Semi-Supervised Fake Reviews Detection based on AspamGAN

Chen Jing-Yu[1], Wang Ya-Jun[2]

[1]Department of Electronics and Information Engineering, Liaoning University of Technology, China
[2]Professor, Department of Electronic and Information Engineering, Liaoning University of Technology, China

E-mail: [1]1966916925@qq.com

Abstract

With the popularization of social software and e-business in recent years, more and more consumers like to share their consumption experiences on social networks and refer to other consumers' reviews and opinions when making consumption decisions. Online reviews have become an essential part of browsing on websites such as shopping, and people's reliance on informative reviews have contributed to the rise of fake reviews. The traditional classification method is affected by the label dataset, which is not only time-consuming, laborious, and subjective, but also the extraction of artificial features also affects the classification accuracy. Due to the relative length of the online text, the possibility of the classifier losing important information increases, this weakens the model's detection capability. To solve this aforementioned problem, a semi-supervised Generative Adversarial Network (AspamGAN) fake reviews detection method incorporating an attention mechanism is proposed. Using labeled and unlabeled data to correctly learn input distributions, the features required for classification are automatically discovered using deep neural networks, providing better prediction accuracy for online reviews. The approach includes attention mechanisms in the classifier to obtain an adequate semantic representation and relies on a limited dataset of labeled data to detect false reviews, and is applied on the TripAdvisor dataset. Experimental results show that the proposed algorithm outperforms state-of-the-art semi-supervised fake review detection techniques when the label dataset is limited.

Keywords: Attention mechanism, classifier, fake review, Generative Adversarial Network, semi-supervised

DOI: https://doi.org/10.36548/jaicn.2022.1.002
Received: 10.02.2022, received in revised form: 11.03.2022, accepted: 22.03.2022, published: 30.03.2022

1. Introduction

Fake reviews are common in e-business, social platform, and travel websites [1]. Research shows that most potential consumers are influenced by relevant reviews [2], and more positive reviews increase consumers' desire to buy. The online world is full of watermen manipulating reviews, influencing normal users by disguising themselves as ordinary consumers, and posting fake reviews, thus promoting store sales and interests. And for most consumers, it is difficult to distinguish the authenticity of the reviews with the naked eye. Fake reviews affect the reference value of regular reviews and cause a severe drift in the review orientation of the entire product. Not only does it cause a loss of money to the user, but it also brings a terrible consumer experience. With the accumulation of false reviews, the reputation of the platform's moral credibility in the industry is destroyed, and consumers' satisfaction with the platform is significantly reduced. They may even abandon this platform, which seriously damages the interests of legitimate business merchants. Therefore, the research of false review detection is an inevitable trend of web development.

The purpose of fake review detection is to distinguish the authenticity of a review, i.e., given a review, one needs to distinguish whether it is a genuine review or a fake review. To address the lack of labeled datasets in fake reviews detection [3], researchers have built an online comment corpus, yet only a small number of reviews have been tagged, mainly because manual tagging is time-consuming and laborious, and accompanied by personal subjectivity [4]. It has been shown that a small amount of labeled data interacting with an unlabeled dataset can improve the accuracy of the classifier [5]. Semi-supervised learning methods [6] distinguish false reviews by using predefined sets of features to train classifiers. Deep generative models possess excellent predictive ability in semi-supervised learning, especially GAN. However, the GAN has been chiefly on image rather than the text data because, on one hand, text data are discrete values, and the gradients from the discriminator may not help to improve the generator and on the other hand, there is a sparse reward problem.

Because online reviews are relatively lengthy, the generation of existing generative adversarial network text data is also limited by the sentence length, such as MaskGAN [9]. However, the method was not designed for processing most online reviews. The authors of [10][10] proposed spamGAN, a semi-supervised fake review classification method, which learnt the correct input distribution through labeled datasets and used deep Artificial Neural Networks (ANN) to search the extracts needed for classification, using unlabeled data to

improve the generalization ability of the classifier. The method combined deep learning and reinforcement learning to achieve advanced results in fake review classification and generate samples similar to the training set. Also, the disadvantages of generative adversarial networks were overcome using their method. It is generally believed that the classification model and data determine the classification effect, and the implementation of high precision classification is closely related to the classifier. For text messages, the most important thing is how to capture contextual information. Despite the powerful advantages of spamGAN in text classification, it has a problem that the classifier used is too simple, increasing the possibility of losing important information and weakening the ability to detect false information. This problem limits the model's performance, especially when the online reviews are relatively lengthy and the performance of the model becomes weak.

AspamGAN, a structure that combines an attention mechanism with a classifier, is proposed to address this problem. Generally, for sequential data models (RNN, GRU, LSTM), it is easy to lose important information by operating on the extracted contextual information. The attention mechanism not only captures the focus in the sentence, suppresses other useless information, and focuses the limited attention on the focused information for more attention to details, but also allows for quick access to the most effective information and a better representation of the text, which in turn leads to improved model effects. In order to verify the feasibility of the proposed method, experiments are conducted on the dataset. Experimental results show that AspamGAN outperforms spamGAN in fake review detection when using limited labeled data.

