *Restaurant* domain.

Table 5. Cross-domain results.

Domain	Method	Accuracy	Precision	Recall	F1
Restaurant	SAGE	**7850.0%**	**81.3%**	74.2%	**77.8%**
	SWNN	69.0%	64.4%	**85.0%**	73.3%
	RSBE	71.6%	69.4%	77.2%	72.9%
Doctor	SAGE	5500.0%	57.3%	72.5%	6170.0%
	SWNN	**61.0%**	57.3%	**86.0%**	**68.8%**
	RSBE	60.5%	**60.0%**	65.7%	62.3%

4.5. Hyper-Parameters Tuning

As shown in Table 6, we found that there are four kinds of hyper-parameters which are important to the results of the experiment. Dropout is a common method to avoid overfitting in neural networks models [32], hence, we applied dropout to the output of BiLSTM, self-attention mechanism, attention mechanism, and fully-connected layer. The recurrent dropout is a special kind of dropout used inside of the BiLSTM to avoid overfitting [33]. Table 6 shows the best set of four kinds of hyper-parameters in the in-domain experiment. The best hyper-parameters on threes domains are very similar, which means that one best hyper-parameters setting can be applied to three domains.

Table 6. Best hyper-parameters setting.

Hyper-Parameters	Hotel	Restaurant	Doctor
Dropout rate	0.6	0.6	0.4
Recurrent Dropout rate	0.2	0.2	0.2
Output Dimension of BiLSTM layer	192	192	192
Output Dimension of fully-connected layer	128	128	128

To test the model's robustness on different hyper-parameters and different domains, we compared the influence of different hyper-parameters and different domains on the model's performance. As shown in Table 7 and Figure 5, we chose three important hyper-parameters and use the F1 score to evaluate the model's performance. To make the results clearer, we computed the standard deviation of F1 scores. All three hyper-parameters had a small standard deviation (under 0.014), which indicates that the model is robust to the varying of these three hyper-parameters. We also noticed that the standard deviation of the second hyper-parameter (dimension of the fully-connected layer) was obviously smaller than the other ones, which indicates that the model is more robust to the dimension-of-fully-connected-layer. Comparing the standard deviation of the same hyper-parameter on different domains, we notice that the hotel domain has the smallest standard deviation, which is reasonable because the hotel dataset is much larger than the other dataset.

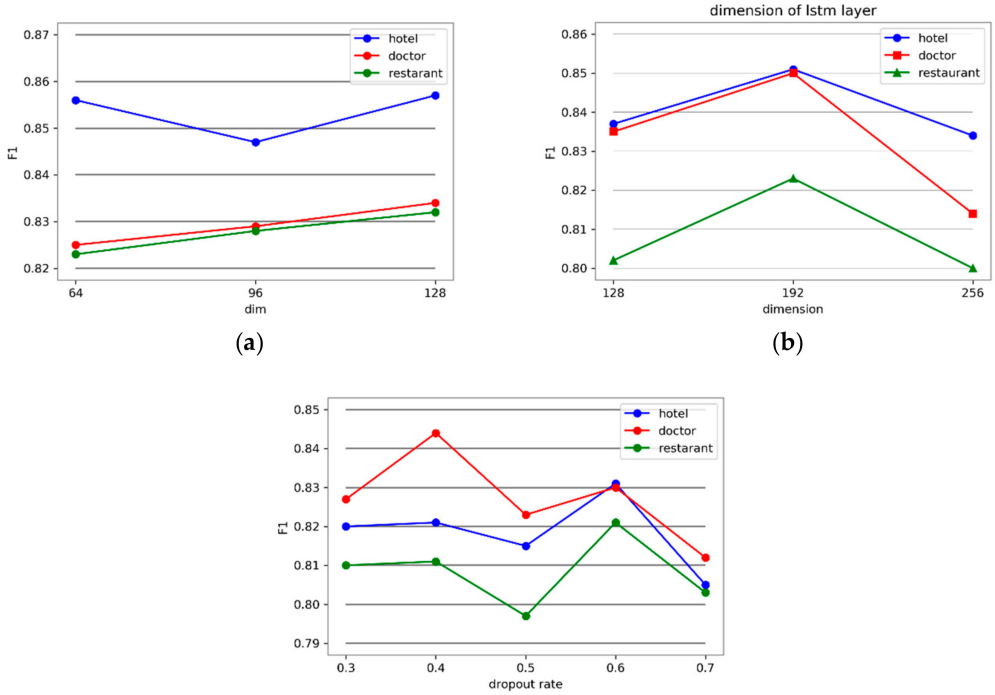


Figure 5. (**a**) the effect of the dimension of the fully-connected layer on the model's performance. (**b**) The effect of the dimension of the LSTM layer on the model's performance. (**c**) The effect of the dropout rate on the model's performance.

