Information Sciences 526 (2020) 274–288

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Generating behavior features for cold-start spam review detection with adversarial learning

Xiaoya Tang, Tieyun Qian*, Zhenni You

School of Computer Science, Wuhan University, Hubei, China

ARTICLE INFO

Article history:
Received 30 January 2019
Revised 31 January 2020
Accepted 21 March 2020
Available online 1 April 2020

Keywords:
Spam review detection
Cold-start problem
Generative adversarial network

ABSTRACT

Due to the wide applications, spam detection has long been a hot research topic in both academia and industry. Existing studies show that behavior features are effective in distinguishing the spam and legitimate reviews. However, it usually takes a long time to collect such features and thus is hard to apply them to cold-start spam review detection tasks. Recent advances leveraged the neural network to encode the various types of textual, behavior, and attribute information for this task. However, the inherent problem, i.e., lack of effective behavior features for new users who post just one review, is still unsolved.

In this paper, we exploit the generative adversarial network (GAN) for addressing this problem. The key idea is to *generate synthetic behavior features (SBFs) for new users from their easily accessible features (EAFs)*. Specifically, we first select six well recognized real behavior features (RBFs) existing for regular users. We then train a GAN framework including a generator to generate SBFs from their EAFs including text, rating, and attribute features, and a discriminator to discriminate RBFs and SBFs. We design a new implementation of generator and discriminator for effective training. The trained GAN is finally applied to new users for generating synthetic behavior features. We conduct extensive experiments on two Yelp datasets. Experimental results demonstrate that our proposed framework significantly outperforms the state-of-the-art methods.

1. Introduction

Online reviews are playing more and more important roles for both product providers and customers. On one hand, the decent reviews and high ratings increase the visibility of the products and push them ahead of unobtrusive ones. On the other hand, positive or negative reviews have great impacts on the consumers' purchase intentions [2,28]. Both give strong incentives for fraudsters to play the system. As a result, an economy of paid reviews has flourished. For example, Washington Post reported on April 23, 2018 that half of the 32,435 reviews for the top 10 Bluetooth headphones on Amazon were problematic based on calculations using the ReviewMeta tool.[1]. Similarly, Dianping processed nearly 6,000,000 illegal reviews in the first half of 2017.[2]

Fake or spam reviews destroy the trusts of customers, and the honest sellers and manufacturers. Consequently, automatic detection of spam reviews becomes an urgent task and arouses great research interests [13,15,31,38,39]. Previous studies

* Corresponding author.
E-mail addresses: xiaoyatang@whu.edu.cn (X. Tang), qty@whu.edu.cn (T. Qian), znyou@whu.edu.cn (Z. You).

[1] https://www.washingtonpost.com/business/economy/.

[2] http://finance.ifeng.com/a/20170424/15317547_0.shtml.

https://doi.org/10.1016/j.ins.2020.03.063
0020-0255/

in spam review detection primarily focused on exacting linguistic and behavioral features. However, there are two inherent drawbacks when using linguistic or behavioral features. On one hand, linguistic features are ineffective in detecting the real-life fake reviews [34,43], as the spammers tend to change their writing styles when posting spam reviews. On the other hand, it usually takes a long time and efforts to collect a large number of samples to make the observations on behavior features. When dealing with the cold-start problem, i.e., *a review is just posted by a new reviewer*, it is hard to construct effective behavioral features for the new reviewer.

The cold-start spam review detection problem is critical in preventing the damage of spams in their early stage. Two recent studies [43,47] proposed to encode various types of textual, behavior, and attributes information into the embeddings of reviews and/or reviewers for the cold-start problem. The embedding technique helps overcome data sparsity with powerful expressive abilities, and hence two embedding models [43,47] show much better performance than the traditional methods. Nevertheless, the inherent problem, i.e., *lack of effective behavior features for new users who post just one review*, remains unsolved.

We build on ideas from many previous studies of finding effective behavior features, but we move one step further in that *we generate the synthetic behavior features (SBFs) for new users who actually do not have such features*. To this end, we resort to the recent advance in generative adversarial network (GAN). More specifically, we first extract six real behavior features (RBFs) for regular users. These RBFs have been proven to be effective in spam review detection [8,20,34,38,42]. We then carefully select three types of easily accessible features (EAFs) including text, rating, and attribute features. The rationale is that *such features should exist for both regular users and new users*. Taking these EAFs as the input, we generate synthetic behavior features in the generator of the GAN, and the competition is performed between the synthetic and real behavior features in the discriminator of the GAN. The trained GAN is finally applied to new users to get synthetic behavior features which are actually not yet observed for these new users.

The traditional GAN involves two types of information like texts and images, and aims to transform one into the other, e.g., images to texts or vice versa. Unlike texts and images, we cannot directly generate SBFs from RBFs as the new users do not have RBFs. Our goal is to transform EAFs into SBFs, meanwhile, we hope the generated SBFs are close to RBFs. That is to say, the EAFs serve as the connection between SBFs and RBFs. Therefore, we propose a new implementation of generator and discriminator under the GAN framework. Firstly, our generator not only takes EAFs as an input but also is conditioned on the auxiliary RBFs. Secondly, our discriminator introduces an extra loss to ensure the matching between EAFs and RBFs.

For example, given a user with his/her id and registration date as follows.

User's id: b-QKRonggw94ftkrqZNvTg

User's registration time: March, 2011

We would like to identify if the following review is a spam.

posting time: 3/8/2012

Review text: Went here on Valentine's Day and was blown away by my meal! Everything was incredible and the only thing I was less than impressed with was my dessert which was an assortment of chocolate treats. Cannot wait to try the brunch here and without kids! I love this place!

Rating score: 4

Due to the lack of enough number of reviews, we cannot collect good behavior features for this user. A RBFs-based classifier will classify this review as a normal one. However, we find our proposed model can make a correct prediction as the GAN module generates effective SBFs based on the user's registration time and the content. The reason might be two-fold. The first is there are a number of spam users registering in the system in March, 2011, indicating that there might be a fake reviewer group [32]. The second is that the style of the content is similar to those of real spam reviews. Both these are captured by the GAN module when generating SBFs, and hence our model successfully recognizes this review as a spam review.

In summary, the main contributions of our work are as follows.

1. We employ the GAN architecture to generate synthetic behavior features by competing with real behavior features which have been proved to be effective in spam detection.
2. The synthetic behavior features are generated from several low-cost features which are easily accessible even for new users in cold-start scenario.
3. We design a novel implementation of generator and discriminator which is critical for achieving superior performance.

We conduct extensive evaluations on two real world Yelp datasets. Results demonstrate that our model significantly outperforms the state-of-the-art baseline methods.

2. Related work

We first review the related work on spam detection and generative adversarial networks (GAN) in this section, we then discuss the potential application of our generated behavior features in other domains.

2.1. Spam review detection

The problem of spam review detection has long been a hot research topic. Existing studies primarily focused on extracting various linguistic features [9,10,14,15,18,21,23,36,45]. However, Mukherjee et al. [34] found that the linguistic features were insufficient for finding spam reviews. Since the spammers may consistently write spam reviews [20], a good number of studies paid attention on finding behavioral features of reviewers [1,8,16,17,24,32,34,41,44]. Another direction was towards the integration of multiple information [12,13,20,38]. Overall, the traditional methods rely on the manually constructed features which require great efforts from domain experts or much time to collect.

Recent years witnessed the booming of deep learning technique in spam review detection. Ren and Zhang [39] found that CNN was more effective than RNN on encoding the spam reviews. Wang et al. [42] learned the embedding of the reviewer and item using a tensor decomposition approach.

The problem of cold-start spam review detection was first introduced in [43] and an embedding learning model was presented to jointly utilize the behavioral information of reviewers and the textual information. You et al. [47] further leveraged the abundant information from attributes and domain knowledge to alleviate data scarcity and enhance representation of entities. While we aim to solve the same problem as that in [43,47], we propose a totally different framework which exploits the adversarial network to generate the behavior features for new reviewers.

2.2. Generative adversarial networks (GAN)

Goodfellow el al. [11] proposed GAN which extends the idea of a generative machine by eliminating the Markov chains. GAN is inspired by two-player zero-sum game and includes a generator and a discriminator, both trained under the adversarial learning idea. The optimization process of GAN is a mini-max game process, and the optimization goal is to reach Nash equilibrium [37]. Salimans et al. [40] presented several techniques for improving the stability of training and the perceptual quality of GAN samples.

