An Easy Data Augmentation Approach for Application Reviews Event Inference

Shikai Guo, Haorui Lin, Jiaoru Zhao, Hui Li, Rong Chen, *Member, IEEE*, Xiaochen Li, and He Jiang, *Member, IEEE*

***Abstract*—Application review event inference aims to assess the effectiveness of application problems in response to user actions, which enables application developers to promptly discover and address potential issues in various applications, thereby improving their development and maintenance efficiency. Despite the development of event inference models for app reviews, which extract them as user action and app problem events and establish a relationship model between events and inference labels, the accuracy of these models is constrained due to limitations in labeling and characterizing noise and the lack of robustness and generalization. To address this challenge, we propose a model called Easy Data Augmentation for Application Reviews Event Inference (short for EDA-AREI), which comprises a denoising component, data augmentation component, and event inference prediction component. Specifically, the denoising component identifies labels and characterizes noisy data to enhance dataset quality, the data augmentation component replaces non-stop words with synonyms to increase textual diversity, and the event inference and prediction component reconstructs the classifier using denoised and augmented data. Experimental results on six datasets of one-star app reviews in the Apple App Store demonstrate that the EDA-AREI method achieves an *Accuracy* of 71.19%, 79.14%, 69.05%, 69.02%, 68.24% and 68.48%, respectively, representing an improvement of 0.83%–2.09% compared to state-of-the-art models. Regarding the *F1-score*, EDA-AREI achieves values of 71.30%, 69.93%, and 68.76% on the threshold_0.5, k-means_2, and random datasets, respectively, outperforming state-of-the-art models by 1.89%–4.02%. Furthermore, EDA-AREI achieves *AUC* values of 75.66% and 73.37% on the threshold_0.5 and k-means_2 datasets, respectively. As a result, EDA-AREI demonstrates substantial improvements in *Accuracy*, as well as enhanced *F1-score* and *AUC* across most datasets, thereby enhancing the model's accuracy and robustness in identifying related action-problem pairs.**

***Index Terms*—Application reviews event inference, data augmentation, Confident Learning.**

I. Introduction

With the increasing popularity of smartphones and mobile devices, more and more software developers are dedicating themselves to the development of mobile applications. Simultaneously, the number of users and downloads in the mobile application market is growing exponentially. To take the lead in application development in this highly competitive market, developers tend to adopt an iterative process to develop, test, and improve the code quality of the applications. Therefore, timely feedback from users is vital for the development and maintenance of mobile applications, including the identification of bugs that need to be fixed, user experience reports for certain features, and identification of new features that users desire [1], [2].

Application developers must collect and leverage user feedback to enhance user satisfaction. Typically, there are two methods for extracting user feedback: (1) utilizing conventional software development channels to gather the actual experience of using the application, such as (i) online forums (e.g., the SwiftKey [3] user feedback forum), (ii) email [4], (iii) bug/change repositories (e.g., Bugzilla [5]), and (iv) crash reporting systems [6]; and (2) using web-based market portals and user reviews of market applications to collect feedback. Modern App marketplaces, such as the Apple App Store, Google Play, and the Android App Store, now offer a simpler way for users to rate Apps, which provides feedback to App developers in the form of ratings and reviews [1]. These reviews present user feedback on various aspects of the App, such as quality, functionality, performance, etc.

However, there are several barriers that hinder App developers from obtaining valuable information from user reviews compared to traditional bug-reporting channels. Firstly, the proportion of "informative" user reviews that provide useful information to developers is relatively low. According to the literature [1], [7], [8], only about one-third of user reviews are helpful to developers, making it difficult for them to filter and process useful information from the many App reviews. Moreover, the quality of user reviews varies widely, ranging

Manuscript received 15 May 2023; revised 12 August 2023; accepted 6 September 2023. Date of publication 20 September 2023; date of current version 17 October 2023. This work was supported in part by the National Natural Science Foundation of China under Grants 62032004 and 61902050, in part by Xinghai Scholar Awards of DUT, in part by the Dalian Excellent Young Project under Grant 2022RY35, and in part by the Fundamental Research Funds for the Central Universities under Grants DUT22RC(3)028 and DUT22ZD101. Recommended for acceptance by G. Canfora. *(Corresponding author: Hui Li.)*

Shikai Guo is with the School of Information Science and Technology, Dalian Maritime University, Dalian 116026, China, and also with the Dalian Key Laboratory of Artificial Intelligence, Dalian 116024, China (e-mail: shikai.guo@dlmu.edu.cn).