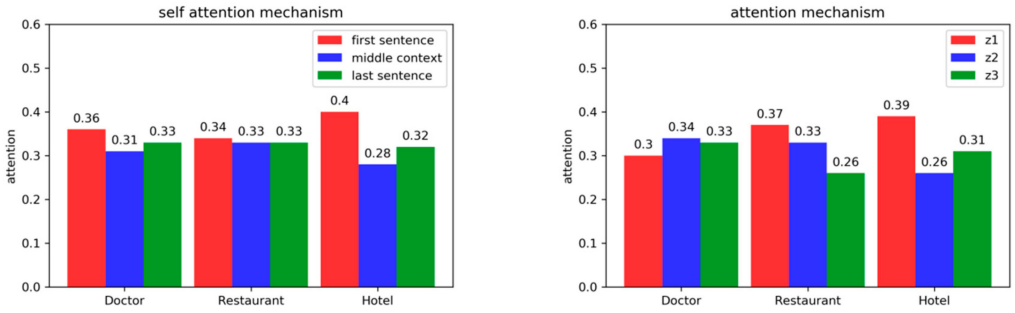
Table 7. The model's performance on different hyper-parameters and domains.

Hyper-Parameters		Hotel	Restaurant	Doctor	Standard Deviation
Dropout rate	0.3	0.820	0.810	0.827	**0.006**
	0.4	0.821	0.811	0.844	**0.013**
	0.5	0.815	0.797	0.823	**0.010**
	0.6	0.831	0.821	0.830	**0.004**
	0.7	0.805	0.803	0.812	**0.004**
	standard deviation	**0.008**	**0.008**	**0.010**	-
Dimension of fully-connected layer	64	0.856	0.823	0.825	**0.015**
	96	0.847	0.828	0.829	**0.008**
	128	0.857	0.832	0.834	**0.011**
	standard deviation	**0.004**	**0.003**	**0.003**	-
Dimension of LSTM layer	128	0.837	0.802	0.835	**0.016**
	192	0.851	0.823	0.850	**0.012**
	256	0.834	0.801	0.814	**0.013**
	standard deviation	**0.007**	**0.01**	**0.014**	-

To test the model's robustness on different domains, we computed the standard deviation of the model on different domains. The result shows that the standard deviations on different domains were all smaller than 0.016, which indicates that the model's performance is stable on different domains.

4.6. Visualization of Attention

As shown in Figure 6, we visualized the attention weights of the self-attention mechanism and the attention mechanism on three datasets. It is obvious that the weights of the first sentence, last sentence were higher than the middle context in the self-attention mechanism, which validates our assumption that the first sentence and the last sentence is more important than the middle context in a review. In particular, the rule is most significant in the *Hotel* dataset which is the largest dataset.



(**a**) (**b**)

Figure 6. (**a**) The attention weights of the self-attention mechanism described in Section 3.2 on three datasets. (**b**) The attention weights of attention mechanism described in Section 3.3 on three datasets.

However, the weights of attention mechanism failed to show significant rule, because the attention mechanism is stacked on the self-attention mechanism, the output of the self-attention mechanism contains no information about the order of sequence except for the absolute encoding we add to the output sequence. In other words, it is difficult to identify which of (z1, z2, z3) represents the first sentence or last sentence. We noticed, however, that the weights on hotel dataset in the attention mechanism were very similar to those in the self-attention mechanism, and the hotel dataset was much larger than other datasets, which may indicate that the sequence (z1, z2, z3) still keeps the order of (first sentence, mid context, last sentence).

5. Conclusions

This paper proposes an integrated model based on the structure of the review for deceptive review detection. Firstly, we split a review into three parts: the first sentence, middle context, and the last sentence. Then we used four independent bidirectional LSTM models to encode the three parts and the whole review. After that, to integrate the output of the four LSTM encoders, we stacked two layers of attention mechanism to get a final representation of the review, finally, the classification result was obtained through a fully-connected neural network based on the final representation.