GAN has been applied to the research fields like vision tasks [40], speech and language processing [22,48]. However, non existing work on spam review detection adopted this technique. We aim to exploit GAN to generate the behavior features which are effective in spam detection but new users lack such information in the cold-start scenario.

2.3. Behavior computing

Behavior is a particularly important concept in various types of fields like science, economics, and politics [5], and thus behavior computing [4,5,46] has aroused great research interests in recent years. Behavior refers to manner of behaving or acting, and the action or reaction of any material under given circumstances in dictionaries. Human activities can be collected from physical instruments such as radar [6], accelerometer [7], and wearable sensors [26,27,49], and a good number of methods like probabilistic model and deep learning algorithms [25,27,35] have been developed for activity recognition.

While the aforementioned studies focus on physical activities, our research pays more attention to the social behaviors collected from online web sites. Despite the difference between the collection and analysis of physical activities and those of social behaviors, our method for generating behavior features can be applied to other domains whenever the real activity features are hard to collect.

3. Real behavior and easily accessible features

We present real behavior features and easily accessible features in this section.

3.1. Real behavior features (RBFs)

We first introduce six real behavior features which are well recognized to be effective in spam detection [8,20,34,38,42]. Note these features only exist for *regular reviewers*. Different from new ones, regular reviewers registered in the system long ago and ususally posted a number of reviews. The six RBFs are as follows.

(1) Activity Window (**AW**) [33,34] is the time difference between the first and the last review of a reviewer. Spam reviewers are more likely to post multiple spam reviews in one day or within a few days. Hence their activity window is usually shorter than that of normal users.
(2) Maximum Number of Reviews (**MNR**) [34,38] is the maximum number of reviews a reviewer posts in a day. Spam reviewers often do not spend too much time for a bit of benefit. Thus most of them will write many spam reviews in one day.
(3) Percentage of Positive Reviews (**PR**) [34,38] is the percentage of high rating (4 or 5) reviews of one reviewers' all reviews. Most spammers aim to promote a product, and PR can reflect their purpose.
(4) Review Count (**RC**) [33,34] is the total number of reviews posted by reviewers. AW reflects reviewers' short-term activity trends, while RC reflects long-term ones.
(5) Reviewer Deviation (**RD**) [8,20,34,38,42] is the average rating deviation of a reviewer from rating of other reviewers on the same product. Empirically, the rating of spam reviewers will deviate from that of normal ones.

Table 1
Terms used in this paper.

Term	Interpretation
RBF	real behavior feature
SBF	synthetic behavior feature
EAF	easily accessible feature
EAF+	positive EAF matching RBF or SBF
EAF-	negative EAF mismatching RBF or SBF
TF	text feature (one type of EAF)
RF	rating feature (one type of EAF)
AF	attribute feature (one type of EAF)
$\mathcal{L}_D$	loss function for discriminator
$\mathcal{L}_G$	loss function for generator
Θ_D	parameter space in discriminator
Θ_G	parameter space in generator

(6) Maximum Content Similarity (**MCS**) [8,38] is the maximum cosine similarity between the reviewer's two comments. In order to save time, spam reviewers tend to post similar or duplicate reviews.

All the above real behavior features can be extracted from regular users, and will be used to contest against the synthetic behavior features in GAN.

3.2. Easily accessible features (EAFs)

We then present three types of features which will be used to generate syntactic behavior features. These features are selected according to one criterion, i.e., *easily accessible for both regular and new reviewers*. We notice that even for a new reviewer, he/she has the following information.

Firstly, a reviewer must post a review. Hence we use the review as one source of EAF to extract *text features (TF)*. We adopt the convolutional neural network (CNN) to pre-train a 100-dimension embedding for the review as those in [43,47].

Secondly, a reviewer must rate the product at the same time as he/she posts the review. We use the deviation of rating score of this review from the average ratings on the same product to extract *rating features (RF)*. We discretize the value into a 100-dimension vector. If the value falls into an interval, the corresponding dimension is set to 1 and other dimensions are set to 0. For example, the rating ranges from 0 to 5. If a rating deviation is 1.23, then the 25th entry of 100-dimension vector will be 1 and all other entries will be 0.

Thirdly, a reviewer should have a timestamp when he/she registered in the system and the review must have a posting timestamp. The register and posting timestamp is the attribute of a reviewer and review, respectively. We use the deviation between these two timestamps to exact *attribute features (AF)*. The time deviation value is discretized into a 100-dimension vector using the same method as that for rating deviation.

In summary, we extract three types of easily accessible features (EAFs), including text, rating, and attribute features. We will investigate their effects in generating synthetic behavior features later. For ease of reading, we present the above terms and those used in this paper later in Table 1.

4. Methodology

Our overall framework consists of two parts. One is the bfGAN model to generate the SBFs for the test data. The other is the normal classification model to predict whether a sample is a spam or not. Since the classification is a straight-forward procedure, we mainly focus on the bfGAN model and then briefly introduce the classification model.

4.1. Architecture of bfGAN model

In this section, we present the details of our behavior feature generating (bfGAN) model for addressing the cold-start spam review detection problem. The key idea is to learn a mapping from EAFs of regular users to SBFs and keep SBFs close to RBFs, then build new users' SBFs from their EAFs. Motivated by recent advances in deep learning, we design a novel GAN framework with new implementing strategies to achieve this goal.

We first present the architecture of our bfGAN model in Fig. 1. It consists of two basic components, i.e., the generator and the discriminator.

The left part in Fig. 1 is the generator which is used to generate SBFs from the input EAFs. We elaborately select three types of EAFs including text, rating, and attribute features. In Fig. 1, they are shown in yellow, blue, and pink, respectively. The right part is the discriminator which will make a discrimination between SBFs and RBFs using a classifier. We will present the details in the following sections.

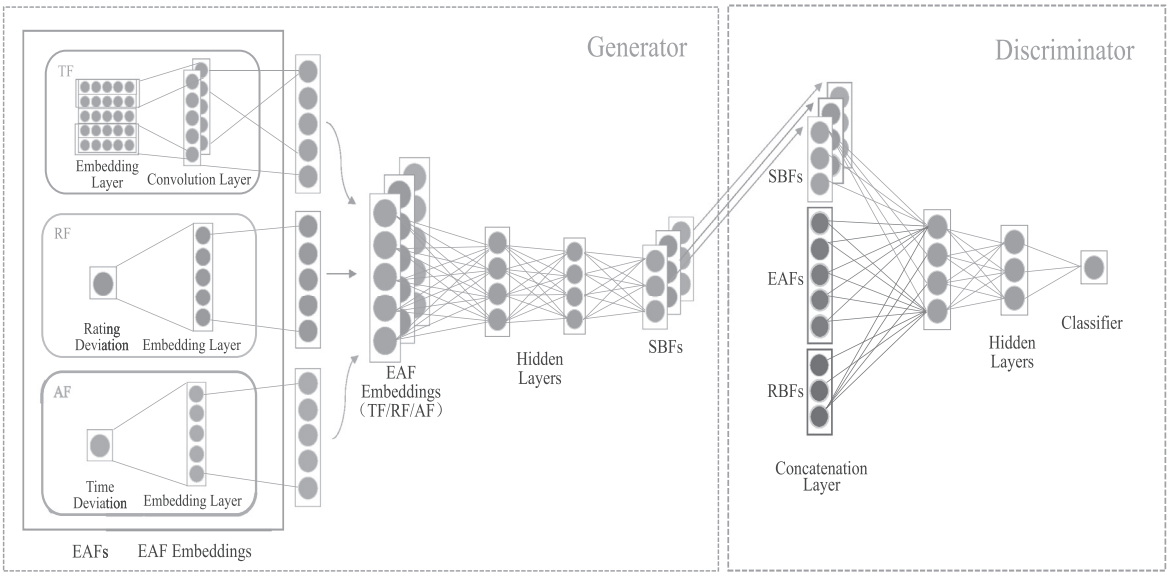


Fig. 1. Architecture of our bfGAN model.

