Received: 22 December 2019 | Revised: 25 March 2020 | Accepted: 6 April 2020

DOI: 10.1111/coin.12329

SPECIAL ISSUE ARTICLE

Sentiment classification with adversarial learning and attention mechanism

Yueshen Xu[1] | Lei Li[1] | Honghao Gao[2] | Lei Hei[3] | Rui Li[1] | Yihao Wang[4]

[1]School of Computer Science and Technology, Xidian University, Xi'an, China

[2]Computing Center, Shanghai University, Shanghai, China

[3]Center of Journal Publication, Xidian University, Xi'an, China

[4]Department of Biomedical Engineering, University of Melbourne, Melbourne, Victoria, Australia

Correspondence
Honghao Gao, Computing Center, Shanghai University, Shanghai, China.
Email: gaohonghao@shu.edu.cn

Funding information
National Natural Science Foundation of China, Grant/Award Number: 61702391; Natural Science Foundation of Shaanxi Province, Grant/Award Number: 2018JQ6050

Abstract

Sentiment classification is a key task in sentiment analysis, reviews mining, and other text mining applications. Various models have been proposed to build sentiment classifiers, but the classification performances of some existing methods are not good enough. Meanwhile, as a subproblem of sentiment classification, positive and unlabeled learning (PU learning) problem widely exists in real-world cases, but it has not been given enough attention. In this article, we aim to solve the two problems in one framework. We first build a model for traditional sentiment classification based on adversarial learning, attention mechanism, and long short-term memory (LSTM) network. We further propose an enhanced adversarial learning method to tackle PU learning problem. We conducted extensive experiments in three real-world datasets. The experimental results demonstrate that our models outperform the compared methods in both traditional sentiment classification problem and PU learning problem. Furthermore, we study the effect of our models on word embedding. Finally, we report and discuss the sensitivity of our models to parameters.

KEYWORDS

adversarial learning, attention mechanism, LSTM network, PU learning problem, sentiment classification

1 | INTRODUCTION

Sentiment analysis is one of key tasks in natural language processing (NLP) and has received a lot of attention in recent years.[1,2] As a basic problem of sentiment analysis, *sentiment classification* aims to classify reviews into different sentimental polarities.[3] Sentiment classification is extremely useful in many real-world applications, such as mining sentimental polarities from movie reviews, product reviews, and news reviews.[4] Along with the rapid development of e-commerce sites and social networking sites, the volume of reviews increase dramatically, and those online sites need to analyze the polarities from large volumes of reviews with high accuracy.[5] So it becomes an urgent task to solve the sentiment classification problem and to improve the classification accuracy.

Many works have tried to address sentiment classification using various techniques, including K-nearest neighbor (KNN), support vector machine (SVM), Naïve Bayes, neural networks, and some other methods.[5-7] Among these, the methods extended from neural networks have achieved great performances. Meanwhile, as an effective technique for improving the robustness of machine learning methods, *adversarial learning* has been also studied in tasks of text mining and natural language processing, such as text classification and text generation.[8-10] In this article, we exploit adversarial learning in solving sentiment classification problems. Besides the tasks of natural language processing, adversarial learning has been also studied in some other artificial intelligence tasks, such as image generation.[11] However, there are few works exploiting adversarial learning in sentiment classification. For adversarial learning, we have the following observations and motivations.

1. Traditional classifiers are likely to suffer from overfitting problem.[12] That is, a classifier overwhelmingly fits a certain words distribution in training reviews set and is trained to obtain a collection of parameters, but fails to fit the words distribution in test reviews set or new reviews set.
2. Adversarial examples are inputs formed by adding small perturbations with the intent of causing classifiers (eg, neural networks) to misclassify[13,14] and can attack the generalization and fitness of classifiers. It can be inferred that a new classifier that is capable of resisting the attack of adversarial examples can achieve promising performance.
3. Previous works have shown that models based on neural networks are vulnerable to adversarial examples. To defend the attack from adversarial examples, *adversarial learning* is developed by taking the adversarial examples into training phase.[12] Adversarial learning can enhance the robustness of models to adversarial examples and can improve model generalization. In this article, we treat the sentences with perturbed word embeddings as adversarial examples.
4. Sentiment classifiers based on neural networks usually take the distributed representation of words (ie, word embeddings) as input. Models for sentiment classification depend heavily on sentimental words to produce classification labels.[15]

   However, word embeddings lack the ability to effectively distinguish the semantic difference of some opposite sentimental words. For example, the word embedding of sentimental word "good" can be close to that of the opposite sentimental word "bad," as the two words are likely to be involved in a similar context of words in sentences. In sentiment classification problems, such potential inability in distinguishing opposite words will harm the performance of classifiers that are based on neural networks. Through experiments, we find that adversarial learning is capable of improving the quality of word embeddings, including those of sentimental words. We will report the results of word embeddings in the experiment section.

Besides the traditional sentiment classification problem, in this article, we also aim to solve the positive and unlabeled learning (PU learning) problem. In PU learning problem, there only exist positive labeled and unlabeled reviews, without any negative labeled reviews. PU learning problem indeed exists in real-world cases and takes an important role in sentiment analysis.[16] PU learning problem was studied by previous works.[17,18] On the one hand, as a subproblem of sentiment classification, PU learning problem is closely related to traditional sentiment classification. We aim to bring a deep understanding to PU learning problem, by studying PU learning problem and traditional sentiment classification together. On the other hand, in real-world cases, e-commerce sites and social networking sites can confront traditional sentiment classification as well as PU learning problem. For example, in some e-commerce websites, the polarities of many reviews are not labeled manually or cannot be labeled automatically as there are no associated ratings. It will be very helpful to dig negative reviews out to better evaluate the quality of commodities. Meanwhile, with the assistance of mined negative reviews, the producers can improve their commodities correspondingly. It will also improve the applicability of the proposed solution if we can discuss the two problems jointly. In this article, we aim to solve PU learning problem based on the model that is built for traditional sentiment classification. The two models will form the complete solution and will be evaluated in the same datasets. Our goal is that in real-world applications, no matter for the situation of solving traditional sentiment classification problem, or for the situation of solving PU learning problem, our models can be employed. Note that, it is more difficult to exploit adversarial learning in PU learning problem, due to the following two reasons.

1. Adversarial learning requires that all training data have been labeled, but there are no negative labeled reviews in PU learning problem.
2. The evaluation metric for PU learning problem is usually F1-score, precision or recall, which cannot be modeled in the current loss function of adversarial learning, which increases the difficulty in optimization during training process.

In this article, we aim to solve the sentiment classification problems comprehensively under one framework of adversarial learning and neural network. First, we solve the traditional sentiment classification problem, in which we have positive labels as well as negative labels in reviews. We then solve the PU learning problem, where only positive labels are given. In the proposed solution of PU learning problem, we first identify negative reviews from unlabeled reviews. Then, we build an attention-based long short-term memory (LSTM) network, enhanced with an improved adversarial learning method. We evaluate our models in three real-world datasets, for both traditional sentiment classification and PU learning problem. The experimental results demonstrate that our models achieve the best performance, compared with a series of existing methods.

In summary, the contributions of this article are as follows.

1. We propose a solution with two models, to solve traditional sentiment classification problem and PU learning problem, respectively. The proposed solution is based on adversarial learning and attentive LSTM network.
2. In PU learning problem, the procedure that distinguishes positive labels from unlabeled texts is likely to introduce noise into training review texts, which can further hurt the classification

performance. To tackle this issue, we propose an enhanced adversarial learning method, adding a new perturbation to the word embeddings in LSTM network.
3. We study the effect of our models on word embedding, especially the relationship of word embedding with adversarial learning. This study exploits a new reason to supplement the explanations of why adversarial learning works in sentiment classification.
4. We conduct comparison experiments in three real-world datasets, and the experimental results demonstrate that our models can achieve better performance than compared methods. We conduct comparison experiment on training time of different models and also conduct sensitivity experiments to give instructions for selecting parameters.

The rest of this article is organized as follows: Section 2 discusses the related work and Section 3 elaborates the details of our models. Section 4 presents the experimental results and gives further analysis. Section 5 reports the sensitivity experimental results and analyzes the parameters selection. Section 6 concludes the whole article and discuss the future work.

2 | RELATED WORK

There have been many works studying sentiment classification, which employ a variety of methods, including machine learning methods (eg, KNN, SVM, and Naïve Bayes) and neural network methods.[6,19-21] PU learning (positive and unlabeled learning) problem is a subproblem of sentiment classification, where there are no negative labels in corpus, but only positive and unlabeled reviews. PU learning problem also exists in real-world e-commerce sites and social networking sites. As for adversarial learning, it has been verified to be effective to improve the robustness of models in many applications.