We compared the RSBE model with two baseline methods [4,6]. In general, RSBE performs better than the compared methods in the in-domain and mix-domain experiment, which verifies the effectiveness of our method for deceptive review detection. The results of hyper-parameters tuning experiments indicate that our model is robust to different hyper-parameters and domains. The visualization of attention indicates that the structure of our model is reasonable since the weights of the first sentence and the last sentence is significantly higher than the middle context as we expected.

However, the model failed to perform well in the cross-domain experiment. In fact, the cross-domain experiment is a zero-shot learning task [34], because the test domain is unseen while training. The dictionary of different domains can be very different, therefore it is difficult for a model trained on a special domain to transfer to another domain. In the next study, we may try two approaches to the mentioned problems. One is to extract domain-independent features to train the model, such as the syntactic structure of sentences, high-frequency words; another is to make use of unlabeled data of the target domain. It is much easier to get the unlabeled data than the labeled data. The unlabeled data cannot give information about whether a review is deceptive or not, but it contains

rich domain-information which is useful for domain adaption. There are several domain-adaptive approaches which make use of unlabeled data [35,36]. It is worth applying these methods to RSBE, since the unlabeled data is easy to get, and we will verify it in the future.

Author Contributions: Conceptualization, M.-H.C.; Data curation, M.-S.C.; Formal analysis, Z.-Y.Z. and Y.-Q.L.; Funding acquisition, J.-L.L.; Methodology, Z.-Y.Z. and M.-H.C.; Software, M.-S.C. and J.-L.L.; Validation, Y.-Q.L.; Visualization, J.-J.L., J.-L.L.; Writing—original draft, Z.-Y.Z.; Writing—review & editing, J.-J.L., M.-H.C.

Funding: The Undergraduate Innovation and Entrepreneurship Training Program (2018265).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Jindal, N.; Liu, B. Opinion spam and analysis. In Proceedings of the 2008 International Conference on Web Search and Data Mining (WSDM '08), Palo Alto, CA, USA, 11–12 February 2008; pp. 219–230.
2. Ott, M.; Choi, Y.; Cardie, C.; Hancock, J.T. Finding Deceptive Opinion Spam by Any Stretch of the Imagination. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (HLT '11), Portland, OR, USA, 19–24 June 2011; pp. 309–319.
3. Ott, M.; Cardie, C.; Hancock, J.T. Estimating the prevalence of deception in online review communities. In Proceedings of the 21st international conference on World Wide Web (WWW '12), Lyon, France, 16–20 April 2012; pp. 201–210.
4. Li, J.; Ott, M.; Cardie, C.; Hovy, E.H. Towards a General Rule for Identifying Deceptive Opinion Spam. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, Baltimore, MD, USA, 23–25 June 2014; pp. 1566–1576.
5. Li, J.; Cardie, C.; Li, S. Topics pam: A topic-model based approach for spam detection. In Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, Sofia, Bulgaria, 4–9 August 2013; pp. 217–221.
6. Li, L.; Qin, B.; Ren, W.; Liu, T. Document representation and feature combination for deceptive spam review detection. *Neurocomputing* **2017**, *254*, 33–41. [CrossRef]
7. Ren, Y.; Ji, D. Neural networks for deceptive opinion spam detection: An empirical study. *Inf. Sci.* **2017**, *385–386*, 213–224. [CrossRef]
8. Shang, Y. Resilient consensus of switched multi-agent systems. *Syst. Control Lett.* **2018**, *122*, 12–18. [CrossRef]
9. Shang, Y. Resilient Multiscale Coordination Control against Adversarial Nodes. *Energies* **2018**, *11*, 1844. [CrossRef]
10. Shang, Y. Hybrid consensus for averager–copier–voter networks with non-rational agents. *Chaos Solit. Fract.* **2018**, *110*, 244–251. [CrossRef]
11. Feng, S.; Banerjee, R.; Choi, Y. Syntactic stylometry for deception detection. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, Baltimore, MD, USA, 22 June 2014; pp. 171–175.
12. Xu, Q.; Zhao, H. Using deep linguistic features for finding deceptive opinion spam. In Proceedings of the COLING, Mumbai, India, 8–15 December 2012; pp. 1341–1350.
13. Banerjee, S.; Chua, A.Y. A linguistic framework to distinguish between genuine and deceptive online reviews. In Proceedings of the International Conference on Internet Computing and Web Services, Baltimore, MD, USA, 22 June 2014.
14. Fusilier, D.H.; Montesygomez, M.; Rosso, P.; Cabrera, R.G. Detection of opinion spam with character n-grams. In Proceedings of the Conference on Intelligent Text Processing and Computational Linguistics, Cairo, Egypt, 14–20 April 2015; pp. 285–294.
15. Fusilier, D.H.; Montesygomez, M.; Rosso, P.; Cabrera, R.G. Detecting positive and negative deceptive opinions using PU-learning. *Inf. Process. Manag.* **2015**, *51*, 433–443. [CrossRef]
16. Hai, Z.; Zhao, P.; Cheng, P.; Yang, P.; Li, X.; Li, G. Deceptive Review Spam Detection via Exploiting Task Relatedness and Unlabeled Data. In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, Austin, TX, USA, 1–5 November 2016; pp. 1817–1826.
17. Feng, S.; Xing, L.; Gogar, A.; Choi, Y. Distributional Footprints of Deceptive Product Reviews. In Proceedings of the International Conference on Weblogs and Social Media, Dublin, Ireland, 4–7 June 2012.