4.2. Generator in bfGAN

In our bfGAN model, the generator is used to learn the mapping from the input EAFs to SBFs under the constraint of keeping SBFs close to RBFs. As shown in Fig. 1, the generator contains six layers in total. The first three layers are used to do normalization and get the embeddings of text, rating, and attribute features. Text embeddings are pre-trained through a CNN, and rating and attribute embeddings are obtained through the embedding layers. We then use two non-linear hidden layers to transform EAFs into SBFs.

The generator is designed to generate SBFs from the input EAFs, and hence the generated SBFs should match the EAFs. We call this task loss $\mathcal{L}_t$ and define it using the multi-nominal logistic loss (a.k.a. cross entropy loss).

$$\mathcal{L}_t = \min_{\Theta_G} J(D(EAF_+ \oplus SBF), 1), \tag{1}$$

where J is the cross entropy function, D is the discriminative function consisting of multiple hidden layers activated by *tanh*, and $\oplus$ denotes the concatenation of two embeddings. Moreover, as our goal is to use SBFs to simulate RBFs for new users, we hope SBFs are close to RBFs in training. To achieve this, we add a closeness loss $\mathcal{L}_c$ to encourage the generator to generate SBFs that have similar distribution as RBFs. $\mathcal{L}_c$ is also defined using the cross entropy loss.

$$\mathcal{L}_c = \min_{\Theta_G} J(RBF, SBF), \tag{2}$$

The overall loss for the generator G is the linear combination of the task loss $\mathcal{L}_t$ and the clossness loss $\mathcal{L}_c$, and is defined as:

$$\begin{aligned}\mathcal{L}_G = \min_{\Theta_G} \; & \lambda J(D(EAF_+ \oplus SBF), 1) \\ & + (1-\lambda) J(RBF, SBF),\end{aligned} \tag{3}$$

where the balancing parameter λ controls the weights of two losses. We will investigate its effects in our experiments.

4.3. Discriminator in bfGAN

As shown in Fig. 1, the discriminator in our bfGAN model consumes the SBFs from the generator, the EAFs and RBFs from the training data as input, and it outputs 1 if it judges SBFs are similar to RBFs and 0 otherwise from a classifier.

Our discriminator is designed to guide the training by distinguishing the synthetic and real behavior features. However, the discriminator can not observe the (RBF, SBF) pairs in the training data. Instead, its inputs are the RBFs that match EAFs and the SBFs generated by EAFs. In other words, the discriminator should judge the (EAF+, RBF) pairs from the realistic training data as real and the (EAF+, SBF) pairs from the generator as fake. Similarly to that in the generator, we simply define two loss functions $J(D(EAF_+ \oplus RBF), 1)$ and $J(D(EAF_+ \oplus SBF), 0)$ for this purpose.

Besides the error introduced by the generator, another source of error in the discriminator may come from the unrealistic behavior features. In order to separate two sources of error and simplify learning dynamics, we add a third type of input consisting of RBFs with mismatched EAFs, which the discriminator should learn to score as fake. We formally define a loss

function $J(D(RBF \oplus EAF_{-}), 0)$ to achieve this. Taking all these into consideration, we define the function loss $\mathcal{L}_D$ for the discriminator as follows.

$$\begin{aligned}\mathcal{L}_D = \min_{\Theta_D} \; & J(D(EAF_{+} \oplus RBF), 1) \\ & + \frac{1}{2} J(D(EAF_{+} \oplus SBF), 0) \\ & + \frac{1}{2} J(D(RBF \oplus EAF_{-}), 0),\end{aligned} \tag{4}$$

4.4. Learning procedure for bfGAN

To train our bfGAN model, we view the (RBF, EAF+) pairs extracted from regular reviewers as joint observation. We train the generator to generate SBFs, as well as optimize the discriminator to distinguish SBFs and RBFs. We summarize the learning procedure of our model in Algorithm 1.

Algorithm 1 The learning procedure of bfGAN model.

Require: $X_{train} = \{r_i, x_i, y_i, z_i\}_{i=1}^{m}$, where X_{train} denotes the training set of reviews from regular users and m is its size. i denotes the i-th review and its reviewer in X_{train}, from which we can extract six RBFs, denoted as r_i, and three types of EAFs denoted as x_i, y_i, and z_i, respectively.

Ensure: the trained bfGAN network

1: initialize the parameters of the model, including Θ_G and Θ_D for the generator G and the discriminator D.
2: **repeat**
3: **for** each batch in X_{train} **do**
4: forward EAF_{+} to G to generate SBFs;
5: update Θ_G by minimizing Eq. 3;
6: get EAF_{-} through random sampling;
7: forward SBFs, EAF_{+}, EAF_{-}, RBFs to D;
8: update Θ_D by minimizing Eq. 4;
9: **until** the model reaches equilibrium point or a predefined number of iterations;
10: **return** the trained bfGAN network

It works as follows. Line 1 initializes the parameters including Θ_G and Θ_D for generator and discriminator. Then for each batch in the training set, line 4 generates three types of SBFs in the generator using three types of EAFs in training data; line 5 updates the parameters by optimizing the loss function $\mathcal{L}_G$; line 6 gets the negative samples; lines 7–8 train the discriminator by optimizing its loss function $\mathcal{L}_D$. During the learning procedure, the SBFs and RBFs share the weights in the discriminator. The procedure repeats until the model reaches the equilibrium point or the predefined number of iterations.

The overall dataflow chart for our framework is presented in Fig. 2. The left part is for the bfGAN model and the right one is for the classification model.

As can be seen in the left part of Fig. 2, we are provided with a set of users' reviews, which are divided into the training set and the testing set. We can extract EAFs which include TFs, RFs, and AFs from both the training and testing set. However, RBFs only exist in training set. The train EAFs and train RBFs are used to learn the bfGAN model which can generate SBFs. More precisely speaking, we train the parameters Θ_G and Θ_D in bfGAN by minimizing the loss function $\mathcal{L}_G$ and $\mathcal{L}_D$ for the generator and the discriminator, respectively.

4.5. The overall flowchart and classification model

After the bfGAN model is ready, we apply the trained bfGAN network to the test EAFs in the test data to generate test SBFs for them. These SBFs will be used to represent each test sample during the classification procedure.

For a clear presentation, we further provide the overall flowchart for the entire framework in Fig. 3, where the middle left part denotes the bfGAN procedure, and the right and lower part denotes the classification procedure.

Since we have presented the detail for the bfGAN model in the previous subsections, here we introduce the classification procedure. Following the prior studies [43,47], we adopt the linear SVM as our classification model.

Specifically, we first train a linear SVM classifier [3] using the training samples which are represented by their RBFs. Then the trained SVM classifier is applied to the test data. Note that the test samples are represented with their SBFs generated by the bfGAN model. Finally, each test sample will be labeled as a spam or not a spam by the SVM classifier.

[3] https://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVC.html#sklearn.svm.LinearSVC.