In sentiment classification, Liu[3] gave a comprehensive explanation of the whole sentiment analysis problem, which covered sentiment classification problem in detail. Pang et al[22] studied several basic machine learning methods that could be used to solve sentiment classification, such as Naïve Bayes, maximum entropy and support vector machine (SVM). In recent years, neural network models have achieved superior performance over traditional machine learning methods. The authors in Reference 23 provided an extensive survey on the application of neural network models in sentiment analysis, which also covered sentiment classification.

In detail, convolutional neural networks (CNN) have been employed in sentiment classification problem.[24-28] The authors in Reference 25 introduced CNN to solve sentiment classification in short review texts. Recurrent neural network (RNN) and the extended models, such as gated recurrent unit (GRU) and long short-term memory (LSTM), were also employed in sentiment classification problem.[4,29] Tang et al[4] utilized GRU to capture the sentiments in reviews. The authors in Reference 29 tried to solve a fine-grained sentiment classification problem, in which the sentiment was learned specific to each aspect. Zhou et al[30] designed an attention-based LSTM network for crosslingual sentiment classification. In this article, our proposed solution is not for any specific scenario of sentiment classification problems, such as aspect-level or crosslingual sentiment classification, but for the general sentiment classification problem. We will also study the incorporation of neural network and adversarial learning in sentiment classification.

PU learning problem was first studied in Reference 16. In PU learning problem, there is a preliminary task to construct a classifier to identify negative-labeled reviews from positive and unlabeled review texts. Li and Liu[31] proposed a whole framework and identified negative-labeled reviews using Rocchio technique. Du Plessis et al[32] studied the design of the loss function in

PU learning problem. The authors established the generalization error bounds for loss function in PU learning problem. Ren et al[18] and Li et al[17] focused on a specific but valuable problem, that was to detect deceptive reviews from positive and unlabeled reviews. In this article, we aim to solve PU learning problem in one framework as traditional sentiment classification, which is also based on adversarial learning and attentive neural network. The procedure for identifying negative-labeled reviews is highly likely to bring erroneous labels. One typical case is the false negative reviews, which should be positive reviews, but are wrongly labeled to be negative. These wrongly labeled reviews can be treated as noise in training data, which will further harm the performance of sentiment classifiers. We design a new adversarial learning method to attack these erroneous labels and further improve the classification performance.

Adversarial learning aims to improve the robustness of machine learning models by exposing a model to adversarial examples during training process. Adversarial learning was first introduced in the problem of image classification, where the input image pixels were continuous values,[12] and researchers studied adversarial learning technique from many aspects.[33,34] The authors in Reference 14 proposed a new algorithm of crafting adversarial examples. Han et al[35] provided an overview of the application of adversarial learning in sentiment analysis. Miyato et al[8] adapted adversarial learning to solve text classification in a semisupervised setting. Wu et al[36] employed adversarial learning in relation extraction problem and proposed an improved neural network architecture.

Sachan et al[37] proposed a mixed loss that combined entropy minimization loss, adversarial loss, and virtual adversarial loss, promoting the accuracy in text classification task. Adversarial learning has been also studied in domain adaption problem. Ganin et al[38] constructed a classifier with domain transfer ability and achieved good performance in sentiment analysis tasks. Crafting visible adversarial examples and finding interpretable textual adversarial examples in text domain have been given attention in these years.[39,40] In this article, instead of crafting adversarial examples, we exploit the potential of adversarial learning in sentiment analysis, including traditional sentiment classification problem and PU learning problem.

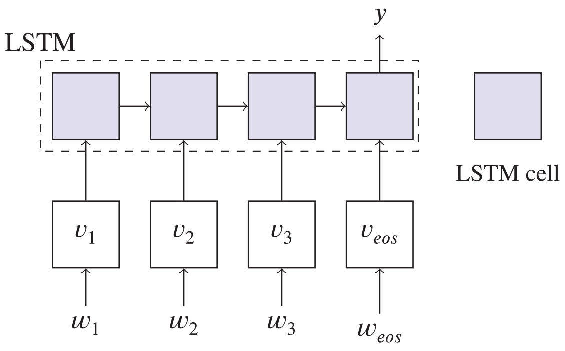
3 | ADVERSARIAL LEARNING FOR SENTIMENT CLASSIFICATION

In this section, we first state the base models employed in our methods, and then elaborate the two models, which are for traditional sentiment classification and PU learning problem, respectively.

3.1 | The base models

Word embedding. As a way of distributed representation, word embedding is prevailingly used as the input of words in NLP tasks, especially in the case that the base models are extended from neural networks. As a technique of feature extraction from words, word embedding represents a word as a vector containing continuous real numbers, instead of purely discrete word tokens (ie, one-hot representation). The word embedding results can be represented by a matrix $\mathbf{W}$, in which one column vector denotes the distributed representation of one word, and the size of rows is the dimension. $\mathbf{W}$ can be learned using neural network tools, such as *word2vec* and *global vector* (GloVe).[41,42]

FIGURE 1 The basic LSTM model for classification task in NLP



LSTM network. Recurrent neural network (RNN) takes sequential data as input and finishes the computation via recursive cells. Standard RNN has several problems in training process, such as gradient vanishing and gradient exploding. To address these issues, LSTM network is developed and achieves superior performance.[43] Formally, each cell in LSTM is computed as follows.

$$X = \begin{bmatrix} h_{t-1} \\ x_t \end{bmatrix}, \tag{1}$$

$$f_t = \sigma(W_f \cdot X + b_f), \tag{2}$$

$$i_t = \sigma(W_i \cdot X + b_i), \tag{3}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tanh(W_c \cdot X + b_c), \tag{4}$$

$$o_t = \sigma(W_o \cdot X + b_o), \tag{5}$$

$$h_t = o_t \odot \tanh(c_t). \tag{6}$$

At time step t, the previous hidden output h_{t-1} and the current input x_t together form the input X (see Equation (1)). There are three gates in an LSTM cell, which are forget gate, input gate, and output gate. Forget gate outputs a value in $[0, 1]$, to indicate the amount of information from previous cell that needs to be dumped in Equation (2). Input gate first decides those values that LSTM will update by i_t in Equation (3) and further computes a candidate cell state c_t using Equation (4). Finally, the output gate decides which part of the candidate cell states will be outputted, and the cell output h_t is computed by Equation (6).

Let T be a piece of review represented by a sequence of m words, as $T = \{w_t | t = 1, \dots, m\}$, and T is tagged with a label as y. Each word w_t is embedded into a k-dimensional word vector $v_t = \mathbf{W} \times w_t$, where $\mathbf{W} \in \mathbb{R}^{k\times|V|}$ is a word embedding matrix to be learned, and V denotes the vocabulary. Figure 1 shows the basic LSTM model for classification task in NLP. w_{eos} denotes the end mark of a review, and v_{eos} is the word embedding result of w_{eos}.

Attention mechanism. In recent years, attention mechanism has become a compelling technique in sequence models, which can improve the capability of models in handling long-range dependencies.[44] In NLP tasks, attention mechanism gives the model a chance to capture the important part of the input that needs more attention. Guided by a weight vector learned from the input text and the result that is produced so far, attention-based model captures more information based on more comprehensive modeling of the input. In detail, we learn an attention vector $\vec{\boldsymbol{\alpha}}$ as follows.

$$u_i = \tanh(W_s h_i + b_s), \tag{7}$$

$$\alpha_i = \frac{\exp(u_i^T u_s)}{\sum_i \exp(u_i^T u_s)}, \tag{8}$$

$$\omega = \sum_i \alpha_i h_i, \tag{9}$$

where α_i denotes each element in $\vec{\alpha}$. The final output of an attentive LSTM is ω (see Equation (9)), which can be treated as a weighted sum over all outputs of all cells in LSTM. In sentiment classification, the attention weights of sentimental words are usually larger than other ordinary words.[15] With the output ω of the attentive LSTM network, a fully connected layer and a softmax nonlinear layer are used to map ω to the probability distribution over each class and further to obtain the label y.