2. Related Works

Fake review detection was introduced by Prof. Jindal [10] team at the University of Illinois in 2007. Most existing techniques for fake review detection are supervised methods based on predefined feature detection of logistic regression with the product. Jindal et al., [10] used logistic regression in combination with reviewer characteristics. Ahmed[12] proposed a supervised machine learning approach to identify fake reviews. Moreover, the comment extraction and behavioral features were also applied to improve classification results. Ott et al., [5] extracted n-gram features and trained plain Bayesian and support vector machines for classification based on these features. Feng et al., [12] used features such as phonetic labels, context-free parsing, and spatiotemporal features. Guo et al., [15] used a graph-based algorithm.

The neural network approach for fake review detection considers reviews as input, extracts some critical features from the text in the dataset, and uses a relevant algorithm to train the extracted features for classification. The authors of [17] predicted Stock Prices Using Pretrained Neural Networks. The authors of [18] discussed various common deep learning emotion recognition algorithms while leveraging the eXnet library to achieve improved accuracy. The authors of [19] incorporated Apache Spark's separate evaluation cutoff goals and AI initiatives that drive the underlying MultiLayer Perceptron (MLP), leveraging common cascade learning ideas. The authors of [16] and [21] proposed GRNN and DRI-RCNN models to classify fake reviews. The GRNN was used to learn the contextual information of text data and the DRI-RCNN augumented the RNN by learning the word embedding vectors of the labeled data of the reviews.

Few semi-supervised methods based on fake review detection currently exist. Li [8] classified text with a simple Bayes classifier by the Co-Training method using features of reviews, products, and reviewers. Hernández et al., [7] developed the labeled and unlabeled samples to improve the performance of classification. Rakibul [22] used the Expectation-maximization algorithm. Research for text classification mainly focused on addressing relative gradients and sparse rewards in sentence generation via GANs. SeqGAN[23] addressed these issues by treating sequence generation as reinforcement learning. MCTS was employed to solve the problem of sparse rewards, however, the complexity of MCTS increased. StepGAN [24] and MaskGAN used an actor-critic[25] method to set the reward mechanism, but the reliability has been limited. SpamGAN combines SeqGAN, StepGAN, and MaskGAN's advantages and treats sequence generation as a sequential decision problem. The generator continuously improves their "actions" by interacting with the "environment." In other words, for each text to be generated, the discriminator and classifier give a reward to the generator, whose sole purpose is to maximize the total reward obtained. To maximize the reward, the parameters of the generator are updated by the policy gradient. AspamGAN adds on top of this, the connection between texts, which is more suitable for long texts.

3. AspamGAN

To optimize the performance of classification by using the unlabeled data, an attention mechanism is added to the classifier. The proposed model mainly consists of the components mentioned below. Its general workflow is shown in Fig. 1.

For a given class label, the generator is responsible for learning to generate new sentences similar to D_L (i.e., fake sentences), where D_L is the labeled dataset, with a large number of unlabeled reviews D_U. It is used to improve the classification effect of the classifier. $D=D_L \cup D_U$ is used for subsequent training. The discriminator informs the generator whether the generated reviews are actual or not by learning the difference between genuine reviews and false sentences rewarding. The quality of new sentences is continuously improved in the competition between generator and discriminator. The class label '$\boldsymbol{c}$' of the dummy sentences generated by the generator is controlled, i.e., it is restricted by the class label. The classifier is trained using real reviews labeled in D_L and fake sentences are generated by the generator, which is adopted to improve the reasoning ability of the classifier. The classifier and the generator guide each other. The better the false sentences generated by the generator, the higher the classification accuracy of the classifier which brings more rewards to the generator.

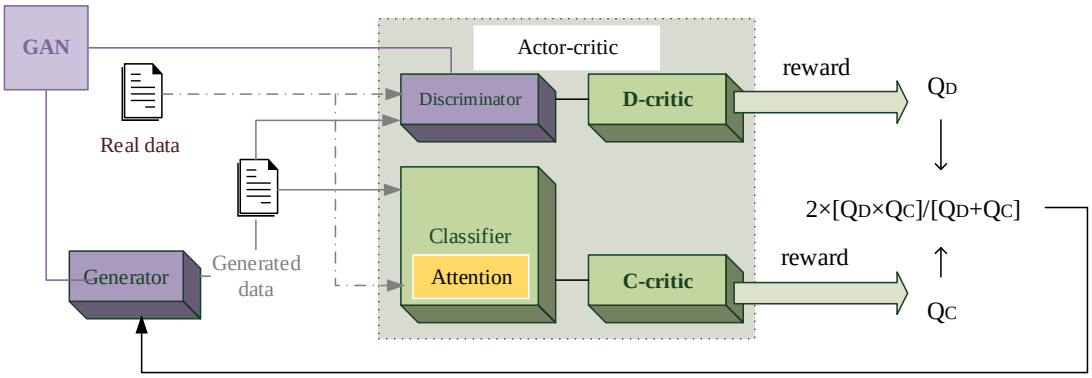

Figure 1. AspamGAN Architecture

3.1 Generator

$P_R(x_{1:T}, \boldsymbol{c})$ is the joint probability of sentences $x_{1:T}$ and classes label $\boldsymbol{c} \epsilon C$ from the actual dataset. The noise distribution P_Z and class label distribution P_C are defined for sampling where, the class label sampling is 0 (when $\boldsymbol{c}$ is 0, it means the sample is fake. When $\boldsymbol{c}$ is 1, it means that the sample is real). Random noise z is $-z$ and the class label is 1, and it is still z to ensure that the generator can more effectively perceive the difference in categories. The noise vector z and the class label $\boldsymbol{c}$ are given as input. After passing through the neural network with the parameter ∂_g, the generator will generate a distribution $G(x_{1:T}|z, c, \partial_g)$. The main purpose of the generator is to make the generated distribution as close as possible to the true distribution. Together, z and $\boldsymbol{c}$ form the context vector, which is

connected into complete sentences at time steps [23], ensuring the true class label of each retained false sentence. The generator generation process is shown in Fig. 2.