Haorui Lin, Jiaoru Zhao, Hui Li, and Rong Chen are with the School of Information Science and Technology, Dalian Maritime University, Dalian 116026, China (e-mail: lhr_dlmu@163.com; zjr@dlmu.edu.cn; li_hui@dlmu.edu.cn; rchen@dlmu.edu.cn).

Xiaochen Li and He Jiang are with the School of Software, Dalian University of Technology, Dalian 116024, China, and also with the Key Laboratory for Ubiquitous Network and Service Software of Liaoning Province, Dalian 116024, China (e-mail: xiaochen.li@dlut.edu.cn; jianghe@dlut.edu.cn).

Digital Object Identifier 10.1109/TSE.2023.3313989

from useful reviews that offer suggestions for improvement or describe specific problems (e.g., "It would be nice to be able to watch other gamers," "Location positioning is inaccurate") to general, useless reviews that offer only praise or complaints (e.g., "I love this App!", "This App is a bit of a dud, not very useful"). Second, App stores contain a large number of reviews, and manual checking of all App reviews by developers is labor-intensive and time-consuming. Pagano et al. [1] reported that common mobile Apps receive about 23 reviews per day, while popular Apps such as Facebook receive an average of 4,275 reviews per day. Finally, user reviews are often unstructured text, which makes parsing difficult. As a result, developers must read large amounts of text data to understand the real needs of users [6].

Currently, user reviews have garnered the attention of many researchers [6], [8], [9], [10], [11], [12]. Chen et al. [8] proposed AR-Miner, which utilizes techniques such as text analysis, machine learning, and topic analysis to identify and group informative reviews. Maalej and Nabil [11] collected and classified overall reviews that describe application problems based on whether they contain bug information, requests for new features, etc. Panichella et al. [12] classified application reviews according to whether they are relevant to the maintenance and development of the software. Pagano et al. [1] extracted valuable information by analyzing the subject matter of user reviews. However, these works focused mainly on the surface of reviews and lacked a deeper understanding of application reviews. Therefore, the following work will concentrate on mining application reviews more deeply.

Recently, Guo et al. [13] reported that user reviews often contain deeper value and meaning than traditional data mining methods suggest. Specifically, user reviews of applications typically convey a short story about the user's experience with the application. A user story is viewed as an ordered collection of events in which the user interacts with the application. The events in a story are event phrases in a sentence describing a single action. Therefore, an application review short story includes at least two types of events: user action events and application problem events. Application problems are application behaviors that are contrary to the user's expectations, while user action events are performed by the user when interacting with the application, which usually reflects the user's expectations of the application. Therefore, it is valuable to extract and synthesize these user action and application problem events.

Therefore, Guo et al. [13] proposed a model for extracting and synthesizing application problem stories from application reviews, named Caspar. The model mainly comprises two parts, namely the event extraction task and the event inference task. The event extraction task aims to efficiently extract and synthesize application problem stories from application reviews as action-problem pairs. On the other hand, the event inference task aims to infer the effectiveness of application problems in response to user actions. This task actively learns inferential relationships between user actions and application problems and uses the learned relationships to infer the relevant application problems corresponding to user actions. Event inference may help developers identify and address problems that may arise in different applications in a timely manner, especially when the user action is known but the application problem is unknown. Furthermore, event inference may help developers leverage reviews from all applications with similar functionality, addressing the limitations of analysis. It may also assist less popular applications in improving their user experience by utilizing reviews from other applications, which have limited user reviews and information.

Although previous work on event inference tasks for application reviews has established links between user action and application problem event pairs to learn the inference relationship between them, the performance of event inference models for application reviews is still limited by two challenges.

Labelling and characterizing noise. The event inference task relies on a dataset that is manually labeled, and this task trains an event classifier by sequentially combining action-problem pairs formed by user actions and application problem events in a review. However, the accuracy of the classified event category is limited by the error of the event classifier, resulting in frequent labeling and representation noise in the synthesized action-problem pairs. As a result, the event inference model is not accurately learned, and this problem becomes more prominent as the size of the data pair increases. The main challenge, therefore, is to effectively deal with the noise in the data and improve the effectiveness of the application review event inference model.

Lack of robustness and generalization. Application reviews provide valuable feedback from real users regarding application functionality and performance. However, this feedback can be highly personalized and expressed in diverse linguistic forms. The existing models are usually trained on a specific corpus (such as one-star App reviews) and may not generalize well to new samples or diverse expressions, leading to poor robustness and generalization. A lack of robustness may result in inaccurate judgments and instability of the model, while a lack of generalization reduces the model's predictive ability for new data. Therefore, the main challenge is to enhance the model's ability to maintain accurate judgments with new external data and improve its robustness and generalization.