3.2 | The built model in traditional sentiment classification

Adversarial learning is an approach of improving classifiers' performance through promoting the competence of resisting noise and data unbalance. Formally, $\vec{x} = \{T_n\}$ $(n = 1, 2, \dots, N)$ is the set of review texts and taken as the input, where N is the number of labeled reviews. T_n denotes a piece of review and y_n is the associated label of T_n indicating the sentimental polarity (positive or negative). Inspired by the idea in Reference 13, we build a Θ-parameterized classifier, where y_n will be predicted by $p(y_n|T_n;\Theta)$. The loss function of the classifier is defined as the negative log-likelihood of the labels over all review texts.

$$L(\vec{x};\Theta) = -\sum_{n=1}^{N} \log p(y_n|T_n;\Theta). \tag{10}$$

The proposed adversarial learning method adds perturbations $\vec{e}_{\text{adv}}$ to the input $\vec{x}$ and optimizes the loss function as below.

$$L(\vec{x};\Theta) = L(\vec{x} + \vec{e}_{\text{adv}};\Theta) \tag{11}$$

$$\vec{e}_{\text{adv}} = \underset{\|\vec{e}\| \le \epsilon}{\arg\max}\, L(\vec{x} + \vec{e};\hat{\Theta}), \tag{12}$$

where $\hat{\Theta}$ is a constant copy of the current Θ, $\vec{e}$ denotes the perturbations, and ϵ is the threshold of the norm of $\vec{e}$ (ie, the perturbations) that is used to select $\vec{e}_{\text{adv}}$. The following task is to identify $\vec{e}_{\text{adv}}$ that will maximize the loss function $L(\vec{x} + \vec{e}_{\text{adv}};\Theta)$ (Equation (11)). So $\vec{e}_{\text{adv}}$ can be regarded as the worst case of perturbations $\vec{e}$ against the current classifier $p(y_n|T_n;\Theta)$.

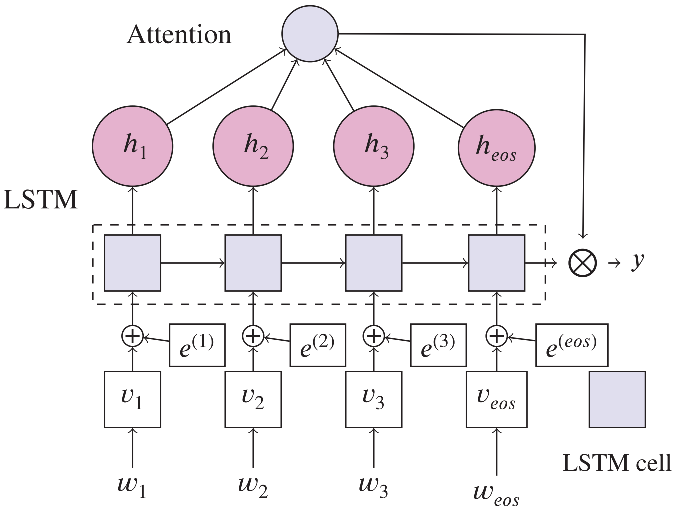


FIGURE 2 The built model for sentiment classification with perturbed word embeddings and attentive LSTM

As Equation (11) is computationally intractable for neural networks, Reference 12 proposed a fast gradient sign method (FGSM) to calculate the approximate perturbations, which is as follows.

$$\vec{e}_{adv} = \epsilon \frac{\vec{g}}{||\vec{g}||}, \quad \text{where } \vec{g} = \nabla_{\vec{x}} L(\vec{x}; \hat{\Theta}). \tag{13}$$

In Equation (13), $||\vec{g}||$ denotes the norm of gradients over all words in $\vec{x}$. As the perturbations generated by adversarial learning are composed of continuous values, we add the adversarial perturbations to the results of word embedding. Suppose that the word embedding vectors of all words in the input form a set E. The perturbations on E are computed as

$$\vec{e}_{\text{adv}} = \epsilon \frac{\vec{g}}{||\vec{g}||}, \quad \text{where } \vec{g} = \nabla_E L(E; \hat{\Theta}). \tag{14}$$

To obtain a model that is robust to adversarial perturbations, we define the loss function $\hat{L}(\Theta)$ as a combination of the classification loss L_{cls} (Equation (15)) and the adversarial loss L_{adv} (Equation (16)).

$$L_{\text{cls}} = L(E; \Theta), \tag{15}$$

$$L_{\text{adv}} = L(E + \vec{e}_{\text{adv}}; \Theta), \tag{16}$$

$$\hat{L}(\Theta) = \alpha L_{\text{cls}} + (1 - \alpha) L_{\text{adv}}, \tag{17}$$

where α is a parameter to control the ratio of classification loss and adversarial loss. The architecture of the model for sentiment classification with perturbed word embeddings and attentive LSTM is shown in Figure 2, where $e^{(i)}$ $(i = 1, 2, \dots)$ denotes the element in $\vec{e}_{\text{adv}}$. h_i $(i = 1, 2, \dots)$ is the hidden output of each LSTM cell. We name this model as *sentiment classification with adversarial learning* (SCAT for short).

3.3 The enhanced model in PU learning problem

As for PU learning problem, our solution is based on the method presented in Section 3.2. Compared with traditional sentiment classification, in PU learning problem, there is one extra step before conducting classification. The step is to distinguish the reviews with negative labels from the positive and unlabeled review texts. We adopt a two-step strategy to finish this task, following the suggestions in Reference 16. In detail, we use the Rocchio technique[45] to generate positive and potential negative review texts from unlabeled review texts. In Rocchio technique, each document is represented by a vector, and each element in the vector is the value that is computed with *tf-idf* (term frequency-inverse document frequency).

Let D denote the whole set of training texts, and let C_j denote the set of training reviews in class c_j. In this article, we have two classes, that is, j being 1 represents the positive review and j being -1 represents the negative review. To build a Rocchio classifier, a representative vector $\vec{c}_j$ of C_j is constructed for each class c_j first, which is as follows.

$$\vec{c}_j = \eta \frac{1}{|C_j|} \sum_{\vec{d} \in C_j} \frac{\vec{d}}{||\vec{d}||} - \rho \frac{1}{|D - C_j|} \sum_{\vec{d} \in D - C_j} \frac{\vec{d}}{||\vec{d}||}, \tag{18}$$

where η and ρ are parameters that control the weights of similar and dissimilar training examples. $\vec{d}$ denotes a piece of review, and $||\vec{d}||$ denotes the norm of review $\vec{d}$ (ie, the number of words in $\vec{d}$). Then for each test review td, the similarity of td with each representative vector is measured by cosine similarity. Finally, td is assigned to the class, the representative vector of which is the most similar to td. We use P to denote the positive set and U to denote the unlabeled set. The overall procedure of Rocchio technique is stated as follows.

1. Assign each review in P to class label 1;
2. Assign each review in U to class label -1;
3. Build a Rocchio classifier using P and U;
4. Use the classifier to classify U. Those reviews in U that are classified to be negative will form the negative reviews set.

Although the dataset is fully formed, the potential unreliability of the identified negative labeled texts increases the noise that is likely to harm the performance of neural networks. More specifically, the noise refers to the positive reviews, which are labeled to be negative by the Rocchio classifier. To tackle those noisy data in PU learning problem, we add a new random perturbation $\vec{r}$ to word embedding results E, due to the following two reasons.

1. The first is that adding a new random perturbation can help the gradient computation escape from the nonsmooth surrounding area of each word embedding.[33]
2. The second is that a random perturbation on word embeddings input can take the role of regularization, to defend the potential overfitting and obtain a better generalization performance.[46]

The added random perturbation in current word embedding results E generates new perturbed word embedding results E', which are as follows.

$$\vec{r} = \beta \times \text{sign}(\mathcal{N}(\mathbf{0}^k, \mathbf{I}^k)), \tag{19}$$

$$E' = E + \vec{r}, \tag{20}$$

$$\vec{e}_{\text{adv}} = \epsilon \frac{\vec{g}}{\|\vec{g}\|}, \quad \text{where } \vec{g} = \nabla_{E'} L(E'; \hat{\Theta}). \tag{21}$$

We choose Gaussian distribution to generate the random adversarial perturbation (Equation (19)). $\mathcal{N}(\mathbf{0}^k, \mathbf{I}^k)$ is the Gaussian distribution, where $\mathbf{0}^k$ is the mean vector and $\mathbf{I}^k$ is the covariance matrix (k is the dimension of word embedding). β controls the extent of trusting Gaussian distribution to generate the adversarial perturbation. sign$(\cdot)$ is the multidimensional indicator function, and the input of sign$(\cdot)$ is a k dimensional vector. Inspired by our model in traditional sentiment classification, the loss function in PU learning problem is constructed as follows.

$$L_{\text{cls}} = L(E'; \Theta), \tag{22}$$

$$L_{\text{adv}} = L(E' + \vec{e}_{\text{adv}}; \Theta), \tag{23}$$

$$\hat{L}(\Theta) = \alpha L_{\text{cls}} + (1 - \alpha) L_{\text{adv}}, \tag{24}$$

where $\vec{e}_{\text{adv}}$, Θ, and α have the same meaning as those in Equations (15), (16), and (17) (Section 3.2). We name this model as *PU learning problem with adversarial learning* (PUAT for short). Figure 3 demonstrates the model of PUAT, where $r^{(i)}$ $(i = 1, 2, \ldots)$ denotes the element in $\vec{r}$ and $e^{(i)}$ $(i = 1, 2, \ldots)$ denotes the element in $\vec{e}_{\text{adv}}$. h_i $(i = 1, 2, \ldots)$ is the hidden output of each LSTM cell.