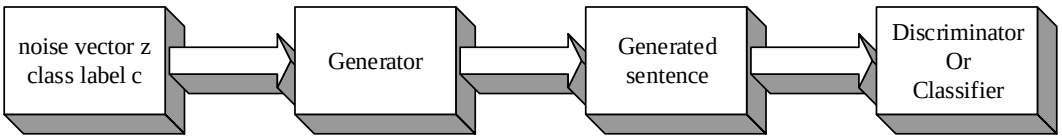

Figure 2. Generator generative process

When sampling from the distribution $G(x_{1:T}|z,c,\partial_g)$, the word tokens are generated by autoregression, and the distribution of the token sequence is decomposed into sequential conditional sequences by,

$$\mathrm{G}(x_{1:T}|z,c,\partial_g)=\prod_{t=1}^{T}G(x|x_{1:t-1},z,\boldsymbol{c},\partial_g) \quad (1)$$

During the pre-training period, the real sentences from the source are used, and the maximum likelihood function loss is minimized by,

$$L_{MLE}^{G}=-\sum_{t=1}^{T}log\ G(x_t|x_{1:t-1},z,\boldsymbol{c},\partial_g) \quad (2)$$

3.2 Discriminator

In the framework, the primary function of the discriminator with parameters ∂_d is to judge whether the sentence is real (sampled from P_R) or fake (generated by the generator), and output a probability score of $D(x_{1:T}|\partial_d)$. The higher the score $D(x_{1:T}|\partial_d)$, the greater the probability that the sentence is an actual sentence. Unlike the literature [24] that calculates the value in the end of the sentence, the discriminator generates a score $Q_D(x_{1:t-1},x_t)$ at each time step, and then generates the overall score by averaging.

$$D(x_{1:T}|\partial_{\mathrm{d}})=\frac{1}{T}\sum_{\mathrm{t=1}}^{\mathrm{T}}Q_{\mathrm{D}}\,(x_{1:t-1},x_t) \quad (3)$$

$Q_D(x_{1:t-1},x_t)$ is the score produced by the time step t, and the score is entirely based on the previous sentence. The discriminator can provide this value directly. From the perspective of discriminator D, it can distinguish between real samples and fake samples as much as possible. $E_{x_{1:T}\sim P_R}[\log(x_{1:T}|\partial_d)]$ means to put real data into the discriminant model and output is $D(x_{1:T}|\partial_d)$. The calculated value and the value of the entire formula should be as large as possible. $E_{x_{1:T}\sim G}\log\big(1-D(x_{1:T}|\partial_d)\big)$ means to put fake data into the discriminant

model $D(x_{1:T}|\partial_d)$. The output calculated value is as small as possible, and the entire formula value is as large as possible. The integration is to make the objective function take the maximum value. Therefore, the minimum loss $L^{(D)}$ is:

$$L^{(D)} = \underset{x_{1:T}\sim P_R}{E} - [\log D(x_{1:T}|\partial_d)] + \underset{x_{1:T}\sim G}{E} - \left[\log\left(1 - D(x_{1:T}|\partial_d)\right)\right] \quad (4)$$

The architecture also includes a critical discriminator network **Error! Reference source not found.**, which is used to judge the score of the discriminator's behavior. The discriminator will also modify the probability of behavior based on the score of the critical network $V_D(x_{1:t-1})$. The policy gradient update that is used for the generator in the confrontation training is:

$$V_D(x_{1:t-1}) = \underset{x_t}{E}[Q_D(x_{1:t-1}, x_t)] \quad (5)$$

The loss function is $Q_D(x_{1:t-1}, x_t)$ and the $V_D(x_{1:t-1})$ is the minimum mean square error between:

$$L^{(D_{critic})} = \underset{x_{1:T}}{E} \sum_{t=1}^{T} \|Q_D(x_{1:t-1}, x_t) - V_D(x_{1:t-1})\|^2 \quad (6)$$

The discriminator is a unidirectional RNN with a dense layer, which outputs the score of an actual sentence at every time step $Q_D(x_{1:t-1}, x_t)$. The discriminant network is an additional fully connected output layer for output at each time step $V_D(x_{1:t-1})$.

3.3 Classifier Based on Attention Mechanism

3.3.1 Principle of attention mechanism

The Attention Mechanism (AM) is used to automatically learn and calculate the contribution of the input data to the output data. The core of the AM is that the context of each target word is different. Luong et al., [25] used the global and local attention mechanisms to obtain the context vector. Guo et al., [27] proposed a multi-scale self-attention mechanism model through which multi-scale features in text can be obtained. In terms of text classification, the attention mechanism expresses the focus of attention on different words, and word representations are aggregated to form sentence vectors. The structured selection of input subsets reduces the data dimensionality, and the learning content is better. Such considerations are more reasonable. For a sentence, it can be regarded as a sequence composed of multiple words. The information before and after the sentence sequence is

learned through the neural network to obtain the information before and after the sentence words which is the semantic encoding of text sentences. Adding this layer to extract the characteristics of essential phrases can further extract the deeper information between the texts. Here, multi-head attention is used. The structure of this power mechanism is shown in Fig. 3.