18. Liu, Y.; Pang, B. Spam Detection based on Annotation Extension and Neural Networks. *Comp. Inf. Sci.* **2019**. Available online: https://pdfs.semanticscholar.org/a312/7f6c118a6e29be12679cefda14a363f9028e.pdf (accessed on 3 June 2019).
19. Sun, C.; Du, Q.; Tian, G. Exploiting Product Related Review Features for Fake Review Detection. *Math. Probl. Eng.* **2016**, 1–7. [CrossRef]
20. Breiman, L. Bagging predictors. *Mach. Learn.* **1996**, *24*, 123–140. [CrossRef]
21. Schapire, R.E. The strength of weak learnability. *Mach. Learn.* **1990**, *5*, 197–227. [CrossRef]
22. Breiman, L. Random forests. *Mach. Learn.* **2001**, *45*, 5–32. [CrossRef]
23. Kégl, B. The return of AdaBoost. MH: Multi-class Hamming trees. *arXiv* **2013**, arXiv:1312.6086.
24. Perrone, M.P.; Cooper, L.N. When Networks Disagree: Ensemble Methods for Hybrid Neural Networks. 1992. Available online: https://pdfs.semanticscholar.org/5956/40253ffdfd12e04ac57bd78753f936a7cfad.pdf?_ga=2.149320566.1196925254.1559288762-513896128.1544690129 (accessed on 3 June 2019).
25. Hansen, L.K.; Salamon, P. Neural network ensembles. *IEEE Trans. Pattern Anal. Mach. Intell.* **1990**, *10*, 993–1001. [CrossRef]
26. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef] [PubMed]
27. Williams, R.J.; Zipser, D. A learning algorithm for continually running fully recurrent neural networks. *Neural Comput.* **1989**, *1*, 270–280. [CrossRef]
28. Graves, A.; Schmidhuber, J. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Netw.* **2005**, *18*, 602–610. [CrossRef] [PubMed]
29. Bengio, Y.; Ducharme, R.; Vincent, P.; Janvin, C. A neural probabilistic language model. *J. Mach. Learn. Res.* **2003**, *3*, 1137–1155.
30. Bojanowski, P.; Grave, E.; Joulin, A.; Mikolov, T. Enriching Word Vectors with Subword Information. *Trans. Assoc. Comput. Linguist.* **2017**, *5*, 135–146. [CrossRef]
31. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Polosukhin, I. Attention is All You Need. In Proceedings of the 2017 Neural Information Processing Systems (NIPS), Long Beach, CA, USA, 4–9 December 2017; pp. 5998–6008.
32. Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* **2014**, *15*, 1929–1958.
33. Gal, Y.; Ghahramani, Z. A theoretically grounded application of dropout in recurrent neural networks. In Proceedings of the 30th Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain, 4–9 December 2016; pp. 1019–1027.
34. Palatucci, M.; Pomerleau, D.A.; Hinton, G.E.; Mitchell, T.M. Zero-shot Learning with Semantic Output Codes. In Proceedings of the 22nd International Conference on Neural Information Processing Systems (NIPS'09), Vancouver, BC, Canada, 7–10 December 2009; pp. 1410–1418.
35. Ganin, Y.; Lempitsky, V. Unsupervised domain adaptation by backpropagation. In Proceedings of the International Conference on Machine Learning, Lille, France, 6–11 July 2015; pp. 1180–1189.
36. Peng, M.; Zhang, Q.; Jiang, Y.; Huang, X. Cross-Domain Sentiment Classification with Target Domain Specific Information. In Proceedings of the Meeting of the Association for Computational Linguistics, Melbourne, Australia, 15–20 July 2018; pp. 2505–2513.