To address these challenges, we propose a model named Easy Data Augmentation for Application Reviews Event Inference (EDA-AREI), which aims to denoise the action-problem pairs in the original application review data. This approach addresses the problem of label and characterization noise in action-problem pairs, which can result in inaccurate learning of the event inference model. Additionally, it enhances the diversity of textual representations of the denoised action-problem pairs and improves the robustness and generalization of the application review event inference model. Specifically, EDA-AREI consists of three components: the denoising component, data augmentation component, and event inference prediction component. The denoising component extracts feature information from the action-problem pairs and uses a classifier to generate a prediction probability for each instance. Then, the Confident Learning technique is used to estimate the joint distribution of the predicted true labels and the original noise labels and filter out the error samples, generating clean data

output. The data augmentation component performs synonym replacement (SR) operations on the application problem events in each action-problem pair instance to increase the variety of text representations. This approach allows the model to learn richer information and improve its robustness and generalization. The event inference prediction component relearns the denoised and enhanced data to build the classifier. In the testing phase, we use the trained classifiers to predict and analyze the labels of action-problem pairs in the test set.

To validate the performance of EDA-AREI, we conducted a series of comprehensive experiments on extracting synthetic action-problem pairs from one-star App reviews in the Apple App Store [13]. We used *Accuracy* as the evaluation metric to assess the performance of EDA-AREI in predicting inference labels for App review action-problem pairs. The experimental results show that applying EDA-AREI to remove data noise and increase text diversity can effectively improve the accuracy of model predictions. Specifically, EDA-AREI outperformed the baseline by 2.09% for the dataset with negative samples filtered based on similarity thresholds, and 1.25%, 1.54%, and 0.83% for the datasets with negative samples filtered based on different clustering values, and 1.28% for the dataset with negative samples filtered at random.

In summary, the main contributions of this article are as follows:

- We propose a confidence learning method based on Bi-LSTM to estimate the joint distribution of noisy and true labels in the dataset, clean the dataset of annotation and representation noise, and enhance the label quality of the dataset. This method can effectively learn the inference relationship between user actions in application reviews and application problems, thereby improving the prediction performance of the model.
- To enhance the robustness and generalization of the application reviews event inference model, we propose a text augmentation technique called synonym substitution (SR) to enrich the text and increase the diversity of text representation. This helps to improve the robustness of the model and its ability to generalize to unknown data.
- We conduct extensive experiments to evaluate the effectiveness of EDA-AREI on synthetic action-problem pairs extracted from one-star application reviews. The experimental results show that EDA-AREI outperforms the baselines in terms of *Accuracy*.

The remainder of this article is organized as follows. In Section II, we discuss the motivation for our work. Section III presents the details of EDA-AREI. In Sections IV and V, we describe the experimental setup and present the results, respectively. Threats to validity are discussed in Section VI, and related studies are summarized in Section VII. Finally, we present the conclusion and future work in Section VIII.

II. Motivation

A. Usage Scenario

The task of inferring whether there exists an inferential relationship between known user actions and unknown application problems is referred to as the application review event inference task. Application reviews provide developers and vendors with a wealth of feedback and play a critical role in the subsequent development and maintenance of applications. Fig. 1 illustrates a recent one-star user review on Twitter regarding Twitter's rolling notification problem. From this review, we can extract the user action event "scroll the notification ring" and the corresponding application problem event "unable to see the notification scroll 2 hours ago." This example highlights the correlation between the user action event and the application problem event within the same review. Therefore, to better exploit the value of information contained in application reviews, we seek to gain a deeper understanding of the inference relationship between user action events and application problem events based on the extracted user action events.

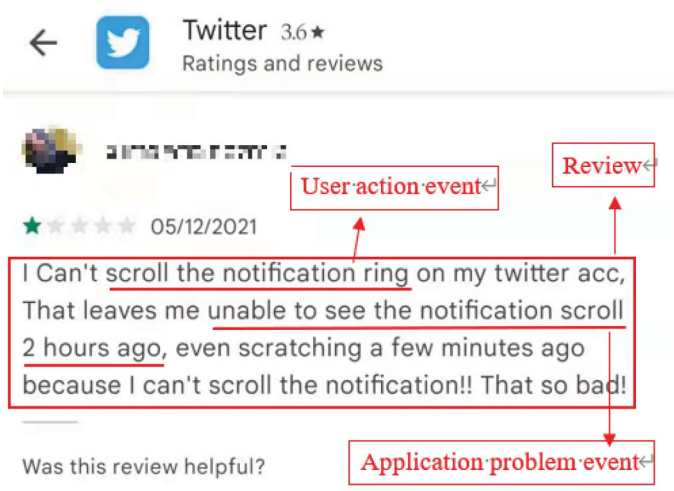


Fig. 1. One-star user reviews about Twitter.