Using the multi-head attention mechanism, the method model can use different sequence positions to obtain spatial representation information for sequence data processing. Multi-head attention to extract the meaning of multiple semantics first defines the number of hyperparameter heads. The dimension of the word vector must be divisible by the head, and therefore the number of heads is set to 8.

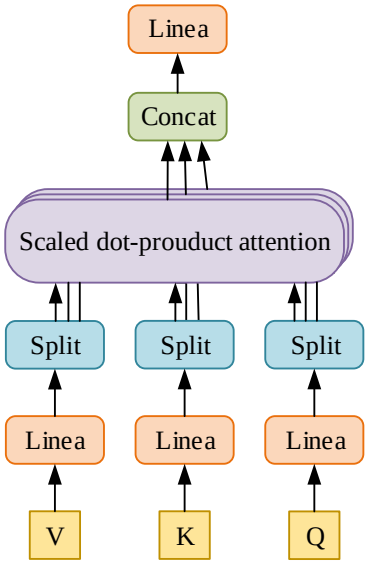

Figure 3. Schematic diagram of attention structure

The attention function pipes a query, which is a set of key-values into an output. Query, Key, and Value first undergo a linear transformation and then input to the zoom dot product attention. After calculating the dot product of query and all keys, it is divided by the root d_K (d_K is the dimension to prevent the gradient from disappearing). And a softmax function is used to obtain the weight threshold of AM. The calculation process is as follows:

$$Attention(Q,K,V) = softmax\left(\frac{QK^T}{\sqrt{d_K}}\right)V \tag{7}$$

Finally, multiple attentions are connected.

$$Multihead(Q,K,V) = Concat(head_1, \dots head_8)W^0$$

$$head_i = Attention\left(QW_i^Q, KW_i^K, VW_i^V\right) \tag{8}$$

where, W^0 is the weight matrix.

3.4 Classifier based on attention mechanism

When a comment sample $x_{1:T}$ is given as an input, the classifier with parameters ∂_C predicts whether the sentence is a false comment, and compares the real class label to adjust its own parameters for training and learning. The classifier assigns a prediction score $Q_C(x_{1:t-1}, x_t, c)$ at each time step, which is considered to be the probability of belonging to class c. At each time step $Q_C(x_{1:t-1}, x_t)$, both generate a score and average the score to develop the overall score.

$$C(x_{1:T}, c|\partial_C) = \frac{1}{T}\sum_{t=1}^{T} Q_C(x_{1:t-1}, x_t, c) \tag{9}$$

The classifier loss consists of two parts: $L^{(C_R)}$ is the cross-entropy loss of real sentences; $L^{(C_G)}$ is the cross-entropy loss of fake sentences. Because the fake sentence is the result of the noise vector training, the cross-entropy loss must be minimized, and the Shannon entropy must also be minimized. In $L^{(C_G)}$, β is the balance coefficient. On one hand, the minus sign maximizes the objective function, and on the other hand, after taking the negative Shannon entropy, $H\left(C(c|x_{1:T}, \partial_c)\right)$ is for minimum entropy regularization[28], allowing the classifier to classify fake sentences more accurately.

$$L^C = L^{(C_R)}L^{(C_R)} + L^{(C_G)}$$

$$L^{(C_R)} = \underset{(x_{1:T}, c)\sim P_R(x,c)}{E}[-\log C(c|x_{1:T}, \partial_C)] \tag{10}$$

$$L^{(C_G)} = \underset{c\sim P_R, x_{1:T}\sim G}{E}\left[-\log C(c|x_{1:T}, \partial_c) - \beta H\left(C(c|x_{1:T}, \partial_c)\right)\right]$$

Like the discriminator, it also includes a critical classifier network to evaluate the classifier's score $Q_C(x_{1:t-1}, x_t, c)$.

$$V_C(x_{1:t-1}, c) = \underset{x_t}{E}[Q_C(x_{1:t-1}, x_t, c)] \tag{11}$$

The classifier implementation is similar to the discriminator. The classifier also uses a unidirectional recurrent neural network and combined with a multi-headed attention

mechanism. It is spliced with review sentences as a classification prediction input vector, and the output is based on the probability distribution of predicted class labels. The critical network of the same classifier uses a dense layer to estimate $V_D(x_{1:t-1}, c)$ for each period of time. Then $L^{(C_{critic})}$ is minimized.

$$L^{(C_{critic})} = \underset{x_{1:T}}{E} \sum_{t=1}^{T} \| Q_C(x_{1:t-1}, x_t, c) - V_C(x_{1:t-1}, c) \|^2 \tag{12}$$

3.5 Application of reinforcement learning in AspamGAN

Reinforcement emphasizes how to act based on the environment to obtain the maximum expected reward. The schematic diagram of the application of reinforcement learning in AspamGAN is shown in Figure 4.