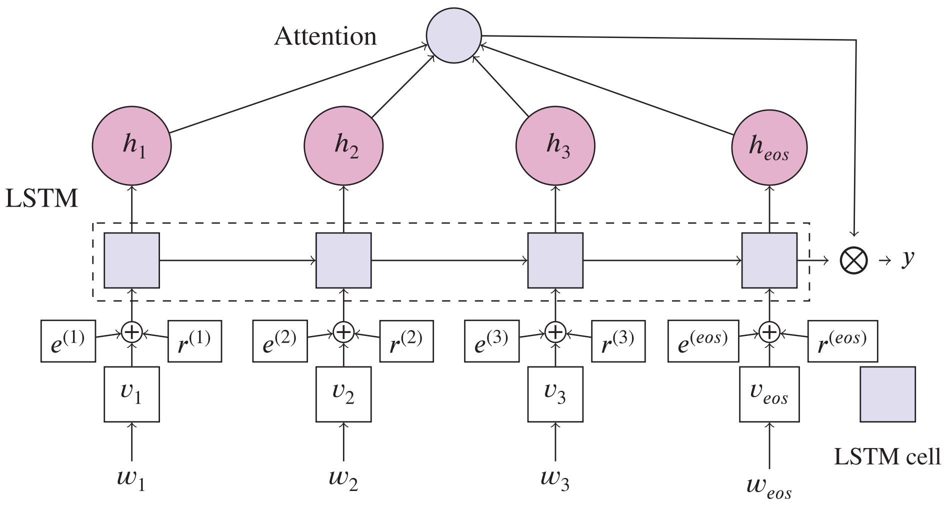


FIGURE 3 The PUAT model with perturbed and random word embeddings and attentive LSTM

TABLE 1 Statistics of IMDB dataset, Elec dataset, and SST-2 dataset

Dataset	#label	#training	#test	Avg.	Max
IMDB dataset	2	25 000	25 000	239	2506
Elec dataset	2	25 000	25 000	107	4983
SST-2 dataset	2	6920	1821	19	54

4 | EXPERIMENT AND EVALUATION

4.1 | Experimental setting

Datasets.We evaluated our methods in three real-world datasets, that is, IMDB dataset, Elec dataset ,and SST-2 dataset (SST is short for Stanford sentiment treebank)[1]. The IMDB dataset contains movie reviews and has been used in evaluation of sentiment classification by many researchers.[5,47] The Elec dataset contains electronic product reviews collected from Amazon and has been also widely used in sentiment classification tasks.[48] There are two sentiment classes in the two datasets, including positive class and negative class. For IMDB dataset and Elec dataset, we randomly selected 90% reviews from training reviews set to form the training set and the remained 10% reviews form the validation set. SST-2 dataset contains movie reviews and is also used in binary sentiment classification task.[49] The statistics of the three datasets are shown in Table 1.

In Table 1, #label denotes the number of classes, and there are two sentiment classes in the three datasets, including positive class and negative class. #training and #test represent the number of reviews in the training set and test set. Avg. is the average length of reviews in each dataset, and Max denotes the maximum length of the reviews. For IMDB dataset and Elec dataset, we randomly selected 90% reviews from training reviews set to form the training set and the remained 10% reviews form the validation set. For SST-2 dataset, we utilized the predefined validation set for validation. From Table 1, it can be observed that we evaluate our methods in two types of reviews datasets, from the aspect of average review length. In the first type of datasets, we evaluate methods on long reviews (IMDB dataset and Elec dataset). For IMDB dataset, the average review length is 239, and for Elec dataset, the average review length is 107. In the second type of datasets, we evaluate methods on short reviews (SST-2 dataset). For SST-2 dataset, the average review length is 19.

Implementation. We implemented our codes in TensorFlow.[50] We compare our models to the following models that can be used to solve the classification problem on review texts, including

1. SVM and Naïve Bayes. SVM and Naïve Bayes are two classic classification methods that are also commonly used in text mining and natural language processing applications. Reference 22 showed that the two methods can be used in sentiment classification problem. We utilize commonly used *tf-idf* (term frequency-inverse document frequency) as features for both models.
2. Long short-term memory (LSTM) and gated recurrent unit (GRU). LSTM and GRU are two popular variants of RNN and have been also applied on sentiment classification tasks. We

[1]https://nlp.stanford.edu/sentiment/index.html

TABLE 2 Test accuracy in IMDB dataset, Elec dataset, and SST-2 dataset

	Test accuracy		
Method	**IMDB dataset**	**Elec dataset**	**SST-2 dataset**
Naïve Bayes	0.857	0.834	0.755
SVM	0.847	0.861	0.785
LSTM	0.906	0.874	0.790
GRU	0.855	0.877	0.805
Attentive LSTM	0.911	0.878	0.808
Adversarial LSTM	0.917	0.886	0.810
SCAT	**0.936**	**0.896**	**0.818**

built an LSTM network with the hidden size being 128. The parameters and configuration of GRU network are the same as those in LSTM.

3. Attention-based LSTM or attentive LSTM. This model is proposed in Reference 51, as a hierarchical attention-based LSTM network, and achieves good performance on a series of text classification tasks.
4. Adversarial LSTM. This model is proposed in Reference 8, which adapts adversarial learning in basic LSTM network and achieves good performance on semisupervised classification problem.

We have implemented and run all compared models. Regarding the preprocessing, the words whose document frequencies are less than 2 are removed from the reviews in both datasets. The reason is that those words that less frequently appear will enlarge the whole vocabulary size and further obviously increase training time.[52] In our methods (SCAT and PUAT), the LSTM network is configured with 128 hidden units. The number of hidden units of our methods is explored and set in validation set.

Parameter setting. The parameters are set based on the evaluation results on the validation set. The word embeddings are initialized by GloVe,[42] and the dimension of word embedding vector k is 200. The parameter α in Equations (17) and (24) is set to 0.5. The parameter ϵ in Equations (14) and (21) is set to 1.0. We will report the experimental results of parameters sensitivity in Section 5, and discuss the selection of α and ϵ.

For the optimization of model parameters, we used Adam optimizer.[53] Based on the results on validation set, we set the learning rate to 0.001, batch size to 256, and dropout rate to 0.8. To provide a fair comparison, all neural networks are set with similar model capacity. In detail, the number of hidden units in neural networks is 128, and the word embedding size is 200.

4.2 | Evaluation in traditional sentiment classification

Table 2 shows the test performances of our method SCAT and the compared methods in IMDB dataset, Elec dataset, and SST-2 dataset. It can be seen that our method SCAT achieves the highest test accuracy in all datasets.

On the one hand, SCAT performs better than the classic methods. Take SVM as an example. SVM achieves 0.847 test accuracy in IMDB dataset, 0.861 test accuracy in Elec dataset, and 0.785

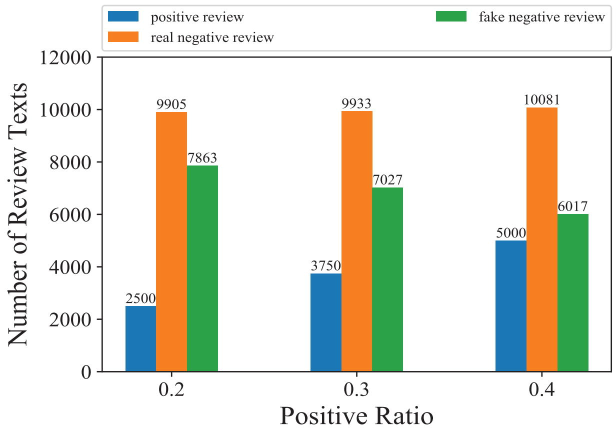


FIGURE 4 The number of review texts of the three different review types in IMDB dataset. The bars from left to right represent the positive review, real negative review, and fake negative review, respectively.

test accuracy in SST-2 dataset. By contrast, SCAT achieves a higher test accuracy of 0.936 in IMDB dataset, 0.896 in Elec dataset, and 0.818 in SST-2 dataset. On the other hand, SCAT outperforms the neural network-based methods. Compared with adversarial LSTM, which achieves a 0.917 test accuracy in IMDB dataset, 0.886 test accuracy in Elec dataset, and 0.810 test accuracy in SST-2 dataset, our method also achieves better results, which are 0.936, 0.896, and 0.818. In all datasets, the improvements achieved by SCAT are significant over baseline models based on paired t-test ($P < .005$).

4.3 | Evaluation in PU learning problem

4.3.1 | The generation of training sets in PU learning problem

Similar to the article,[16] we take the following steps to generate PU learning problem dataset from the training set. We randomly select p percent of positive reviews as the positive set P, and the remaining positive reviews and negative reviews are disassociated with their labels and are used to form the unlabeled set U. The task is to classify negative reviews from unlabeled set U. We change the value of p in the range from 20% to 40% to provide a comprehensive evaluation.