Event inference models based on application reviews are of great importance for application development and quality assurance. They serve the following purposes: (1) They allow for the determination of whether an application problem is caused by the corresponding user behavior or is a random event. This is particularly useful for situations where the user behavior is known, but the application problem is unknown. Identifying such problems in advance can help preempt future issues, ensure the robustness and security of the application, and aid in its development; (2) They can address the limitations of application review analysis. By allowing developers to use reviews from other applications with similar functionality, their analysis is not limited to the current application reviews; (3) They can improve the scalability of application reviews. In the case of applications with few reviews or low popularity, extending reviews from other applications to themselves can fully exploit the additional information available to obtain insights into user needs and unknown bugs, further improving the application and enhancing user experience.

However, the current quality of user action and application problem pairs extracted from application reviews is low and contains a significant amount of data noise, including labelling

TABLE I
EXAMPLES WHERE THE MOISE LABEL IS 1 AND THE TRUE LABEL SHOULD BE 0

SID	231522
Review	The app crashes every time I try to update my status and says "photo cannot be uploaded at this time" when I try to upload pics.... I really wish they would fix this!!! I love the app but I'm sick of the crashes!!! :(
Events	The app crashes
	I try to update my status
	says "photo can not be uploaded at this time"
	I try to upload pics
	I really wish they would fix this

and representation noise. In supervised learning of neural networks, the quality of training set labels strongly influences the learning effect. The presence of a large amount of data noise can reduce the accuracy of the inference model for application review events. Therefore, to improve the accuracy of the model, the data noise must be denoised, and the quality of the labels for action-problem pairs must be improved. Additionally, as the application receives thousands of user reviews every day, and different users have different language expression habits, there may be different problem formulations while the actual meaning remains the same. To prevent overfitting of the model to noisy data and to improve the robustness and generalization of the model, the data must be effectively extended to increase the diversity of textual expressions. By performing these operations, we can train an effective inference model for application review events that will assist application developers in predicting whether there is an inference relationship between user action and possible application problems and ensuring the quality of the application.

B. Motivating Examples

We utilized a dataset based on the original application review event inference dataset id [13], where each id represents the corresponding user action and the textual content of the application problem. However, the dataset is affected by a considerable amount of data noise. Table I presents examples of noise in the dataset, where the action-problem pair has a noisy relationship that is related, but the true relationship should be non-related. Therefore, the label '1' mentioned in Table I represents that the action event and problem event in the sample are related. Similarly, label '0' represents the non-related relationship between the action event and problem event. In Table I, we present a raw unextracted one-star review about Facebook and the five events (comprising both user action events and application problem events) extracted from that review. The action event and corresponding problem event in our examples of noise are highlighted in red. It is evident that two main action-problem event pairs can be constructed, namely "I try to update my status - The app crashes" and "I try to upload pics–says 'photo cannot be uploaded at this time"'. However, the action-problem event pair in the dataset is given as "I try to update my status - says 'photo cannot be uploaded at this time'," and the label is 1. Hence, we consider this sample to be a noisy sample with a true label of 0.

TABLE II
EXAMPLES WHERE THE NOISE LABEL IS 0 AND THE TRUE LABEL SHOULD BE 1

SID	91827
Review	Whenever I try to use the lens it won't work
Events	I try to use the lens
	it won't work
	it takes forever to load (application problem with negative samples)

We have identified data samples with a noise label of 0 that should be labeled as 1. As shown in Table II, we provide an original review about Snapchat, the two events that were extracted, and the application problem that was randomly selected in the construction of the negative sample. These three events form the positive sample "I try to use the lens–it won't work," and the negative sample "I try to use the lens–it takes forever to load." However, the semantics of the application problem for the positive sample and the application problem for the negative sample are similar, as they both convey the semantic message that "trying to use the lens" fails. Therefore, we consider this sample to be a noisy sample with a true label of 1.

The presence of data noise hampers the model's ability to learn valid feature information, which not only impacts the model's learning but also causes over-fitting to noisy data with increasing number of iterations, ultimately compromising the model's robustness and generalization ability. Therefore, motivated by the aforementioned examples, we aim to clean and augment the action-problem pair dataset effectively to reduce the high degree of noise and increase the diversity of data representation. This will enhance the accuracy of the application reviews event inference model, helping application developers anticipate and resolve potential application problems proactively, thereby enhancing the efficiency and quality of application development.

III. THE EDA-AREI MODEL

We first present an overview of the EDA-AREI framework in Section III-A. We will then present details of the components included in EDA-AREI in Sections III-B–III-D.