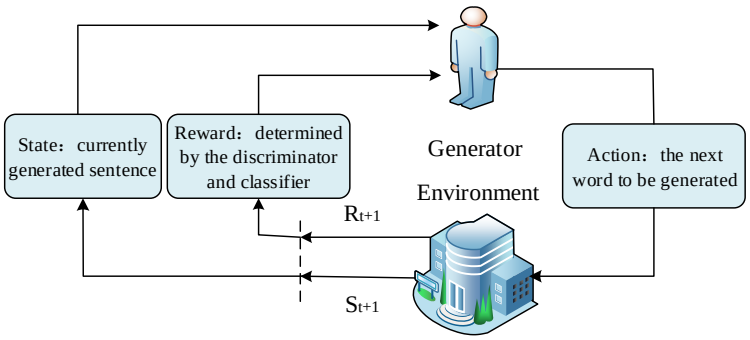

Figure 4. Application of reinforcement learning in AspamGAN

In adversarial training, the sequence generation problem is a decision problem [23]. In the sequence decision framework, the generator interacts with the environment and is considered an agent in reinforcement learning. And the policy gradient method is used to maximize the expected reward. The reward is the feedback acquired via the discriminator and classifier. The current state is the text generated $s_t = x_{1:t-1}$ by the current generator. Action a_t is the next word, which is selected based on the random strategy $G(x_t | x_{1:t-1}, z, c, \partial_g)$. For the generated reviews $x_{1:T}$ with class label c, the reward received by the generator is decided by the discriminator and classifier. Scores $D(x_{1:T}|\partial_d)$ (formula 3) and $C(x_{1:T}, c|\partial_c)$ (formula 11) are used to generate sentence reward $R(x_{1:T})$.

$$R(x_{1:T}) = 2 \cdot \frac{D(x_{1:T}|\partial_d) \cdot C(x_{1:T}, c|\partial_c)}{D(x_{1:T}|\partial_d) + C(x_{1:T}, c|\partial_c)} \tag{13}$$

Only when the generator reaches the last state can the generator get the reward, and the reward in the sentence generation process is 0. Therefore, the generator seeks to maximize the cumulative reward.

$$L^{(G)} = \underset{x_{1:T} \sim G}{E}[R(x_{1:T})] \tag{14}$$

To maximize $L^{(G)}$, the generator continuously optimizes the mapping relationship between actions and states through learning from experience and finally finds the optimal strategy [29]. Specifically, the Critic is employed to forecast the value, and the Actor is used to output the strategy. The discriminator's intermediate score $Q_D(x_{1:t-1}, x_t)$ and the classifier's intermediate score $Q_C(x_{1:t-1}, x_t, c)$ are combined as given in Equation 15. The hybrid value is regarded as the estimated value of $Q(x_{1:t-1}, c)$, that is, the expected return of the sentence $x_{1:t}$. In order to reduce the variance, the advantage function $Q(x_{1:t}, c) - V(x_{1:t-1}, c)$ is used instead of $Q(x_{1:t}, c)$, where $Q(x_{1:t}, c)$ is shown in Equation 15.

$$Q(x_{1:t}, c) = 2 \cdot \frac{Q_D(x_{1:t-1}, x_t) \cdot Q_C(x_{1:t-1}, x_t, c)}{Q_D(x_{1:t-1}, x_t) + Q_C(x_{1:t-1}, x_t, c)} \tag{15}$$

$$V(x_{1:t-1}, c) = 2 \cdot \frac{V_D(x_{1:t-1}) \cdot V_C(x_{1:t-1}, c)}{V_D(x_{1:t-1}) + V_C(x_{1:t-1}, c)}$$

In Equation 16, α is a linear decreasing factor. During adversarial training, the gradient equation of Equation 16 is used to perform gradient ascent to update the generator.

$$\nabla_{\partial_g} L^{(G)} = \underset{x_{1:T}}{E} \sum_{t}^{T} \alpha[Q(x_{1:t}, c) - V(x_{1:t-1}, c)] \times \nabla_g \log G\left(x_t \middle| x_{1:t-1}, z, c, \partial_g\right) \tag{16}$$

3.6 Experimental Algorithm

3.6.1. Pre-training

Before starting the confrontation training, the Generator, Discriminator, and Classifier must be pre-trained. Its first task is to prevent the collapse of the model [30]. Equation 2 Maximum Likelihood Estimation (MLE) [31] is used to pre-train the generator. Once the generator is pre-trained, D_L、D_U and the fake sentences sampled in the generator $G\left(x_{1:T} \middle| z, c, \partial_g\right)$ are used to pre-train the minimum loss value $L^{(D)}$ of Equation 4. The classifier only uses real sentences from D_L. The real sentences and their label training are

used to minimize cross-entropy loss L^{C_R}. Critical networks use formula 6 and formula 12 to train their losses $L^{(D_{critic})}$、$L^{(C_{critic})}$.

3.6.2 Adversarial training

After the pre-training, there will be confrontation training. The specific steps are as follows: Firstly, the parameters of the discriminator and classifier are kept unchanged, and then the generator is trained. By sampling the class label c from $P(c)$, the generator creates a batch of fake sentences based on the class label. The discriminator and classifier are used to calculate $Q(x_{1:t}, c)$ and $V(x_{1:t-1}, c)$ at each time step. The generator is updated through the strategy gradient in formula (16). In order to improve the stability and robustness of training in the process of adversarial training, this work repeatedly uses the real sample data *D* to update using the maximum likelihood estimation formula. Next, fake sentences from the labeled dataset D_L and unlabeled dataset D_U and the generator are used to train the discriminator. The minimum loss $L^{(D)}$ is used to update the discriminator, i.e., formula (4), and $D_{Dcritic}$ (formula 6) is used to train the evaluator of the discriminant network. Similarly, formula (10) and formula (12) are used to train the classifier and its classification evaluator.