As stated in Section 3.3, we used *tf-idf* to compute the weight of each word and further to form feature vectors. We then built a Rocchio classifier on the positive set P and unlabeled set U. The reviews in set U that are classified to be negative form the negative set N. Figures 4 (IMDB dataset) and 5 (Elec dataset) show the details of negative reviews generated by Rocchio classifier with different positive ratios (0.2, 0.3, and 0.4). It can be seen that the whole training set is divided into positive set and negative set. The negative set consists of two parts, including real negative review texts and fake negative review texts. The fake negative review texts are the original positive reviews but misclassified as negative reviews by Rocchio classifier. Also, it can be found that along with the positive ratio increasing (0.2 to 0.4), the proportion of fake negative review texts decreases.

4.3.2 | Experimental results in PU learning problem

We conduct the experiments under three cases of positive ratios (0.2, 0.3, and 0.4), which correspond to the positive ratio settings in Section 4.3.1. We will report the test performance of each

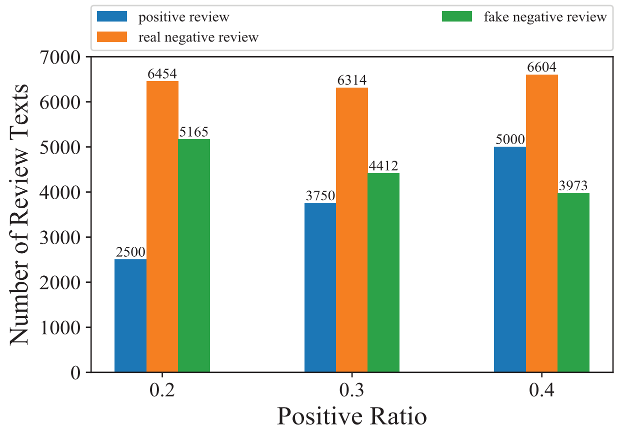


FIGURE 5 The number of review texts of the three different review types in Elec dataset. The bars from left to right represent the positive review, real negative review, and fake negative review, respectively

TABLE 3 Test performance in IMDB dataset and Elec dataset in PU learning problem with positive ratio being 0.2

	IMDB dataset			**Elec dataset**		
Method	**F1-score**	**Recall**	**Test accuracy**	**F1-score**	**Recall**	**Test accuracy**
Naïve Bayes	0.159	0.086	0.509	0.608	0.480	0.595
SVM	0.341	0.207	0.599	0.777	0.706	0.797
GRU	0.526	0.368	0.655	0.760	0.673	0.783
LSTM	0.512	0.354	0.647	0.772	0.706	0.779
Attentive LSTM	0.609	0.458	0.695	0.775	0.721	0.786
Adversarial LSTM	0.530	0.370	0.665	0.799	0.739	0.808
SCAT	0.513	0.353	0.652	0.804	0.767	0.810
PUAT	**0.650**	**0.502**	**0.723**	**0.804**	**0.772**	**0.814**

method. The evaluation metrics include F1-score, recall, and test accuracy. F1-score is a widely used metric in classification problems and tends to give an integrated evaluation, as F1-score combines recall and precision. The reported F1-score is computed on positive class, as in PU learning problem, there are only positive labeled reviews. F1-score is computed as

$$\text{F1-score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}. \tag{25}$$

To give a comprehensive evaluation, we also report the results of recall and test accuracy. Furthermore, we will discuss the relationship between the positive ratio and test performance. In the generation of training sets, we have generated positive labeled sets P and negative labeled sets N under different cases of positive ratios (0.2, 0.3, and 0.4). Different positive ratios decide different proportions of positive reviews and negative reviews in final training set.

The parameter β in Equation (19) (Section 3.3) is set to 1.0 in all experiments. We will discuss the selection of β in following Section 5. The experimental results are shown in the following Tables 3-6, and 7.

TABLE 4 Test performance in IMDB dataset and Elec dataset in PU learning problem with positive ratio being 0.3

	IMDB dataset			**Elec dataset**		
Method	**F1-score**	**Recall**	**Test accuracy**	**F1-score**	**Recall**	**Test accuracy**
Naïve Bayes	0.280	0.164	0.527	0.704	0.626	0.670
SVM	0.509	0.347	0.666	0.824	0.807	0.824
GRU	0.676	0.531	0.742	0.815	0.796	0.823
LSTM	0.683	0.539	0.747	0.820	0.805	0.825
Attentive LSTM	0.719	0.588	0.772	0.815	0.806	0.822
Adversarial LSTM	0.689	0.544	0.764	0.819	0.779	0.828
SCAT	0.708	0.569	0.778	0.822	0.791	0.832
PUAT	**0.815**	**0.751**	**0.832**	**0.825**	**0.808**	**0.839**

TABLE 5 Test performance in IMDB dataset and Elec dataset in PU learning problem with positive ratio being 0.4

	IMDB dataset			**Elec dataset**		
Method	**F1-score**	**Recall**	**Test accuracy**	**F1-score**	**Recall**	**Test accuracy**
Naïve Bayes	0.421	0.270	0.565	0.732	0.686	0.696
SVM	0.626	0.464	0.723	0.829	**0.831**	0.828
GRU	0.752	0.639	0.788	0.821	0.813	0.830
LSTM	0.774	0.675	0.792	0.825	0.803	0.832
Attentive LSTM	0.775	0.681	0.793	0.831	0.823	0.839
Adversarial LSTM	0.799	0.716	0.813	0.830	0.819	0.838
SCAT	0.805	0.732	0.822	0.834	0.825	0.842
PUAT	**0.818**	**0.749**	**0.824**	**0.835**	**0.831**	**0.845**

Tables 3,4, and 5 present F1-score, recall, and test accuracy results of all methods in PU learning problem. Table 6 presents the results of area under curve (AUC) estimator of different models in different positive ratio settings. The AUC estimator is computed according to the standard computation formula of AUC.[54] It can be found that the proposed PUAT method achieves the highest F1-score, recall, test accuracy, and AUC values in all three positive ratio settings (0.2, 0.3, and 0.4). Furthermore, we can have following observations.

1. First, in all three positive ratio settings, the proposed PUAT method achieves better F1-score and test accuracy results than the traditional classifiers, including SVM and Naïve Bayes. Take SVM as an example. In the case of positive ratio being 0.2, SVM achieves a 0.341 F1-score and a 0.599 test accuracy in IMDB dataset, and a 0.777 F1-score and a 0.797 test accuracy in Elec dataset. By contrast, PUAT achieves a higher performance of a 0.650 F1-score and a 0.723 test accuracy in IMDB dataset, and a 0.804 F1-score and a 0.814 test accuracy in Elec dataset, also in the case of positive ratio being 0.2.

TABLE 6 Results of AUC estimator in IMDB dataset and Elec dataset with different positive ratios

		AUC estimator	
Positive ratio	**Method**	**IMDB dataset**	**Elec dataset**
0.2	Naïve Bayes	0.743	0.732
	SVM	0.756	0.807
	LSTM	0.877	0.857
	GRU	0.874	0.858
	Attentive LSTM	0.889	0.864
	Adversarial LSTM	0.878	0.865
	SCAT	0.903	0.877
	PUAT	**0.911**	**0.888**
0.3	Naïve Bayes	0.752	0.748
	SVM	0.780	0.824
	LSTM	0.897	0.873
	GRU	0.891	0.879
	Attentive LSTM	0.904	0.887
	Adversarial LSTM	0.896	0.889
	SCAT	0.916	0.895
	PUAT	**0.922**	**0.899**
0.4	Naïve Bayes	0.765	0.753
	SVM	0.804	0.827
	LSTM	0.910	0.872
	GRU	0.911	0.877
	Attentive LSTM	0.916	0.882
	Adversarial LSTM	0.910	0.884
	SCAT	0.924	0.896
	PUAT	**0.932**	**0.907**

2. PUAT also outperforms the neural network-based methods. For example, in the case of positive ratio being 0.3, adversarial LSTM achieves a 0.689 F1-score and a 0.764 test accuracy in IMDB dataset and achieves a 0.819 F1-score and a 0.828 test accuracy in Elec dataset. By contrast, the proposed PUAT achieves a superior performance of a 0.815 F1-score and a 0.832 test accuracy in IMDB dataset, along with a 0.825 F1-score and a 0.839 test accuracy in Elec dataset.
3. Compared with the performance achieved by LSTM and GRU, the F1-score results of SVM are competitive, especially in Elec dataset. That is, the F1-score results of SVM are close to those of LSTM and GRU or even better than those of LSTM and GRU.