A. Overview

To address the challenges posed by the presence of a significant amount of data noise and the lack of model robustness and generalization ability in the application reviews event inference dataset, we propose the EDA-AREI model, whose overall structure is illustrated in Fig. 2. EDA-AREI comprises three components, namely, the denoising component, data augmentation component, and event inference prediction component.

The denoising component employs a classifier to vectorize the representation of action-problem pairs, and is trained to obtain predicted probabilities for each sample in the dataset,

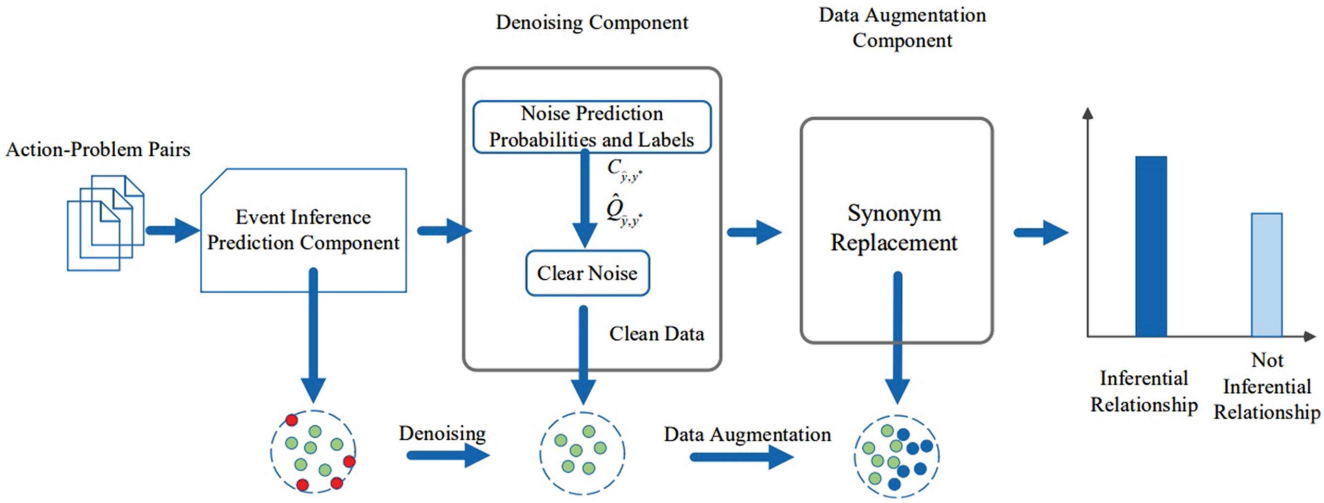


Fig. 2. Overall framework of EDA-AREI.

i.e., the probability that action-problem pairs have an inferential relationship or not. Based on these probabilities and noise labels, the noisy data are counted, jointly estimated, sorted, and pruned, resulting in denoised data that are cleaner and free from annotation and representation noise.

The data augmentation component enhances the textual representation by performing a synonym replacement (SR) operation on the application problem events in each sample. This operation is applied to the denoised dataset to increase the diversity and richness of the application problem text representation, thereby resulting in an expanded dataset.

The event inference prediction component is retrained on the expanded dataset. First, each set of action-problem pairs is vectorized and then inputted into the event inference classifier Bi-LSTM network, which learns text features and inference relationships between action-problem pairs by aggregating information from both forward and backward sequences. This enables the model to capture complete past and future contextual information for each point in the input sequence and output layer. Finally, the trained model is employed to predict and analyze the action-problem pairs.

The details of EDA-AREI are presented in Algorithm 1. Firstly, the noisy instances are partitioned into t groups, with the last group selected as the test set and the remaining groups used for training the model to generate prediction probabilities for each group of noisy instances. Subsequently, the training set is denoised to eliminate the annotation and representation noise and obtain a cleaner training set. Then, the denoised training set is augmented using the SR technique to enhance the richness and diversity of the text representation. Lastly, the classifier is retrained on the augmented training set, and the prediction probability is computed for each instance in the test set.

Algorithm 1 EDA-AREI

Input: A dataset $X = \langle x, \tilde{y} \rangle$ with n examples, m class labels, and t folds of dataset division.