4. Analysis of Experimental Results

4.1 Dataset

This topic uses real reviews from the twenty most popular hotels in Chicago on TripAdvisor and fake reviews from 20 hotels on Amazon Mechanical Turk. In the end, 20 hotels are selected for this project to collect 20 real reviews and 20 fake reviews, a total of 800 reviews. At the same time, the duplicate labeled reviews are removed, and a total of 1596 tagged reviews are used. In addition, 32,297 unmarked reviews are used from the TripAdvisor website.

4.2 Data preprocessing

Data preprocessing refers to cleaning the original data. Most of the reality are missing, inconsistent and dirty data that cannot be directly processed and mined. Since the comments in the dataset used by the text are all English text, the preprocessing of the comment text in this paper includes the following operations:

1) Lowercase all English letters. Because Chinese and English texts are different. English texts are case-sensitive, such as "like" and "Like"; these two strings

are different, but they must be considered as the same word when counting words. So the upper- and lower-case letters in the text must be converted to lower case letters.

2) Strip numbers and their special characters. Due to the fact that many users in online comments use some special symbols, such as punctuation marks, emoticons or digital symbols, in order to express their feelings and emotions, or to express accurately, these symbols are not suitable for text classification, and however deleting them can reduce the dimension of features. When performing semantic feature extraction on comment text, these special symbols cannot be recognized for feature extraction or vectorization. Therefore, non-English characters and Arabic numerals need to be removed during text processing.

3) Remove stop words. Stop words have little substantive meaning compared to other words in natural language processing. For example, "the", "a", etc.; these words appear in a large number in almost every comment, and their contribution to text classification and semantic expression is low, and so stop words need to be removed.

4) Participle. The word segmentation methods for Chinese and English texts are different. Selecting according to the provided text is one of the word segmentation methods for English texts. If words, punctuation marks or other characters in the text are separated by spaces, such as "a little of both .", then the split() method can be used directly.

4.3 Evaluation Indicators

The main evaluation metrics commonly used in NLP to measure image or text classification are accuracy, precision, recall and F1 value. To verify the feasibility of the proposed model, two evaluation metrics, *Accuracy* and *F1* value are applied.

The *Accuracy* formula is as follows:

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{17}$$

where, TP (True Positive) is the positive reviews for the correct prediction, and FP (False Positive) is the positive reviews for the incorrect prediction, FN (False Negative) is

the negative reviews that indicate that the prediction is wrong, and *TN* (True Negative) is the negative reviews indicate that the prediction is correct.

The *F1* value formula is as follows:

$$F1 = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall} \tag{18}$$

$$Precision = \frac{TP}{TP+FP} \tag{19}$$

$$Recall = \frac{TP}{TP + FN} \tag{20}$$

Among them, *Precision* represents the proportion of useful parts in the entire detection that ensues to the whole detection results as useful, and *recall* represents the proportion of useful parts in the entire detection that ensues to the useful parts of the entire dataset.

4.4 Experimental environment

The experimental environment is Python=3.6 TensorFlow=1.14. The practical Python libraries include tensorflow, texar, importlib, numpy, logging, time, random libraries, etc.

4.5 Parameter settings

This article uses a total of 1,600 labeled reviews, including 800 real reviews and 800 fake reviews. All review texts are uniformly converted to English lowercase and assigned to the word level. At the same time, the size of the dictionary is set to 10,000, and these words are all from the corpus. The maximum sequence length is set to T=128, which is close to the median of the review text of the entire dataset.

At the same time, the dictionary also includes <eos>, <bos>, and <pad>, among which <eos> and <bos> are added to the beginning and end of each review. To ensure the consistency of sentence length, reviews with text length less than T are filled with <pad>, and longer reviews are truncated and replace words that are not in the dictionary.

In the AspamGAN model structure, the generator consists of 2 GRU layers, each layer of GRU has 1024 neural units, and the last layer of Dense layer outputs the probability of 10,000 words, which is used to generate sentences. The discriminator contains two GRU layers, each with 512 neural units, and is connected to a fully connected layer to output

category probabilities. The discriminator evaluator is an additional dense layer for estimate calculation. The structure of the classifier is roughly similar to that of the discriminator. The classifier also consists of two GRU layers, each with 128 neurons, and a multi-head attention structure in the encoder contains 8 attention heads and 512 units. At the same time, a fully connected layer is connected to output category labels, and the discriminator evaluator has an additional dense layer for estimate calculation.

In the model, the three module structures are trained using the ADAM optimizer, and the dimensions of the neural network and word embedding are both set to 50. In the maximum likelihood estimation training of the generator, the learning rate is set to 0.001 and the weight attenuation to 5e-3. The gradient clipping is set to the maximum global norm of 5. In the discriminator and classifier, the learning rate and weight attenuation settings are the same, set to 0.0001 and 1e-4, respectively, and the weight attenuation of the two evaluators is set to 1e-3. When performing reinforcement learning training, the optimizer's learning rate is set to 0.00005, and the weight attenuation to 1e-7.