   An important reason is in the existence of misclassified reviews (noise), that is, the true positive review texts that are misclassified to be negative. The LSTM network is easy to suffer

		Metric	
Positive Ratio	**Method**	**F1-score**	**Test accuracy**
0.2	Naïve Bayes	0.275	0.542
	SVM	0.442	0.622
	LSTM	0.481	0.628
	GRU	0.480	0.626
	Attentive LSTM	0.501	0.642
	Adversarial LSTM	0.512	0.639
	SCAT	0.536	0.657
	PUAT	**0.560**	**0.658**
0.3	Naïve Bayes	0.447	0.603
	SVM	0.602	0.683
	LSTM	0.583	0.678
	GRU	0.590	0.679
	Attentive LSTM	0.540	0.656
	Adversarial LSTM	0.622	0.704
	SCAT	0.628	0.690
	PUAT	**0.645**	**0.705**
0.4	Naïve Bayes	0.588	0.663
	SVM	0.718	0.729
	LSTM	0.638	0.699
	GRU	0.650	0.705
	Attentive LSTM	0.660	0.716
	Adversarial LSTM	0.704	0.741
	SCAT	0.714	0.737
	PUAT	**0.758**	**0.758**

TABLE 7 Results of F1-score and test accuracy in SST-2 dataset with different positive ratios

from overfitting on such kind of noise. As for SVM, the positive set P and negative set U have been generated by Rocchio classifier, and such data separation provides a preliminary preparation for SVM to find a strong margin to separate positive and negative classes. The noise also leads to bad performances of Naïve Bayes, and Naïve Bayes tends to predict all reviews in test set to be negative.

4. For the analysis of recall, let us start from the computation of recall, which is given by

$$\text{recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \tag{26}$$

where TP (short for true positive) denotes the number of positive reviews that are correctly predicted as positive labeled reviews. FN (short for false negative) denotes the number of positive reviews that are misclassified to be negative reviews. In our model, adversarial learning improves the ability of distinguishing true negative reviews from fake negative reviews, which decreases the number of fake negative reviews.

5. In addition, we can make some interesting observations on the positive ratio p. When p is equal to 1.0, PU learning problem turns to a case that all positive labeled reviews in training set are reserved. When p is near to 0, it indicates that there are few positive review texts, and it will be difficult to build an effective classifier purely using unlabeled review texts. From Tables 3,4, and 5, it can be found that there is a positive correlation between the evaluation metrics and positive ratio p. The reason is that the increasing number of positive review texts can help the classifier learn more useful features.
6. From Tables 3,4, and 5, it can be also found that the overall performance of SCAT and PUAT in Elec dataset is superior to that in IMDB dataset. One observation can be made from Figures 4 and 5, that is, the data imbalance problem is more serious in IMDB dataset than that in Elec dataset. Take the dataset case of positive ratio $p = 0.2$, for example, with the same number of positive reviews (2500), IMDB dataset has 17 768 negative reviews, while Elec dataset only has 11 619 negative reviews. With the increase of positive ratio p (Tables 4 and 5), the data imbalance problem eases in IMDB dataset, and the performance in IMDB dataset is also closer to that in Elec dataset.

4.4 | Performance analysis for word embedding

Through experiments, we found that the improvement in word embedding is another reason for performance improvement brought by adversarial learning. Such improvement in word embedding is particularly important for those sentimental words, such as "good," "bad," "nice," and "cool." In this section, we evaluate the results of word embedding by computing the distance between two sentimental words. The basic motivation is that for sentimental words, the two words that have similar semantic meaning (ie, synonyms) should have a close distance in word embedding space. By contrast, the two words that have opposite meaning (ie, antonyms) should have a far distance in word embedding space.

We compute the cosine distance and Euclidean distance of word embedding results of sentimental words, including antonym pairs and synonym pairs. Cosine distance d_{cosine} and Euclidean distance $d_{\text{Euclidean}}$ are computed as follows.

$$d_{\text{cosine}}(v_i, v_j) = 1 - \frac{v_i \cdot v_j}{\|v_i\| \|v_j\|}, \tag{27}$$

$$d_{\text{Euclidean}}(v_i, v_j) = \sqrt{\sum_{l=1}^{k} (v_{i,l} - v_{j,l})^2}, \tag{28}$$

where v_i and v_j are k-dimensional word embedding vectors of two sentimental words w_i and w_j. $v_{i,l}$ and $v_{j,l}$ represent the lth element in v_i and v_j, respectively. We report the comparison results of several antonym pairs and synonym pairs in Table 8.

For antonyms, the word embedding distances are supposed to be larger, so that the classifier can distinguish them more precisely. It can be found that for antonym pairs, the cosine distances and Euclidean distances in SCAT model are clearly larger than those in LSTM and attentive

TABLE 8 Distance comparison of several typical antonyms pairs and synonyms pairs in word embedding space

	Cosine distance			Euclidean distance		
Antonym pairs	**LSTM**	**Attn-LSTM**	**SCAT**	**LSTM**	**Attn-LSTM**	**SCAT**
<good, bad>	0.7042	0.7085	**0.7568**	7.6724	7.4241	**8.1332**
<better, worse>	0.6471	0.7587	**0.8994**	7.7090	8.3631	**8.5263**
<great, awful>	0.8076	0.8095	**1.1631**	8.7912	9.1091	**11.4326**
Synonym pairs	LSTM	Attn-LSTM	SCAT	LSTM	Attn-LSTM	SCAT
<bad, poor>	0.7986	0.6590	**0.5979**	7.6404	7.5974	**7.5695**
<nice, cool>	0.9851	0.5367	**0.4156**	11.4963	6.5373	**6.1398**
<nice, awesome>	0.9027	0.8706	**0.7295**	9.7257	7.9293	**7.6517**

Abbreviation: Attn-LSTM, attentive LSTM.

LSTM. Take the antonym pair <good, bad> as an example. "good" and "bad" have a 0.7042 cosine distance and a 7.6724 Euclidean distance in LSTM, but have a larger cosine distance (0.7568) and a larger Euclidean distance (8.1332) in SCAT. It means that the word embeddings generated in SCAT can separate antonyms more effectively. For synonyms, the distances between two word embeddings are supposed to be smaller. From Table 8, it can be also seen that the cosine distances and Euclidean distances in SCAT are smaller than those in LSTM and attentive LSTM. Take the synonym pair <nice, cool> as an example. Compared with LSTM and attentive LSTM, our model SCAT achieves closer distances with a 0.4156 cosine distance and a 6.1398 Euclidean distance. The results indicate that adversarial learning improves the quality of word embedding, and such improvement further contributes to the final improvement of classification performance.

4.5 | Training time comparison

Training time is another concern when people deploy neural network models. In this section, we compare the training time of different models in IMDB dataset and Elec dataset. As neural network models are usually trained by iterations of epochs, we report the average training time of one epoch when positive ratio p is 0.4. All experiments were conducted on a machine equipped with a Xeon E5-2680 v4 CPU and a single NVIDIA Telsa M40 GPU. The compared results are presented in Table 9. The training time of Naïve Bayes and SVM is also provided for comprehensive comparison. However, as there is no epoch in training procedures of Naïve Bayes and SVM, we report the training time of Naïve Bayes and SVM.

From Table 9, it can be found that traditional models, including Naïve Bayes and SVM, run faster than neural network models, such as LSTM and attentive LSTM. We can also observe that the models using adversarial learning techniques (adversarial LSTM, SCAT, and PUAT) require more training time. The reason is that the models using adversarial learning techniques require to compute the adversarial perturbations. As for our models, SCAT needs to compute the gradients on E (the set of word embedding vectors) in Equation (14), and PUAT needs to compute the gradients also on E in Equation (21).

Furthermore, it can be seen that the average training time of our models is slightly higher than that of adversarial LSTM model. Note that our models achieve better performances, which

TABLE 9 Training time comparison with positive ratio *p* being 0.4

	Average training time (of one epoch)	
Model	**IMDB dataset (s)**	**Elec dataset (s)**
Naïve Bayes	4.71	1.57
SVM	5.13	1.99
GRU	7.87	6.82
LSTM	7.11	6.32
Attentive LSTM	7.28	6.43
Adversarial LSTM	16.85	15.06
SCAT	16.90	15.15
PUAT	19.38	17.05

have been verified by the experimental results in Sections 4.2,4.3, and 4.4. Take PUAT as an example. Compared with adversarial LSTM in IMDB dataset, PUAT achieves higher F1-score, recall, and test accuracy (see Table 5), while PUAT only spends 2.53 seconds more in one epoch (19.38 seconds for PUAT and 16.85 seconds for adversarial LSTM). The results indicate that considering the superior performances, the training time of our models is acceptable.

5 | PARAMETERS ANALYSIS

In this section, we conduct experiments to analyze the sensitivity of our models to parameters and explore for more inspirations.