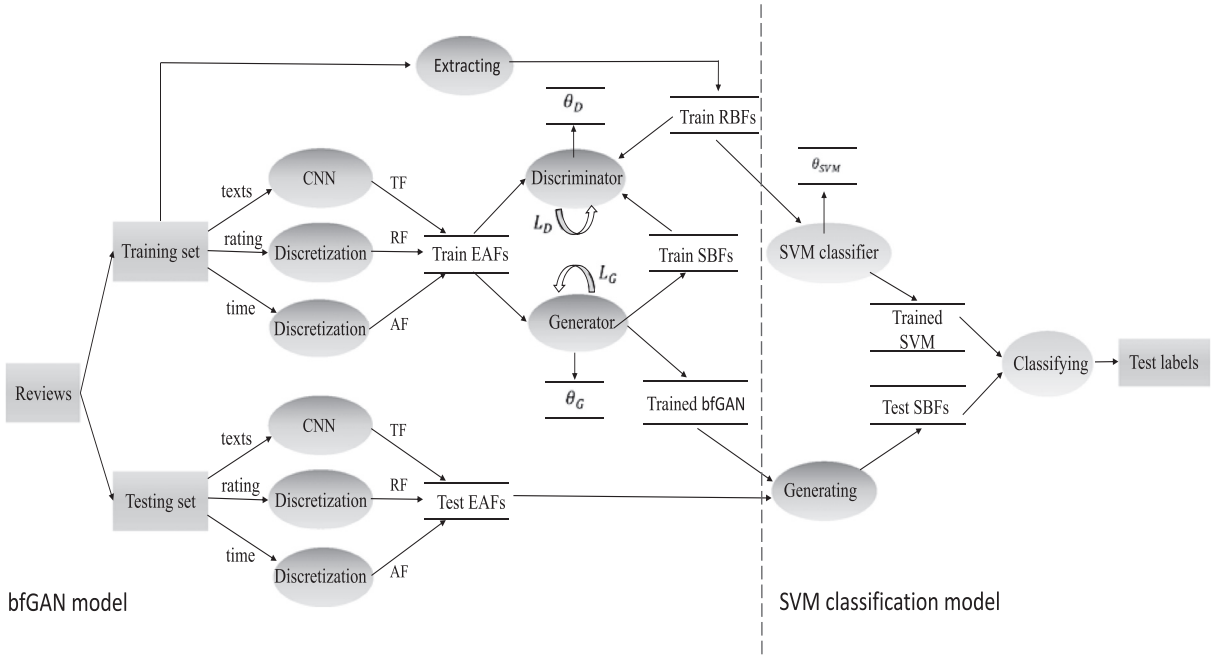


Fig. 2. The overall dataflow chart.

5. Experiments

5.1. Experimental settings

We verify the effectiveness of our proposed model on Hotel and Restaurant datasets which are the subset of the Yelp datasets used in many previous studies [33,34,38]. To tackle the cold-start problem, we use the same split as that in [43,47]. The reviews posted before January 1, 2012 are used as the training data, and the first new reviews posted by the new reviewers after January 1, 2012 are used as the test data.

We use the same SVM method [34,38,43,47] on the same balanced datasets [43,47] to train the classifier on the training data and test it on the test data. We also use the same evaluation metrics including precision (P), recall (R), F1-Score (F1), and accuracy (Acc).

We use three types of EAFs incluing text features (TF), rating features (RF), and attribute features (AF) to generate SBFs. We train one GAN for each type of EAFs using Adam algorithm and set learning rate to 0.00001. We stop iteration when the network becomes stable or the loss in the generator reaches the minimum value in the predefined number of iterations. Following prior studies [43,47], we also pre-train a CNN for the text features. The number of filters in convolution layers is set to 100, which is same as those in [43,47]. The filter size is set to 2.

In the generator network, the dimension of embeddings is 100. The number of neurons in two hidden layers is set to 64 and 32, and tanh is adopted as activation function. The last layer is the mapping layer for generating SBFs, and the number of neurons is 6 which is same as the dimensionality of RBFs.

In the discriminator network, the number of neurons in two hidden layers is also set to 64 and 32, and tanh is adopted as activation function. The number of the neurons in the final classification layer is 1, and sigmoid is adopted as activation function.

5.2. Baseline methods

We compare our model with nine state-of-the-art methods based on linguistic features, behavioral features, and the embeddings of reviews, reviewers, and items. The baselines are listed as follows.

LF [33] captures the linguistic features by extracting bigrams on the labeled review data.

Supervised-CNN [43] uses the same textual information as LF but its features are trained in a supervised convolutional neural network.

LF+BF [33] is a concatenation of linguistic features (LFs) and behavioral features (BFs) including review length, the absolute RD and MCS for the review [47].

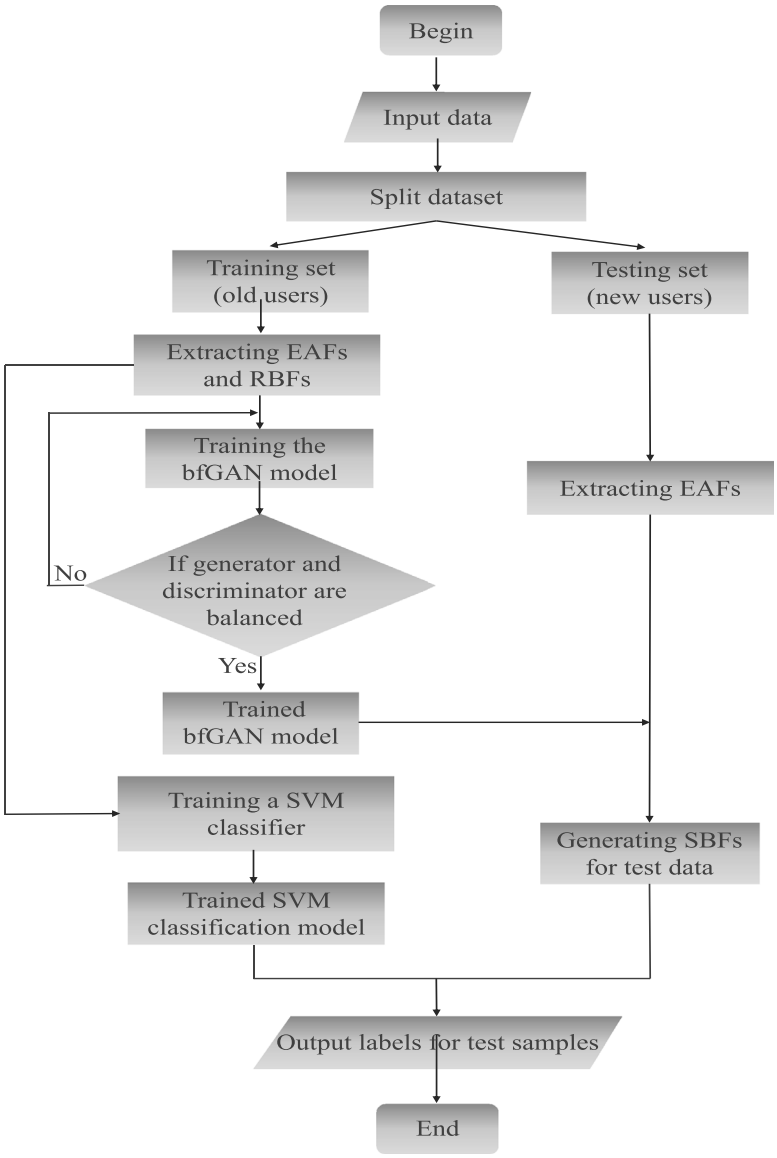


Fig. 3. The overall flowchart.

BF_EditSim+LF [43] first calculates the edit distance between the current review and existing reviews and finds the most similar one, then uses its reviewer's behavioral features as the approximation of the new reviewer.

BF_W2VSim+W2V [43] is similar to the BF_EditSim+LF, but uses the similarity of averaged word embedding (pretrained by Word2vec [29]) to find the most similar review, and concatenates the behavioral features with the average word embedding instead of bigram.

RE* [43] jointly utilizes the TransE [3] to model the behavioral information of reviewers, a pre-trained CNN with same parameters of supervised-CNN to encode the textual information, and a constraint to preserve semantics of the sentiment polarity in rating.

RE+RRE+PRE* [43] is an improved version of RE*, which further concatenates the review embedding, the review's rating embedding and the item's average rating embedding into a long vector as the feature of the review.

AEDA [47] jointly encodes the rich attribute information from reviewers, items, and reviews in addition to the traditional entity information into embeddings. Moreover, it adopts the domain adaptive technique to further improve the performance.

AE [47] refers to a variation of AEDA, which trains on the single domain without domain adaption.

For the baselines, we report the results in [43] and [47] if they have been implemented, since we conduct experiments on the exactly same datasets and same training/testing split. We also use the same hyper-parameters for our model as those in [43,47] for a fair comparison.

Table 2
Comparison with baselines.