4.6 Comparison experiment of SpamGAN and AspamGAN

To verify the effect of the analysis on unlabeled data under labeled data conditions, the labeled samples are increased from 10% to 30%, 50%, 70%, 90% and 100%, and the proportion of unlabeled samples is set to 0%, 50%, 70% and 100%. To verify the feasibility of this method, AspamGAN and SpamGAN are compared, and for the labeled dataset, 20% is used as test data and 80% as training data. Since the structure and experimental data of this experiment are consistent with SpamGAN, the comparison experiment of SpamGAN is not repeated, and hence the experimental results of SpamGAN are used as the main comparison object for experimental analysis. Among them, the experimental methods proposed by spamGAN are compared with some common supervised and semi-supervised algorithms, which are 1) DRI-RCNN; 2) RCNN; 3) Co-Training, and 4) PU-Learning. The experimental results are shown in Tables 1 and 2.

Table 1. Experimental Results of Accuracy on Different Label Datasets

Method	10%labeled	30%	50%	70%	90%	100%

spamGAN-0%	68.12	77.50	79.99	82.50	85.22	85.31
spanGAN-50%	67.79	79.72	79.37	85.50	84.06	85.58
spanGAN-70%	69.55	76.25	82.13	84.06	82.50	85.62
spanGAN-100%	68.07	77.50	83.10	81.56	84.68	84.53
Base classifier	74.69	76.56	81.87	83.43	83.12	83.43
AspamGAN-0%	76.24	81.56	83.75	84.37	86.24	85.31
AspanGAN-50%	71.25	80.62	86.25	86.25	85.30	86.56
AspanGAN-70%	72.50	82.50	82.18	84.68	85.31	86.56
AspanGAN-100%	71.87	81.25	86.56	84.68	87.18	84.50
Base classifier	76.25	80.31	83.75	84.06	86.56	84.68

Table 2. Experimental Results of F1 Values of Different Label Datasets

Method	10%labeled	30%	50%	70%	90%	100%
spamGAN-0%	70.71	79.91	80.41	82.96	85.45	84.18
spanGAN-50%	70.36	79.56	79.41	84.76	84.58	86.26
spanGAN-70%	70.16	77.62	82.34	83.83	82.05	84.63
spanGAN-100%	71.46	77.88	83.71	83.43	85.88	84.97
Base classifier	73.07	77.89	81.79	83.35	83.61	83.68
AspamGAN-0%	76.50	82.83	83.47	85.98	86.01	85.69
AspanGAN-50%	71.19	80.67	86.69	86.70	86.40	87.25
AspanGAN-70%	72.42	82.70	84.02	85.49	85.65	86.76
AspanGAN-100%	75.35	81.02	88.18	84.84	87.36	86.42
Base classifier	76.50	80.49	83.88	85.52	87.53	85.44

4.7 Analysis of results

In the experiment, AspamGAN is compared with the SpamGAN. The experiment uses Accuracy and F1 values as evaluation indicators to assess the performance of the model. Table 1 shows the classification accuracy rate on the test set. When only 10% of the labeled data set is used, the accuracy rates of AspamGAN-0, AspamGAN-50, AspamGAN-70, and AspamGAN-100 are 76.24, 71.25, 72.50, 71.87, and 76.25 respectively. Table 2 is the F1 score of this experiment. Obviously, when the proportion of unlabeled data increases, the F1 score decreases significantly, especially for AspamGAN-100. There may be two reasons: 1) because the unlabeled data far exceeds the amount of labeled data, and the unlabeled data may contain reviews and class tags not generated by the generator; 2) the Amazon labeled data is different from the Chicago unlabeled data. Comparing Tables 1 and 2, it is not difficult to see that the experimental results of AspamGAN are better than SpamGAN in many experiments. The F1 value and accuracy rate are both higher than the results of SpamGAN. The experimental results show that AspamGAN achieves a better recognition effect under a small quantity of labeled samples and dramatically reduces the number of labels.

5. Conclusion

This paper proposes an AspamGAN that integrates the attention mechanism in the classifier, a method that uses limited label data to detect fake reviews. The model can label unlabeled datasets and generate reviews similar to the datasets. After adding the attention mechanism to the classifier, it effectively solves the problem of poor performance due to the too simple classifier. The experimental results show that when the labeled data is limited, the performance of AspamGAN is better than that of SpamGAN, and the overall accuracy and F1 values are improved. It is believed that the insufficiency of the generated data may affect the classification accuracy. In the future work, improving the generator's data and providing better data for the classifier will be focused on.

References

[1] Jindal N, Liu B, Lim E P. Finding unusual review patterns using unexpected rules[C]//Proceedings of the 19th ACM international conference on information and knowledge management. 2010: 1549-1552.

[2] Crowd Learning Hub. https://learn.g2crowd.com/customer-reviews-statistics. 2018.

[3] Rayana S, Akoglu L. Collective opinion spam detection: Bridging review networks and metadata[C]//Proceedings of the 21th acm sigkdd international conference on knowledge discovery and data mining. 2015: 985-994.

[4] Li Y, Pan Q, Wang S, et al. A generative model for category text generation[J]. Information Sciences, 2018, 450: 301-315.

[5] Ott M, Choi Y, Cardie C, et al. Finding deceptive opinion spam by any stretch of the imagination[J]. Proceedings of ACL, 2011, pp. 309-319

[6] Li F H, Huang M, Yang Y, et al. Learning to identify review spam[C]//Twenty-second international joint conference on artificial intelligence. 2011.