5.1 | Impact of loss weight α

The parameter α in Equations (17) and (24) provides a handler for controlling the weights of classification loss and adversarial loss in loss functions. When α is equal to 1, the loss degenerates into the traditional classification loss, and when α is equal to 0, there is only adversarial loss in the loss function. We take the traditional sentiment classification to investigate the sensitivity of our model SCAT to α. We conducted experiments in the value range from 0.0 to 1.0, and the results are shown in following Figure 6.

It can be seen that the test accuracy tends to be stable after α is equal to or larger than 0.5 and achieves the highest value at 0.5. When α is equal to 0, test accuracy is low because the loss function is degraded as pure adversarial loss. When α is equal to 1, the built model degenerates to pure attentive LSTM, which only produces a suboptimal performance. Such a change trend indicates that the classification loss and adversarial loss both take important roles in the loss function. Furthermore, it indicates that the built adversarial loss is indispensable for achieving high classification accuracy. So, in this article, the default value of α in all experiments is set to 0.5.

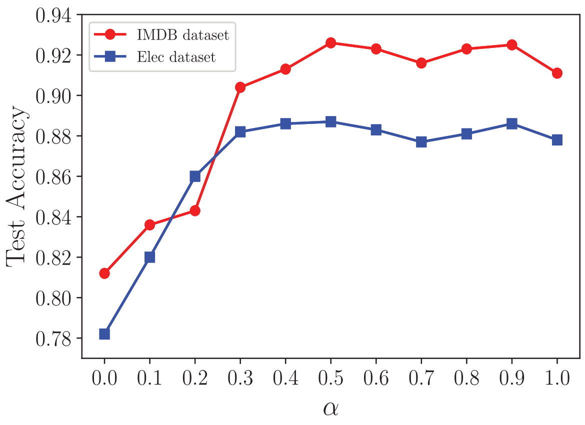


FIGURE 6 Test accuracy varies with α in traditional sentiment classification

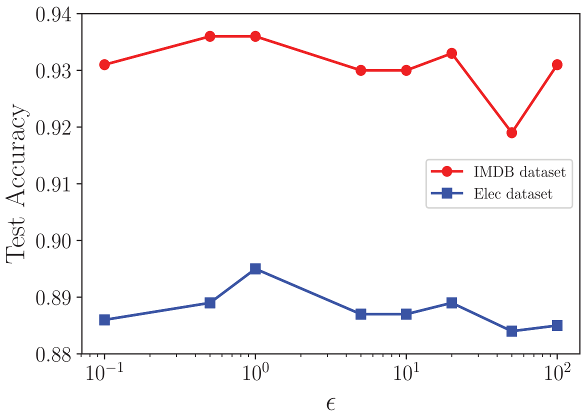


FIGURE 7 Test accuracy varies with ϵ in traditional sentiment classification

5.2 | Impact of ϵ

The parameter ϵ controls the threshold of selecting $\vec{e}_{\text{adv}}$. We take the traditional sentiment classification to investigate the sensitivity of our model SCAT to ϵ. In Equation (12), when ϵ is equal to 0, the adversarial loss degenerates to the ordinary loss. If ϵ is large or comparable to the dimension of word embedding, the nearest neighbor of a word w is likely to be changed to a large extent with the perturbation, and further the content of a review will be probably modified implicitly.[36] The experimental results of the sensitivity to ϵ is shown in following Figure 7. The horizontal axis is set as the logarithmic coordinate ranging from 10^{-1} to 10^{2}.

It can be observed that the test accuracy reaches the highest value at the point of ϵ around 10^{0} (ie, 1.0). Note that at the point around 10^{0} (ie, 1.0), ϵ is far smaller than the dimension of word embedding k (in our experiments, k is equal to 200). In this case, the added perturbation does not change the fundamental sentence semantic or sentence structure in a review. So, in this article, the default value of ϵ in all experiments is set to 1.0.

5.3 | Impact of β

The parameter β controls the trust extent of the adversarial perturbation generated from Gaussian distribution (see Equation (19) in Section 3.3). If β is equal to 0, the proposed PUAT

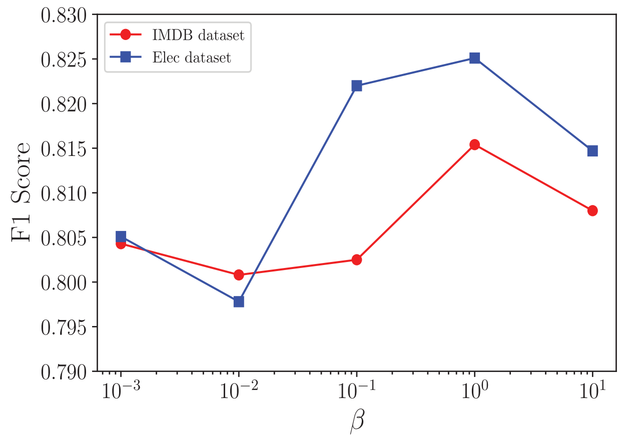


FIGURE 8 F1-score varies with β when the positive ratio is equal to 0.3

model degenerates into the ordinary adversarial learning method. If β is a large value, the random perturbations may exert too much impact on word embeddings, which probably further harms the result of word embedding. We study the sensitivity of PUAT model to β, and take the case that the positive ratio is equal to 0.3 as an example. The experimental results are shown in Figure 8, where the horizontal axis is set as the logarithmic coordinate ranging from 10^{-3} to 10^{1}.

It can be found that the optimal value of β is achieved around the value of 10^{0} (ie, 1.0) in both datasets. The change trend of β in IMDB dataset is smoother than the change trend in Elec dataset. These observations mean that the value of β should be set not too large or too small. In our experiments, β is set to 1.0 in all experiments.

6 | CONCLUSION

In this article, we give a comprehensive study of adversarial learning in sentiment classification. We solve two typical sentiment classification problems, which are traditional sentiment classification and PU learning problem, under one framework that is built based on adversarial learning, attention mechanism, and LSTM network. The framework contains two models: sentiment classification with adversarial learning (SCAT) and PU learning problem with adversarial learning (PUAT). There are two key ideas in our framework. One is to utilize the attention mechanism that is expected to give more weights on sentimental words, and the other is to utilize adversarial learning that is expected to improve the classification performance.

In two datasets, the experimental results demonstrate that our models, PUAT and SCAT, achieve superior performance than the compared models. Such superiority verifies the effectiveness of the proposed way of using attention mechanism and adversarial learning in our models. Adversarial learning brought our models more robustness by improving the quality of word embedding, especially for sentimental words. We gave a detailed discussion on experimental results. We also provided a comparison of training time of different models and studied the sensitivity of our models to parameters.

In future, on the one hand, we plan to investigate some other attention mechanisms in neural network for the task of sentiment classification, such as hierarchical attention mechanism.[55] On the other hand, we will study other types of sentiment classification problems, such as aspect-level sentiment classification.[56]

ACKNOWLEDGMENT
This article is supported by National Natural Science Fund of China (No. 61702391) and Natural Science Foundation of Shaanxi province (No. 2018JQ6050).