No.	Methods	Hotel				Restaurant			
		P	R	F1	Acc	P	R	F1	Acc
(1)	LF	54.5	71.1	61.7	55.9	53.8	80.8	64.6	55.8
(2)	Supervised-CNN	61.2	51.7	56.1	59.5	56.9	58.8	57.8	57.1
(3)	LF+BF	63.4	52.6	57.5	61.1	58.1	61.2	59.6	58.5
(4)	BF_EditSim+LF	55.3	69.7	61.6	56.6	53.9	82.2	65.1	56.0
(5)	BF_W2Vsim+W2V	58.4	65.9	61.9	59.5	56.3	73.4	63.7	58.2
(6)	RE*	62.1	68.3	65.1	63.3	58.4	75.1	65.7	60.8
(7)	RE+RRE+PRE*	63.6	71.2	67.2	65.3	59.0	78.8	67.5	62.0
(8)	AE	76.7	74.2	75.4	75.8	80.3	66.2	72.6	75.0
(9)	AEDA	83.9	74.2	78.7	80.0	82.4	65.1	72.8	75.6
(10)	bfGAN(ours)	81.2	85.7	**83.4**	**83.0**	76.7	73.4	**75.1**	**75.7**

Table 3
Effects of different types of SBFs.

Method	Removed feature	Hotel		Restaurant	
		F1	Acc	F1	Acc
bfGAN	SBFs (TFs)	−1.4	−1.6	−0.1	−0.3
	SBFs (RFs)	−2.4	−2.1	−0.1	+0.1
	SBFs (AFs)	−14.1	−15.0	−11.9	−15.5
AEDA	TFs	+0.4	+0.7	−0.1	−0.1
	RFs	−1.2	−0.9	−1.0	−0.6
	AFs	−22.5	−19.8	−12.1	−12.1

5.3. Comparison with baselines

We conduct the comparison experiments on two Yelp datasets. The results are shown in Table 2. To take a closer look, we further draw the histograms for different methods in terms of their F1 and Accuracy scores in Fig. 4.

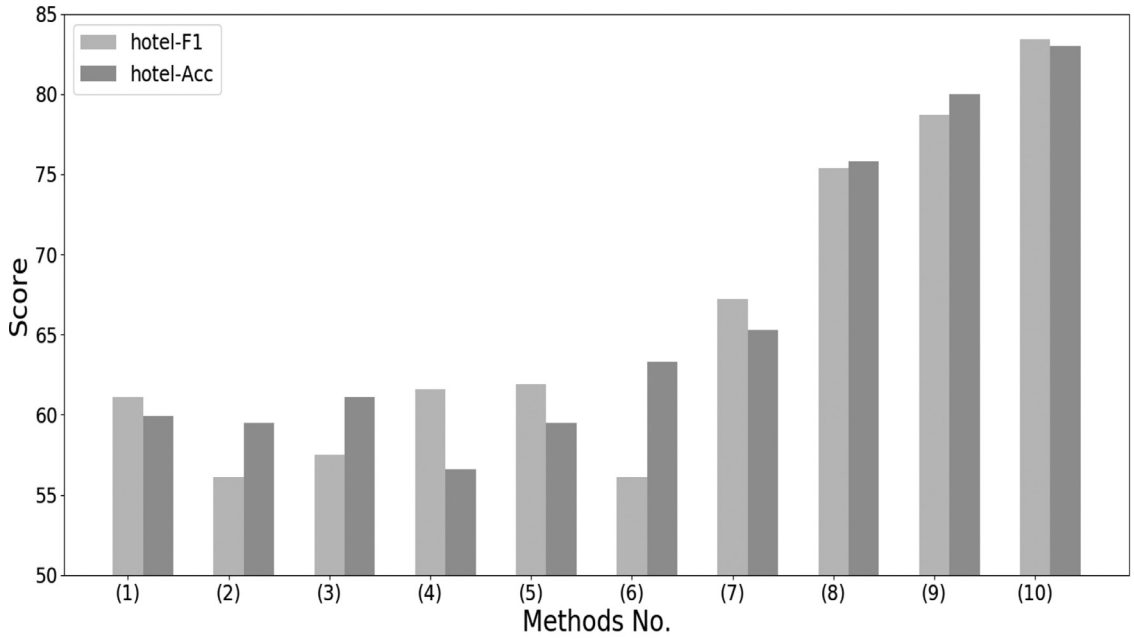
It is clear that our proposed bfGAN achieves the best performance in terms of F1 and Accuracy on both Hotel and Restaurant datasets. We notice that the accuracy of our method over AEDA on Restaurant is tiny. However, it is much better than other baselines on Restaurant and it also outperforms all baselines (including AEDA) on Hotel. Furthermore, the significant improvements of F1 values over all baselines on two datasets clearly demonstrate the effectiveness of our bfGAN model in cold-start spam review detection task. We have the following important observations for Table 2 and Fig. 4.

- LF and Supervised-CNN methods which only use linguistic features are the worst. This is consistent with the results in previous studies [33,34,43,47].
- Adding behavioral features can enhance the performance, as shown by the results of LF+BF, BF_EditSim+LF, and BF_W2Vsim+W2V. However, the improvement is not big. The reason is that it is difficult to extract or find similar behavior features for the reviewer just posting only one review.
- RE*, RE+RRE+PRE*, AE, and AEDA get better results than the above traditional methods since they all adopt embedding technique which can alleviate data sparsity. Besides, AE and AEDA add extra information from attributes and other domains and thus further improve the performance.

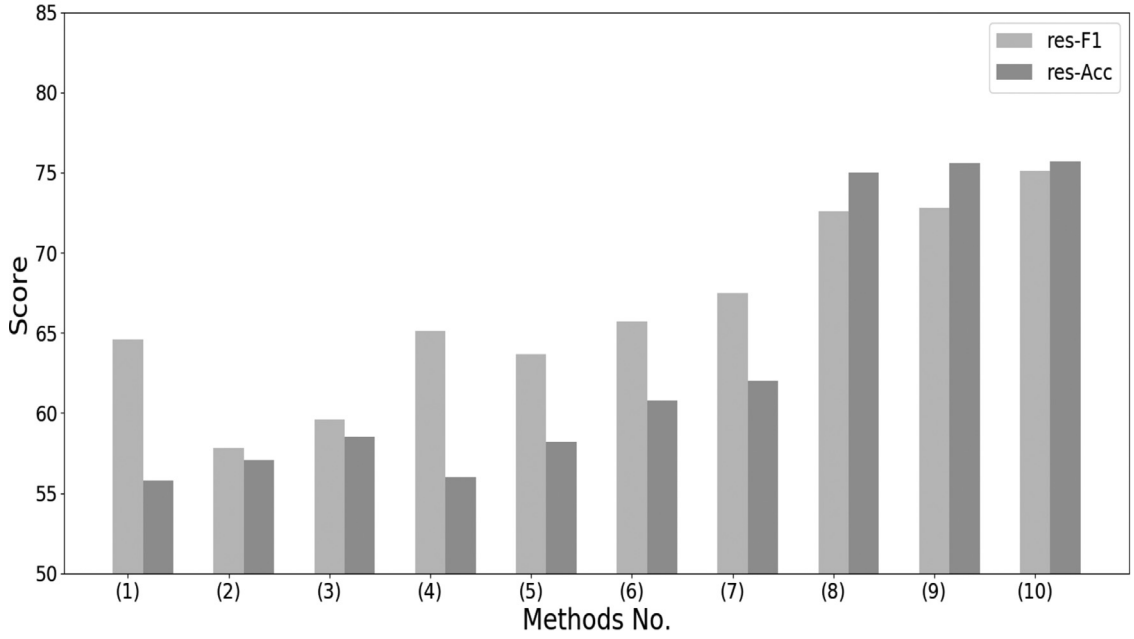
In summary, the baseline methods tackle the spam detection problem either by adding much more information or by using more sophisticated learning technique. Our bfGAN model combines the traditional approaches in finding effective real behavior features and the recent advances in deep learning to generate synthetic behavior features to simulate the real ones. Compared to the state-of-the-art method AEDA, our model uses less information and does not integrate domain adaption into the framework, but it reaches an improvement of 6.0% and 3.2% of F1 over AEDA on Hotel and Restaurant, respectively. This clearly demonstrates that our bfGAN model can generate highly effective SBFs whose distribution is close to that of RBFs and hence achieves significantly better performance than baseline methods.

5.4. Effects of different types of SBFs

As we illustrated in the previous section, we generate one type of SBFs from each of three types of EAFS including text features (TFs), rating features (RFs), and attribute features (AFs). We are interested in which type of SBFs is more effective than others. We evaluate the effects of SBFs by removing this particular type of SBFs from their concatenation when building the classifier and testing on the new users. In addition, since AEDA [47] uses TFs, RFs, and AFs as well, we also list the effects of these features for AEDA for a comparison. The results are shown in Table 3, where a "−" and "+" denotes the decrease and increase of performance, respectively.