Output: The $n \times m$ matrix of predicted probabilities $\hat{P}$, where $\hat{P}[i][j] := \hat{p}(\tilde{y} = j; x, \theta)$.

```
Divide X evenly into T-folds: X_1, X_2, ..., X_t;
for i = 1 to t do
    X_t as a test dataset X_test and the others as the training dataset X_train;
    Build the classifier Bi-LSTM on the X_train;
    Predict the classification probability P̂_i of the X_i using Bi-LSTM;
    Perform CL denoising components to get X_clean_train;
end for
for i = 1 to t − 1 do
    Perform SR on X_clean_train to get X_change_train;
end for
Retrain Bi-LSTM with X_change_train;
Predict the classification probability P̂_t of the X_t using Bi-LSTM;
return P̂.
```

B. Denoising Component

To tackle the issue of the substantial amount of noisy data in the application reviews event inference dataset, we employ the denoising component to eliminate the vast amount of unclean data and guarantee the dataset's quality. Specifically, we employ Confidence Learning (CL) [14] techniques for the implementation.

In the inference dataset of application review events containing noisy labels, $[m]$ represents the ensemble of labels in the dataset and $m \in \{0, 1\}$, where a label of 0 indicates no inference relationship for action-problem pairs and a label of 1 indicates an inference relationship for action-problem pairs. $X =: (x, \tilde{y})^n \in \left(R^d, [m]\right)^n$ represents the dataset of n instances in the dataset and the associated noise label $\tilde{y}$, where $\tilde{y} \in [m]$, x denotes an extracted action-problem pair. $(x, \tilde{y})$ Appearing in pairs in the dataset X denotes action-problem pairs and corresponding labels that need to be cleaned up, i.e. noisy data.

For each example, we assume that it has a potential, uncorrupted, true label y^*, indicating whether the action-problem

pair has an inference relationship. The subset of instances in the dataset with the noise-like label i is denoted as $X_{\tilde{y}=i}$. In addition, for an action-problem pair x in the application review event inference dataset, $\hat{p}\left(\tilde{y}=i; x \in X_{\tilde{y}=i}, \theta\right)$ denotes the predicted probability that the instance belongs to the label $\tilde{y}$ and also denotes confidence learning.

The denoising component of the EDA-AREI model requires two inputs: (1) the sample prediction probabilities $\hat{P}_{k,i} = \hat{p}\left(\tilde{y}=i; x_k, \theta\right), x_k \in X$, where θ denotes the model used to obtain the sample prediction probabilities $\theta : x \to \hat{p}\left(\tilde{y}=i; x_k, \theta\right)$. In the EDA-AREI model, we train the Bi-LSTM network as the model here θ to obtain the sample prediction probabilities for each action-problem pair in the dataset. (2) The noisy label vector $\tilde{y}_k, x_k \in X$, i.e. the labels of the action-problem pairs, is used for all $x_k \in X$, and the two inputs are kept connected by index k.

Confidence learning identifies representational and annotation noise in the existing application review event inference dataset to improve the accuracy of the model, and consists of three main steps.

(1) **Counting:** estimating the joint distribution of noisy labels and true labels.

The joint distribution describes the distribution of the noise labels $\tilde{y}$ and the true labels y^*. For each sample of the event inference dataset, we count the instances where they may belong to a certain class and calibrate them. In order to estimate the joint matrix, the following steps are required.

a) Calculating the confidence joint matrix $C_{\tilde{y},y^*}$.

The confidence joint matrix $C_{\tilde{y},y^*} \in Z_{\geq 0} M \times M$ indicates that an instance x with label $\widetilde{y}=i$ has a sufficiently high probability that $\hat{p}(\widetilde{y}=j; x, \theta)$ belongs to the label $y^*=j$. M represents the number of categories. We consider these instances as noise data that mislead the training of EDA-AREI. The item of $C_{\tilde{y},y^*}$ is the number of each category. Formally, the confidence joint matrix is expressed as

$$C_{\tilde{j},y} := \{x \in X_{\tilde{y}=i} : \hat{p}(\tilde{y}=j; x, \theta) \geq t_j\}, \tag{1}$$

and $\hat{p}(\tilde{y}=j; x, \theta)$ means the probability that a instance x with the noise label $\tilde{y}=i$ was estimated by model θ (Bi-LSTM, SVM etc.) as *j*, i.e. the actual label y^* is *j*. When the estimated probabilities are higher than the per-class threshold t_j, we determine the action-problem pairs as the category of *j*. To calculate $C_{\tilde{y},y^*}$, the per-class threshold t_j is needed. Threshold t_j is the expected self-confidence for each class and is expressed as

$$t_j = \frac{1}{|X_{\tilde{y}=j}|} \sum_{x \in X_{j-j}} \hat{p}(\tilde{y}=j; x, \theta), \tag{2}$$