ORCID
Yueshen Xu https://orcid.org/0000-0001-7210-0543

REFERENCES

1. Pang B, Lee L. Opinion mining and sentiment analysis. *Found Trends Inf Retriev*. 2007;2(1-2):1-135.
2. Wang Y, Huang M, Zhao L, Zhu X. Attention-based LSTM for aspect-level sentiment classification. Paper presented at: Proceedings of International Conference on Empirical Methods in Natural Language Processing (EMNLP); 2016.
3. Liu B. *Sentiment Analysis: Mining Opinions, Sentiments, and Emotions*. Cambridge, MA: Cambridge University Press; 2015.
4. Tang D, Qin B, Liu T. Document modeling with gated recurrent neural network for sentiment classification. Paper presented at: Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP); 2015:1422-1432.
5. Yuan Z, Wu S, Wu F, Liu J, Huang Y. Domain attention model for multi-domain sentiment classification. *Knowl Based Syst*. 2018;155:1-10.
6. Phienthrakul T, Kijsirikul B, Takamura H, Okumura M. Sentiment classification with support vector machines and multiple kernel functions. Paper presented at: Proceedings of International Conference on Neural Information Processing; 2009:583-592.
7. Dey L, Chakraborty S, Biswas A, Bose B, Tiwari S. Sentiment analysis of review datasets using Naïve Bayes and K-NN classifier. *Int J Inf Eng Electr Bus*. 2016;4:54-62.
8. Miyato T, Dai AM, Goodfellow I. Adversarial training methods for semi-supervised text classification. Paper presented at: Proceedings of the International Conference on Learning Representations (ICLR); 2017.
9. Zhang Y, Gan Z, Carin L. Generating text via adversarial training. Paper presented at: Proceedings of the Workshop on Adversarial Training in Neural Information Processing Systems Conference (Workshop in NIPS); 2016:1-6.
10. Zhang Y, Gan Z, Fan K, et al. Adversarial feature matching for text generation. Paper presented at: Proceedings of International Conference on Machine Learning (ICML); 2017:4006-4015.
11. Shrivastava A, Pfister T, Tuzel O, Susskind J, Wang W, Webb R. Learning from simulated and unsupervised images through adversarial training. Paper presented at: Proceedings of CVPR; 2017:2242-2251.
12. Goodfellow Ian J, Shlens Jonathon, Szegedy Christian. Explaining and harnessing adversarial examples. Paper presented at: Proceedings of the International Conference on Learning Representations (ICLR); 2015:1-11.
13. Szegedy C, Zaremba W, Sutskever I, et al. Intriguing properties of neural networks. Paper presented at: Proceedings of International Conference on Learning Representations (ICLR); 2014.
14. Papernot N, McDaniel P, Jha S, Fredrikson M, Celik ZB, Swami A. The limitations of deep learning in adversarial settings. Paper presented at: IEEE European Symposium on Security and Privacy (EuroS&P); 2016:372-387.
15. Zhang Y, Xu J, Yang P, Sun X Learning sentiment memories for sentiment modification without parallel data. Paper presented at: Proceedings of International Conference on Empirical Methods in Natural Language Processing (EMNLP); 2018:1103-1108.
16. Liu B, Dai Y, Li X, Lee WS, Yu PS. Building text classifiers using positive and unlabeled examples. Paper presented at: Proceedings of the 3rd IEEE International Conference on Data Mining (ICDM); 2003:179-186.
17. Li H, Liu B, Mukherjee A, Shao J. Spotting fake reviews using positive-unlabeled learning. *Computacióny Sistemas*. 2014;18(3):467-475.
18. Ren Y, Ji D, Zhang H. Positive unlabeled learning for deceptive reviews detection. Paper presented at: Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP); 2014:488-498.
19. Lin J, Mao W, Zeng DD. Personality-based refinement for sentiment classification in microblog. *Knowl Based Syst*. 2017;132:204-214.

20. Yin Y, Chen L, Xu Y, Wan J, Zhang H, Mai Z. QoS prediction for service recommendation with deep feature learning in edge computing environment. *Mob Netw Appl*. 2019;25(2):391-401.
21. Kuang L, Gong T, OuYang S, Gao H, Deng S. Offloading decision methods for multiple users with structured tasks in edge computing for smart cities. *Future Generat Comput Syst*. 2020;105:717-729.
22. Pang B, Lee L, Vaithyanathan S. Thumbs up? sentiment classification using machine learning techniques. Paper presented at: Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP); 2002:79-86.
23. Zhang L, Wang S, Liu B. Deep learning for sentiment analysis: a survey. *Interdiscip Rev Data Mining Knowl Discov*. 2018;8(4):1253.
24. Kim Y. Convolutional neural networks for sentence classification. Paper presented at: Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP); 2014:1746-1751.
25. Dos SC, Gatti M. Deep Convolutional neural networks for sentiment analysis of short texts. Paper presented at: Proceedings of International Conference on Computational Linguistics (COLING): Technical Papers; 2014:69-78.
26. Salinca A. Convolutional neural networks for sentiment classification on business reviews. Paper presented at: Proceedings of Workshop on Semantic Machine Learning in IJCAI; 2017:45-49.
27. Yu J, Kuang Z, Zhang B, Zhang W, Lin D, Fan J. Leveraging content sensitiveness and user trustworthiness to recommend fine-grained privacy settings for social image sharing. *IEEE Trans Inform Forens Sec*. 2018;13(5):1317-1332.
28. Gao H, Duan Y, Shao L, Sun X. Transformation-based processing of typed resources for multimedia sources in the IoT environment. *Wirel Netw*. 2019.
29. Wang Y, Huang M, Zhao L. Attention-based lstm for aspect-level sentiment classification. Paper presented at: Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP); 2016:606-615.
30. Zhou X, Wan X, Xiao J. Attention-based LSTM network for cross-lingual sentiment classification. Paper presented at: Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP); 2016:247-256.
31. Li XL, Liu B. Learning from positive and unlabeled examples with different data distributions. Paper presented at: Proceedings of the European Conference on Machine Learning (ECML); 2005:218-229.
32. Du Plessis MC, Niu G, Sugiyama M. Analysis of learning from positive and unlabeled data. *Adv Neural Inf Process Syst (NIPS)*. 2014;703-711.
33. Kurakin A, Boneh D, Tramèr F, Goodfellow I, Papernot N, McDaniel P. Ensemble adversarial training: attacks and defenses. Paper presented at: Proceedings of the International Conference on Learning Representations (ICLR); 2018.
34. Goodfellow I, Pouget-Abadie J, Mirza M, et al. Generative adversarial nets. *Adv Neural Inf Process Syst (NIPS)*. 2014;2672-2680.
35. Han J, Zhang Z, Cummins N, Schuller B. Adversarial training in affective computing and sentiment analysis: recent advances and perspectives. arXiv preprint arXiv:1809.08927; 2018.
36. Wu Y, Bamman D, Russell S. Adversarial training for relation extraction. Paper presented: Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP); 2017:1778-1783.
37. Sachan DS, Zaheer M, Salakhutdinov R. Revisiting LSTM networks for semi-supervised text classification via mixed objective function. Paper presented at: Proceedings of ACM Conference on Knowledge Discovery and Data Mining (KDD); 2018.
38. Ganin Y, Ustinova E, Ajakan H, et al. Domain-adversarial training of neural networks. *J Mach Learn Res (JMLR)*. 2016;17:59:1-59:35.
39. Jia R, Liang P. Adversarial examples for evaluating reading comprehension systems. Paper presented at: Proceedings of International Conference on Empirical Methods in Natural Language Processing (EMNLP); 2017:2021-2031.
40. Liang B, Li H, Su M, Bian P, Li X, Shi W. Deep text classification can be fooled. Paper presented at: Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI); 2018:4208-4215.
41. Mikolov T, Sutskever I, Chen K, Corrado GS, Dean J. Distributed representations of words and phrases and their compositionality. *Adv Neural Inf Process Syst (NIPS)*. 2013;3111-3119.

42. Pennington J, Socher R, Manning C. Glove: global vectors for word representation. Paper presented at: Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP); 2014:1532-1543.
43. Hochreiter S, Schmidhuber J. Long short-term memory. *Neural Comput*. 1997;9(8):1735-1780.
44. Bahdanau D, Cho K, Bengio Y. Neural machine translation by jointly learning to align and translate. Paper presented at: Proceedings of the International Conference on Learning Representations (ICLR); 2015.
45. Manning CD, Raghavan P, Schütze H. *Introduction to Information Retrieval*. Cambridge, MA: Cambridge University Press; 2009.
46. Bishop CM. Training with noise is equivalent to Tikhonov regularization. *Neural Comput*. 1995;7(1):108-116.
47. Maas AL, Daly RE, Pham PT, Huang D, Ng AY, Potts C. Learning word vectors for sentiment analysis. Paper presented at: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL); 2011:142-150.
48. Johnson R, Zhang T. Semi-supervised convolutional neural networks for text categorization via region embedding. *Adv Neural Inf Process Syst (NIPS)*. 2015;919-927.
49. Socher R, Perelygin A, Wu J, et al. Recursive deep models for semantic compositionality over a sentiment treebank. Proceedings of International Conference on Empirical Methods in Natural Language Processing (EMNLP); 2013:1631-1642.
50. Abadi M, Agarwal A, Barham P. et al. TensorFlow: large-scale machine learning on heterogeneous distributed systems; 2015.
51. Yang Z, Yang D, Dyer C, He X, Smola A, Hovy E. Hierarchical attention networks for document classification. Paper presented at: Proceedings of NAACL; 2016:1480-1489.
52. Dai AM, Le Quoc V. Semi-supervised sequence learning. *Adv Neural Inf Process Syst (NIPS)*. 2015;3079-3087.
53. Kingma DP, Ba J. Adam: A method for stochastic optimization. Paper presented at: Proceedings of the International Conference on Learning Representations (ICLR); 2015.
54. Hossin M, Sulaiman MN. A review on evaluation metrics for data classification evaluations. *Int J Data Mining Knowl Manag Process (IJDKP)*. 2015;5(2):1-11.
55. Yang Z, Yang D, Dyer C, He X, Smola A, Hovy E. Hierarchical attention networks for document classification. Paper presented at: Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL); 2016:1480-1489.
56. Liu Q, Zhang H, Zeng Y, Huang Z, Wu Z. Content attention model for aspect based sentiment analysis. Paper presented at: Proceedings of the World Wide Web Conference (WWW); 2018:1023-1032.

How to cite this article: Xu Y, Li L, Gao H, Hei L, Li R, Wang Y. Sentiment classification with adversarial learning and attention mechanism. *Computational Intelligence*. 2021;37:774–798. https://doi.org/10.1111/coin.12329