(a) Hotel



(b) Restaurant

Fig. 4. Graphical comparison with baselines in terms of accuracy and F1.

From Table 3, we can find that AFs are more important than TFs and RFs on both datasets. The removal of SBFs (AFs) results in a drop of 14.1% and 11.9% of F1 values on Hotel and Restaurant, respectively. This observation is consistent with that in [47], i.e., spammer reviewers usually post reviews right after registering on the system. We also find that the drop of performance by removing SBFs (AFs) in bfGAN is significantly smaller than that by removing AFs in AEDA. Indeed, after the removal of SBFs (AFs) and AFs, the F1 value on Hotel for bfGAN and AEDA becomes 69.3%, and 56.2%, respectively. Notice

Table 4
Comparison between EAFs and SBFs.

	Features	Hotel				Restaurant			
		P	R	F1	Acc	P	R	F1	Acc
TFs	EAFs	58.5	66.4	62.2	59.7	57.8	62.0	59.5	57.2
	SBFs	61.0	69.1	**64.8**	**62.4**	56.3	65.8	**60.7**	**57.4**
RFs	EAFs	71.4	58.5	64.3	67.5	56.2	52.9	54.5	55.8
	SBFs	66.4	65.4	**65.9**	66.1	55.7	60.0	**57.7**	**56.1**
AFs	EAFs	83.8	73.8	78.4	79.2	82.2	63.8	71.8	75.0
	SBFs	77.9	86.2	**81.8**	**80.9**	76.3	73.8	**75.0**	**75.4**

Table 5
Comparison between SBFs and RBFs.

Features	Hotel				Restaurant			
	P	R	F1	Acc	P	R	F1	Acc
RBFs	67.0	96.3	79.0	75.3	55.5	93.5	69.7	59.3
SBFs	81.2	85.7	**83.4**	**83.0**	76.7	73.4	**75.1**	**75.7**

that this reduced F1 score for our incomplete bfGAN model still outperforms all other baselines. This proves that our bfGAN model is not very dependent on a specific type of features, and is more robust than AEDA.

Another interesting finding is that the removal of TFs in AEDA incurs an increase performance on Hotel. The reason may be that the original text features are easy to be forged and adding such features deteriorates the performance. In contrast, in our model, the SBFs generated from TFs contribute positively to the classifier since the generation procedure is under the constraint that SBFs should be close to RBFs.

5.5. Comparison between EAFs and SBFs

Our SBFs are generated from EAFs using a GAN, hence a natural question would be that which ones are more effective, the generated SBFs or the original EAFs? To answer this question, we compare their effects by using the single type of EAFs and its corresponding type of SBFs. The results are shown in Table 4.

From Table 4, it is clear that SBFs generated by our bfGAN model outperform the original EAFs in almost all cases. For example, the F1 value increases from 78.4% to 81.8% on Hotel and from 71.8% to 75.0% on Restaurant. We believe the reason can be due to the effectiveness of behavior features. Although our SBFs are artificial features, they are generated during the contesting against the RBFs and hence have close distribution with that of RBFs. The superior performance of SBFs suggests that the proposed bfGAN model can produce high quality behavior features for the new users using their easily accessible features (EAFs).

5.6. Comparison between SBFs and RBFs

Since in the cold-start settings, the new reviewer just posted one review, we use the following rules to extract RBFs for the new reviewers.

(1) For activity window feature (AW), we simply set its value for new reviewer to 0. The rationale is that the cold-start review is both the first and the last review of this reviewer and there is no time difference between them.
(2) For maximum number of reviews (MNR), we set its value to 1 because this is just the number of reviews the new reviewer posts in one day. Similarly, we set the review count (RC) and maximum content similarity (MCS) to 1.
(3) For the ratio of positive ratings (PR), we set PR to 1 if the rating of the cold-start review is 4 or 5, otherwise to 0. The value of reviewer deviation (RD) is the rating deviation of this review from the rating of other reviewers on the same product.

We now present the results by RBFs and SBFs in Table 5.

From Table 5, it is clear that SBFs significantly outperforms RBFs in terms of both F1 and Accuracy metrics on two datasets. Furthermore, the extremely high Recall value of 96.3% and 93.5% on Hotel and Restaurant indicates that the classifier of RBFs recognizes almost all spam reviews. However, the low Precision and Accuracy scores of RBFs show that many normal reviews are incorrectly predicted as spam ones. The reason is that in cold-start problem legitimate reviewers only write one review as well. This will mislead the classifier. Hence RBFs in the cold-start setting do not work well. In contrast, with the help of GAN, we can construct SBFs from abundant EAFs for effective detection of cold-start spam reviews.

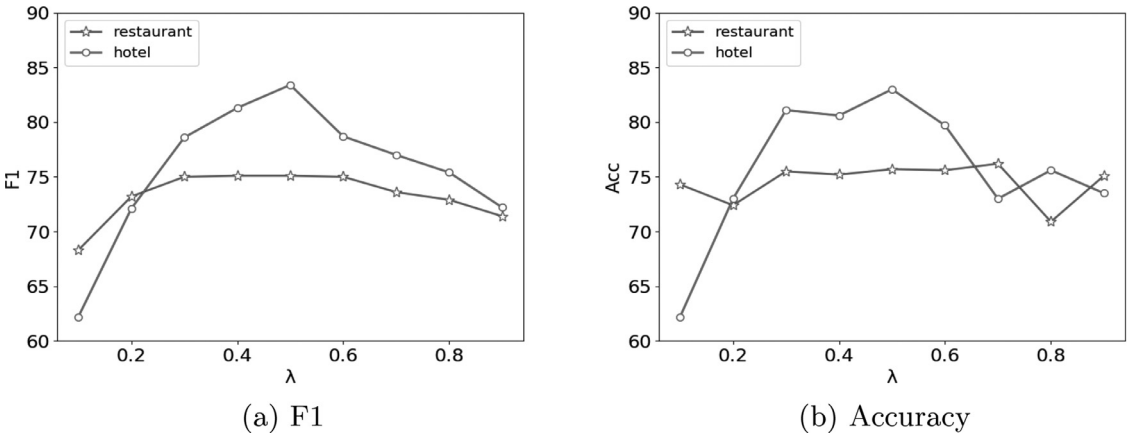


Fig. 5. Effects of balancing factor λ in terms of F1 and Accuracy.

5.7. Effects of balancing factor λ

In order to encourage the generator to generate SBFs close to RBFs, we introduce a closeness loss $\mathcal{L}_c$ and use a factor λ to balance between $\mathcal{L}_c$ and the task loss $\mathcal{L}_t$ in Eq. (3). We investigate its effects and present the results in Fig. 5.

From Fig. 5, it is clear that, with the increase of λ, the classification performance starts to rise and it reaches the peak at $\lambda = 0.5$ in terms of F1.

When λ is set to 0.5, two losses have the same weights and they contribute equally to the generator. The task loss $\mathcal{L}_t$ forces the generator to generate SBFs matching EAFs while the closeness loss $\mathcal{L}_c$ encourages the generated SBFs close to RBFs.

In addition, we observe that when λ is set to 0.1, the results are much worse than those when $\lambda = 0.9$, showing that the task loss has a big impact on affecting the classification results. This is reasonable since the task loss should lead the generation of SBFs. Without the guidance of the task loss, the generated SBFs will collapse to RBFs. In such a case, it is hard to find effective RBFs for new users and hence the performance is poor.

5.8. Convergence evaluation

It is hard to guarantee the convergence of GAN. That is to say, the generator and the discriminator may not reach the equilibrium point. Since the discriminator works based on the synthetic features produced in the generator, we adopt a generator-determined stop criterion in our implementation. We stop iteration while the loss function in the generator reaches the minimum value during the training procedure on a predefined number of iterations. To this end, we first set the predefined number of iterations for TF, RF, and AF to a relatively large value like 500, 100, and 25. We then watch the changes of the scores of loss function $\mathcal{L}_G$ and choose the point with the smallest score. We take the changes of $\mathcal{L}_G$ in generating synthetic behavior features (SBFs) using attribute features (AFs) as an example to show the convergence of the model. The results are shown in Fig. 6.