When the instance with noise label as *i* achieves higher probability to *i*, the threshold t_i will be higher and an instance considered as actual label $y^*=i$ is required to have a bigger probability. Conversely, if the average probability to *j*, from the instances with noise label as *j*, gets a lower value, the instance will be easier to achieve the threshold t_j and estimated as the category of *j*. The thresholds provide a more reasonable method to guess the true label y^*, rather than selecting the category with the greatest prediction probability. For a certain instance, if there are prediction probabilities on more than one category outperforming the threshold, we choose the category with the biggest probability as the true label $y^*=j$. The equation is expressed as

$$j = \underset{l \in [m]: \hat{p}(\tilde{y}=l; x, \theta) \geq t_l}{\arg\max} \hat{p}(\tilde{y}=l; x, \theta). \tag{3}$$

As seen the estimated confident joint matrix $C_{\tilde{y},y^*}$, the diagonal entries counts the clean labels, representing the same true label y^* as the noise label $\tilde{y}$. On the contrary, the non-diagonal entries denote the number of incorrect labels. Confident joint matrix is used as one of the bases for the removal of noise data.

b) Estimation of the joint distribution matrix $\hat{Q}_{\tilde{y},y^*}$.

Using the confidence joint matrix $C_{\tilde{y},y^*}$, we estimate $\hat{Q}_{\tilde{y},y^*}$ as

$$\hat{Q}_{\tilde{y}=i,y^*=j} = \frac{\frac{C_{\tilde{y}=i,y^*=j}}{\sum_{j \in M} C_{\tilde{y}=i,y^*=j}} \cdot |X_{\tilde{y}=i}|}{\sum_{i \in M, j \in M} \left(\frac{C_{\tilde{y}=i,y^*=j}}{\sum_{j \in M} C_{\tilde{y}=i,y^*=j}} \cdot |X_{\tilde{y}=i}| \right)}. \tag{4}$$

The joint distribution matrix accurately reflects the distribution of false (noise) and true labels in the real world, effectively capturing the noise distribution. The main purpose of calculating the joint distribution $\hat{Q}_{\tilde{y},y^*}$ is to identify and eliminate the annotation noise and representation noise in the data. It has been demonstrated in the original article [14] that this estimation method approaches the true distribution as the size of the data increases.

(2) **Cleaning:** identify wrong action-problem pairs and filter them.

After obtaining the confidence joint matrix of action-problem pairs $C_{\tilde{y},y^*}$ and the joint distribution matrix of noise labels and true labels $\hat{Q}_{\tilde{y},y^*}$, we will filter the wrong samples of action-problem pairs using a total of five different methods using sorting, pruning and other heuristics.

a) $C_{\text{confision}}$: the Boolean value of $\tilde{y}_k \neq \arg\max_{j \in [m]} \hat{p}\left(\tilde{y}=j; x_k, \theta\right), x_k \in X$ is used as the basis for evaluating error labels, where true indicates a label error and false indicates clean data, and the samples are filtered for action-problems indicated as true.

b) $C_{\tilde{y},y^*}$: the matrix $C_{\tilde{y},y^*}$ denotes the number of correct labels and the non-diagonal entries denote the number of incorrect labels. Therefore, the action-problems that entered the non-diagonal term cells during the construction of the joint matrix $C_{\tilde{y},y^*}$ is selected for filtering of the samples.

c) Pruning by class (PBC): for each manually-labelled class $i \in [m]$, rank the samples with the lowest probability $\hat{p}\left(\tilde{y}=i; x \in X_i\right)$ within each class, and select $n \cdot \sum_{j \in [m]: j \neq i} \left(\hat{Q}_{\tilde{y}=i,y^*=j}[i]\right)$ action-problem samples to filter.

d) Pruning by noise rate (PBNR): for the non-diagonal entries of the $\hat{Q}_{\tilde{y}=i,y^*=j}, i \neq j$ matrix, the samples are sorted by the maximum interval $\hat{p}_{x,\tilde{y}=j} - \hat{p}_{x,\tilde{y}=i}$ and the $n \cdot \hat{Q}_{\tilde{y}=i,y^*=j}$ action-problems are selected for filtering.

e) Combine pruning by class (PBC) and pruning by noise rate (PBNR): if both the PBC and PBNR methods remove the action-problem pair instance, we choose to remove the instance.

C. Data Augmentation Component

To enhance the robustness and generalization ability of the application reviews event inference model and reduce its sensitivity to new input data, it is necessary to effectively expand the event inference dataset, thereby increasing the diversity and richness of the data samples while maintaining the accuracy of the model's prediction for new action-problem pairs.

In the Data Augmentation component, we utilize a synonym replacement (SR) [15] operation to enrich the text representation of the denoised application problem events with various synonyms. The details are as follows.

Synonym replacement (SR): n non-stop words are randomly selected in a sentence, then synonyms are randomly selected from a synonym dictionary and replaced.