[7] Hernández-Fusilier D, Guzmán-Cabrera R, Montes-y-Gómez M, et al. Using PU-learning to detect deceptive opinion spam[C]//Proceedings of the 4th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis for Computational Linguistics: Human Language Technologies, Atlanta, Georgia, USA. 2013: 38-45.

[8] Li H, Liu B, Mukherjee A, et al. Spotting fake reviews using positive-unlabeled learning[J]. Computación y Sistemas, 2014, 18(3): 467-475.

[9] Fedus W, Goodfellow I, Dai A M. MaskGAN: Better text generation via filling in the_[J]. arXiv preprint arXiv:1801.07736, 2018.

[10] Stanton G, Irissappane A A. GANs for semi-supervised opinion spam detection[J]. Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence (IJCAI-19). 2020: 5204-5210.

[11] Jindal N, Liu B. Opinion spam and analysis[C]//Proceedings of the 2008 international conference on web search and data mining. 2008: 219-230.

[12] Ahmed M. Elmogy,Usman Tariq,Ammar Mohammed,Atef Ibrahim. Fake Reviews Detection using Supervised Machine Learning[J]. International Journal of Advanced Computer Science and Applications (IJACSA), 2021, 12(1): 601-606.

[13] Feng S, Banerjee R, Choi Y. Syntactic stylometry for deception detection[C]//Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers). 2012: 171-175.

[14] Li H, Chen Z, Mukherjee A, et al. Analyzing and detecting opinion spam on a large-scale dataset via temporal and spatial patterns[C]//ICWSM. 2015: 634-637.

[15] Guo Z, Tang L, Guo T, et al. Deep graph neural network-based spammer detection under the perspective of heterogeneous cyberspace[J]. Future generation computer systems, 2021, 117: 205-218.

[16] Akoglu, L., Chandy, R., & Faloutsos, C. (2021). Opinion Fraud Detection in Online Reviews by Network Effects. Proceedings of the International AAAI Conference on Web and Social Media, 7(1), 2-11.

[17] Anand, C. "Comparison of Stock Price Prediction Models using Pre-trained Neural Networks." Journal of Ubiquitous Computing and Communication Technologies (UCCT) 3, no. 02 (2021): 122-134.

[18] Kottursamy, Kottilingam. "A review on finding efficient approach to detect customer emotion analysis using deep learning analysis." Journal of Trends in Computer Science and Smart Technology 3, no. 2 (2021): 95-113.

[19] Brahmane, Anilkumar V., and B. Chaitanya Krishna. "A Novel Approach for Gigantic Data Examination Utilizing the Apache Spark and Significant Learning." In International Conference on Inventive Computation Technologies, pp. 874-882. Springer, Cham, 2019.

[20] Ren Y, Ji D. Neural networks for deceptive opinion spam detection: An empirical study[J]. Information Sciences, 2017, 385: 213-224.

[21] Lai S, Xu L, Liu K, et al. Recurrent convolutional neural networks for text classification[C]//Twenty-ninth AAAI conference on artificial intelligence. 2015.

[22] Hassan R, Islam M R. Detection of fake online reviews using semi-supervised and supervised learning[C]//2019 International conference on electrical, computer and communication engineering (ECCE). IEEE, 2019: 1-5.

[23] Yu L, Zhang W, Wang J, et al. Seqgan: Sequence generative adversarial nets with policy gradient[C]//Thirty-first AAAI conference on artificial intelligence. 2017.

[24] Tuan Y L, Lee H Y. Improving conditional sequence generative adversarial networks by stepwise evaluation[J]. IEEE/ACM Transactions on Audio, Speech, and Language Processing, 2019, 27(4): 788-798.

[25] Millán-Arias C C, Fernandes B J T, Cruz F, et al. A robust approach for continuous interactive actor-critic algorithms[J]. IEEE Access, 2021, 9: 104242-104260.

[26] Luong M T, Pham H, Manning C D. Effective approaches to attention-based neural machine translation[J]. arXiv preprint arXiv:1508.04025, 2015,1412-1421.

[27] Guo Q, Qiu X, Liu P, Xue X Y. Multi-scale self-attention for text classification[C]. Proceedings of the AAAI Conference on Artificial Intelligence. 2020, 34(05): 7847-7854.

[28] Hu Z, Yang Z, Liang X, et al. Toward controlled generation of text[C]//International Conference on Machine Learning. PMLR, 2017: 1587-1596.

[29] Sutton R S, McAllester D A, Singh S P, et al. Policy gradient methods for reinforcement learning with function approximation[C]//NIPs. 1999, 99: 1057-1063.

[30] Guo J, Lu S, Cai H, et al. Long text generation via adversarial training with leaked information[C]//Proceedings of the AAAI Conference on Artificial Intelligence. 2018, 32(1).

[31] Grover A, Dhar M, Ermon S. Flow-gan: Combining maximum likelihood and adversarial learning in generative models[C]//Thirty-second AAAI conference on artificial intelligence. 2018.

Author's biography

Chen Jing-Yu, Bachelor of Electronic Information Engineering. Now studying Master of Information and Information Processing Theory and Application at Liaoning University of Technology majoring in Electronics and Communication Engineering Department and currently engaged in research on fake reviews detection.

Wang Ya-Jun, Professor, Ph.D, is engaged in research related to power electronic technology and application, modern power supply technology, signal and information processing