We can see that it is actually easy to train the model with AFs. In particular, the loss function $\mathcal{L}_G$ can reach a local minimum in about 20 iterations. Similarly, the final iteration number for TF, RF, and AF is set to 300, 21, 16 on Hotel, and 13, 67, 13 on Restaurant, respectively.

5.9. Parameter study

In this subsection, we analyze the impacts of three major parameters in our bfGAN model, including the filter size in CNN, the dimensionality of the embeddings, and the number of neurons in two layers of the network.

The filter in CNN is used to produce a feature map for a set of words within the fixed window size. Normally it is set to a small value around 3 for text convolution operations [19]. Hence we vary the filter size in the set of {2, 3, 4}.

The results are shown in Table 6.

From Table 6, we can find that the filter size does not have big impacts on the performance. For example, the F1 score on Hotel for Filter Size = 2 is same as that for Filter Size = 3, and this also happens to F1 on Restaurant for Filter Size = 3 and Filter Size = 4.

The dimensionality of embeddings in our network denotes the degree of discretization of EAFs (note the embedding in CNN is fixed to 100 following the settings in [43,47]). We examine the impacts of dimensionality by varying it in {50, 100, 150, 200, 250} and show the results in Table 7.

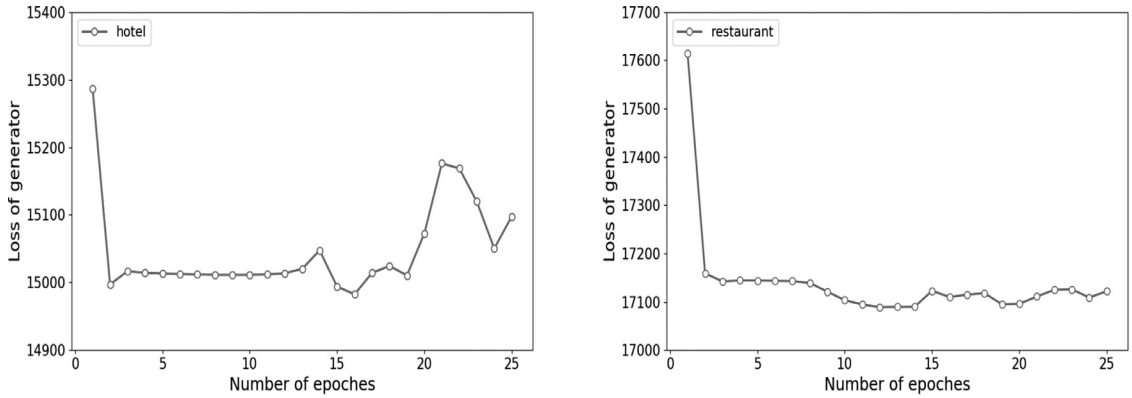


Fig. 6. Convergence test in generating SBFs using AFs.

Table 6
Impacts of filter size in CNN.

Filter Size	Hotel				Restaurant			
	P	R	F1	Acc	P	R	F1	Acc
2	81.2	85.7	83.4	**83.0**	76.7	73.4	75.1	**75.7**
3	78.2	89.4	83.4	82.3	74.8	76.0	**75.4**	75.5
4	83.9	86.6	**83.7**	82.9	75.7	75.0	**75.4**	75.2

Table 7
Impacts of dimensionality.

Dimensionality	Hotel				Restaurant			
	P	R	F1	Acc	P	R	F1	Acc
50	78.6	84.8	81.6	80.9	68.2	79.6	73.5	71.3
100	81.2	85.7	**83.4**	83.0	76.7	73.4	**75.1**	**75.7**
150	85.6	79.7	82.6	**83.2**	74.4	75.3	74.9	74.7
200	84.2	78.8	81.4	82.0	77.5	72.3	74.8	75.7
250	76.2	85.7	80.7	79.5	68.1	78.4	72.9	70.9

Table 8
Impacts of the number of neurons.

Neuron number	Hotel				Restaurant			
	P	R	F1	Acc	P	R	F1	Acc
(128, 128)	81.4	84.8	83.1	82.7	71.5	78.0	74.6	73.4
(128, 64)	82.9	82.9	82.9	82.9	71.2	71.9	74.8	73.6
(64, 64)	77.5	87.6	82.3	81.1	73.7	73.2	73.3	73.3
(64, 32)	81.2	85.7	**83.4**	**83.0**	76.7	73.4	75.1	75.7
(32, 32)	79.8	85.7	82.7	82.0	76.7	74.1	**75.4**	**75.8**

From Table 7, we can find that if the dimensionality is too large or too small, the results will drop a lot. Indeed, the best F1 value occurs when the dimensionality equals 100 on both the Hotel and Restaurant datasets. If the dimensionality is too large, lots of samples may fall into the same interval, and this will reduce the diversity of the data. If the dimensionality is too small, the data points will become very sparse, which will also damage the performance.

The number of neurons indicates the complexity of the model. A large number of neurons denotes the high level of non-linearity and complexity of the network. As suggested in [30], the number of neurons is usually set to be 2^m, where $m = \log_2(n)$ and n is the dimensionality of the input vector. Since n is 100 in our case, it is good to set m between 6 and 7. Consequently, the number of neurons is between 64 and 128. In addition, we set the number of neurons in the second layer to be half of the first layer, i.e., 2^{m-1}, since we tend to extract a denser and more effective features from the first layer. We investigate the impacts of the number of neurons in our two-layer network by vary their values in {(128, 128), (128, 64), (64, 64), (64, 32), (32, 32)}. The results are shown in Table 8.

From Table 8, it is clear that the best results are obtained with the (64, 32) and (32, 32) setting on Hotel and Restaurant dataset, respectively. This infers that a simpler model is more effective than a more complex one, as the latter might incur overfitting. Moreover, the decreasing number of neurons between two layers is usually better than the same number setting. For example, the (64, 64) setting on Restaurant results in a 73.3 F1 value while the (64, 32) one leads to a 75.1 score, showing a significant improvement. This also goes for (128, 128) and (128, 64) settings. Though the (32, 32) setting is better than (64, 32) on Restaurant, their difference is trivial.

6. Conclusion

In this paper, we propose a novel bfGAN model for cold-start spam review detection. To address the problem of lacking effective real behavior features (RBFs) for new users in cold-start scenario, we first propose to distinguish easily accessible features (EAFs) from RBFs and select three types of EAFs including text, rating, and attribute features. We then design a GAN framework to generate synthetic behavior features (SBFs) using EAFs. Since SBFs and RBFs are connected via the third party EAFs, we cannot directly use SBFs to contest against RBFs like texts and images in traditional GAN, we further present a new implementation of GAN by incorporating an extra loss to explicitly guide SBFs to be close to RBFs. We conduct extensive experiments on two Yelp datasets. Results demonstrate that our bfGAN model significantly outperforms the state-of-the-art baselines.

While our model is effective in detecting cold-start spam reviews, it can be further improved in the following issues. The first is that we still require the new users have easily accessible features including the text features, rating features, and attribute features. Some websites may hide the users' attribute features for the reason of privacy, which might deteriorate the performance of the bfGAN model. The second is that it is not easy to train the GAN module in some cases. In the future, we are going to employ other generative models such as variational auto-encoder to produce synthetic behavior features. We also plan to explore domain adaption technique for more effective representations.

Declaration of Competing Interest

We wish to confirm that there are no known conflicts of interest associated with this publication and there has been no significant financial support for this work that could have influenced its outcome.

CRediT authorship contribution statement

Xiaoya Tang: Investigation, Data curation, Conceptualization, Methodology, Software, Visualization, Validation. **Tieyun Qian:** Supervision, Investigation, Conceptualization, Methodology, Writing - original draft, Writing - review & editing. **Zhenni You:** Data curation, Investigation.

Acknowlgedgment

The work described in this paper has been supported in part by the NSFC projects (61572376, 91646206).

References

[1] L. Akoglu, R. Chandy, C. Faloutsos, Opinion fraud detection in online reviews by network effects, in: ICWSM, 2013, pp. 2–11.
[2] M. Anderson, J. Magruder, Learning from the crowd: regression discontinuity estimates of the effects of an online review database, Econ. J. 122 (563) (2012) 957–989.
[3] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, O. Yakhnenko, Translating embeddings for modeling multi-relational data, in: NIPS, 2013, pp. 2787–2795.
[4] L. Cao, Coupling learning of complex interactions, Inf. Process. Manag. 51 (2) (2015) 167–186.
[5] L. Cao, P.S. Yu, Behavior Computing: Modeling, Analysis, Mining and Decision, Springer Publishing Company Incorporated, 2014.
[6] E. Cippitelli, F. Fioranelli, E. Gambi, S. Spinsante, Radar and RGB-depth sensors for fall detection: areview, IEEE Sens. J. 17 (12) (2017) 3585–3604.
[7] C. Debes, A. Merentitis, S. Sukhanov, M. Niessen, N. Frangiadakis, A. Bauer, Monitoring activities of daily living in smart homes: understanding human behavior, IEEE Signal Process. Mag. 33 (2) (2016) 81–94.
[8] G. Fei, A. Mukherjee, B. Liu, M. Hsu, M. Castellanos, R. Ghosh, Exploiting burstiness in reviews for review spammer detection, in: ICWSM, 2013, pp. 175–184.
[9] S. Feng, R. Banerjee, Y. Choi, Syntactic stylometry for deception detection, in: ACL, 2012, pp. 171–175.
[10] T. Fornaciari, M. Poesio, Identifying fake Amazon reviews as learning from crowds, in: ACL, 2014, pp. 279–287.
[11] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative adversarial nets, in: NIPS, 2014, pp. 2672–2680.
[12] S. Gupta, A. Khattar, A. Gogia, P. Kumaraguru, T. Chakraborty, Collective classification of spam campaigners on Twitter: a hierarchical meta-path based approach, in: WWW, 2018, pp. 529–538.
[13] Z. Hai, P. Zhao, P. Cheng, P. Yang, X.L. Li, G. Li, Deceptive review spam detection via exploiting task relatedness and unlabeled data, in: EMNLP, 2016, pp. 1817–1826.
[14] D. Hovy, The enemy in your own camp: how well can we detect statistically-generated fake reviews–an adversarial study, in: ACL, 2016, pp. 351–356.
[15] N. Jindal, B. Liu, Opinion spam and analysis, in: WSDM, 2008, pp. 219–230.
[16] N. Jindal, B. Liu, E.P. Lim, Finding unusual review patterns using unexpected rules, in: CIKM, 2010, pp. 1549–1552.
[17] S. KC, A. Mukherjee, On the temporal dynamics of opinion spamming: case studies on yelp, in: WWW, 2016, pp. 369–379.
[18] S. Kim, H. Chang, S. Lee, M. Yu, J. Kang, Deep semantic frame-based deceptive opinion spam analysis, in: CIKM, 2015, pp. 1131–1140.
[19] Y. Kim, Convolutional neural networks for sentence classification, in: EMNLP, 2014, pp. 1746–1751.

[20] F. Li, M. Huang, Y. Yang, X. Zhu, Learning to identify review spam, IJCAI, 2488, 2011.
[21] H. Li, B. Liu, A. Mukherjee, J. Shao, Spotting fake reviews using positive-unlabeled learning, in: ICDM, 2014, pp. 899–904.
[22] J. Li, W. Monroe, T. Shi, S. Jean, A. Ritter, D. Jurafsky, Adversarial learning for neural dialogue generation, in: EMNLP, 2017, pp. 2151–2156.
[23] J. Li, M. Ott, C. Cardie, E. Hovy, Towards a general rule for identifying deceptive opinion spam, in: ACL, 2014, pp. 1566–1576.
[24] E.P. Lim, V.A. Nguyen, N. Jindal, B. Liu, H.W. Lauw, Detecting product review spammers using rating behaviors, in: CIKM, 2010, pp. 939–948.
[25] L. Liu, L. Cheng, Y. Liu, Y. Jia, D.S. Rosenblum, Recognizing complex activities by a probabilistic interval-based model, in: AAAI, 2016, pp. 1266–1272.
[26] Y. Liu, L. Nie, L. Han, L. Zhang, D.S. Rosenblum, Action2activity: recognizing complex activities from sensor data, in: AAAI, 2015, pp. 1617–1623.
[27] Y. Liu, L. Nie, L. Liu, D.S. Rosenblum, From action to activity: sensor-based activity recognition, Neurocomputing 181 (2016) 108–115.
[28] M. Luca, Reviews, Reputation, and Revenue: The Case of yelp.com, 2011. Harvard Business School Working Papers
[29] T. Mikolov, I. Sutskever, K. Chen, G.S. Corrado, J. Dean, Distributed representations of words and phrases and their compositionality, in: NIPS, 2013, pp. 3111–3119.
[30] G. Mirchandani, W. Cao, On hidden nodes for neural nets, IEEE Trans. Circ. Syst. 36 (5) (1989) 661–664.
[31] A. Mukherjee, A. Kumar, B. Liu, J. Wang, M. Hsu, M. Castellanos, R. Ghosh, Spotting opinion spammers using behavioral footprints, in: KDD, 2013, pp. 632–640.
[32] A. Mukherjee, B. Liu, N. Glance, Spotting fake reviewer groups in consumer reviews, in: WWW, 2012, pp. 191–200.
[33] A. Mukherjee, V. Venkataraman, B. Liu, N. Glance, Fake Review Detection: Classification and Analysis of Real and Pseudo Reviews, Technical Report UIC-CS-2013–03 Tech. Rep., University of Illinois at Chicago, 2013.
[34] A. Mukherjee, V. Venkataraman, B. Liu, N.S. Glance, What yelp fake review filter might be doing? ICWSM, 2013.
[35] H.F. Nweke, Y.W. Teh, M.A. Al-garadi, U.R. Alo, Deep learning algorithms for human activity recognition using mobile and wearable sensor networks: state of the art and research challenges, Expert Syst Appl 105 (2018) 233–261.
[36] M. Ott, Y. Choi, C. Cardie, J.T. Hancock, Finding deceptive opinion spam by any stretch of the imagination, in: ACL, 2011, pp. 309–319.
[37] L.J. Ratliff, S.A. Burden, S.S. Sastry, Characterization and computation of local Nash equilibria in continuous games, in: IEEE TNN, 2013, pp. 917–924.
[38] S. Rayana, L. Akoglu, Collective opinion spam detection: bridging review networks and metadata, in: KDD, 2015, pp. 985–994.
[39] Y. Ren, Y. Zhang, Deceptive opinion spam detection using neural network, in: COLING, 2016, pp. 140–150.
[40] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, X. Chen, Improved techniques for training GANs, in: NIPS, 2016, pp. 2234–2242.
[41] G. Wang, S. Xie, B. Liu, S.Y. Philip, Review graph based online store review spammer detection, in: ICDM, 2011, pp. 1242–1247.
[42] X. Wang, K. Liu, S. He, J. Zhao, Learning to represent review with tensor decomposition for spam detection, in: EMNLP, 2016, pp. 866–875.
[43] X. Wang, K. Liu, J. Zhao, Handling cold-start problem in review spam detection by jointly embedding texts and behaviors, in: ACL, 2017, pp. 366–376.
[44] S. Xie, G. Wang, S. Lin, P.S. Yu, Review spam detection via temporal pattern discovery, in: KDD, 2012, pp. 823–831.
[45] Q. Xu, H. Zhao, Using deep linguistic features for finding deceptive opinion spam, in: COLING, 2013, pp. 1341–1350.
[46] J. Yin, Z. Zheng, L. Cao, Y. Song, W. Wei, Efficiently mining top-k high utility sequential patterns, in: ICDM, 2013, pp. 1259–1264.
[47] Z. You, T. Qian, B. Liu, An attribute enhanced domain adaptive model for cold-start spam review detection, in: COLING, 2018, pp. 1884–1895.
[48] L.T. Yu, W.N. Zhang, J. Wang, Y. Yu, Seqgan: sequence generative adversarial nets with policy gradient, AAAI, 2016.
[49] C. Zhu, W. Sheng, M. Liu, Wearable sensor-based behavioral anomaly detection in smart assisted living systems, IEEE Trans. Autom. Sci. Eng. 12 (4) (2015) 1225–1234.