However, due to the shorter length of user action events compared to application problem events, SR operations on short texts are not as effective as longer texts. Moreover, performing SR on both events may introduce more noise problems. To ensure a richer text representation without introducing more interference, we only perform SR on the application problem event in the action-problem pair. Additionally, to maintain the original inference labels of the action-problem pair after the SR operation, we vary the number of words n replaced by synonyms based on the sentence length l using the expression $n = \alpha l$, where α represents the percentage of words changed in a sentence.

D. Event Inference Prediction Component

After being processed by the denoising and data augmentation components, we obtained a cleaner and richer dataset. Based on this, we use the event inference prediction component to re-learn the inference relationships between action-problem pairs and make predictions about inference labels. This process involves four main sub-steps.

(1) Vectorized representation

We first convert the action-problem pairs into vector representations and then apply Bi-LSTM classification techniques on these vectors. In this work, we utilized Spacy's pre-trained statistical model of English, namely, en_core_web_lg [16], to convert each token into a vector, and each vector had a size of 300 dimensions.

(2) Connecting action-problem pairs

We convert both the action and the corresponding problem into vector representations using the aforementioned language model. Then, we concatenate the two vectors using a special token [SEP]. This concatenation results in an event pair that can be learned jointly. Moreover, the [SEP] token also needs to be converted into a word vector, which is fed into the Bi-LSTM network along with the event pair vector during training.

(3) Learning text features

We train the Bi-LSTM network to learn the inference relationships between action-problem pairs. Bi-LSTM consists of two independent LSTM [17] recurrent layers, where the first layer extracts text features in a forward chronological order, and the second layer extracts text features in a backward chronological order. The two extracted feature vectors are then concatenated to form a new vector representing the final feature expression for the event pair. Bi-LSTM can effectively utilize contextual information, enabling it to capture both past and future information.

The LSTM introduces a gate memory mechanism, which is mainly divided into forget gates, input gates, and output gates. The gate controls are weights for information, which determine how much information can be retained. Different gates have the same calculation but play diverse roles with different parameters. We denote an action event containing t words and the application problem event sentence as $x = (x_1, x_2, \ldots, x_t), x_t \in R^{n \times d}$, where n denotes the number of samples and d denotes the vector dimension of 300 dimensions. The forward information extraction process is as follows

Forget Gate: controls how much information needs to be forgotten about the cell state at the last moment c_{t-1} and the calculation involved is expressed as

$$f_t = \sigma\left(W_{fh} h_{t-1} + W_{fx} x_t + b_f\right), \tag{5}$$

where $W_{fh} \in R^{h \times h}$, $W_{fx} \in R^{d \times h}$ and $b_f \in R^{1 \times h}$ represent the weight coefficients and bias values of the feature extraction process, respectively, and σ is the activation function. The calculation of f_t takes into account both the current input information x_t and historical information h_{t-1}. Considering the impact of all tokens in action-problem pairs, it is more reasonable to use a gate to decide whether the information stays or goes.

Input Gate: consists of two parts: the input section i_t and the candidate state $\tilde{c}_t$. The input gate controls how much of the input information at the current moment needs to be saved to the cell state c_t. The input section i_t determines which information needs to be updated and is expressed as

$$i_t = \sigma\left(W_{ih} h_{t-1} + W_{ix} x_t + b_i\right), \tag{6}$$

where the parameters are the same as the forget gate.

It is worth mentioning that the input information at the current time is not directly used in the input data x_t, but in the candidate state $\tilde{c}_t$ using the same principle of fusing historical and current information. The candidate state $\tilde{c}_t$ represents the temporary state at the current moment and is expressed as

$$\tilde{c}_t = \tanh\left(W_{ch} h_{t-1} + W_{cx} x_t + b_c\right), \tag{7}$$

where the parameters are the same as the forget gate.

Cell Status: combined information from the forget and input gates, recording the history up to the current moment. Cell states c_t contain long-term memory information, which consists of the cell status from the last time c_{t-1} and the candidate state $\tilde{c}_t$ in time *t*. The cell status memorizes more historical information and is expressed as

$$c_t = c_{t-1} \odot f_t + i_t \odot \tilde{c}_t. \tag{8}$$

Output Gate: consists of the output section O_t and the hidden state h_t which controls how much information from the current cell state c_t needs to be output to the external hidden state h_t. The output section O_t is expressed as

$$o_t = \sigma\left(W_{oh} h_{t-1} + W_{ox} x_t + b_o\right), \tag{9}$$

TABLE III
APPLICATION REVIEWS EVENT INFERENCE DATASET

Datasets	Description
threshold_0.5	The similarity threshold is <0.5. First, we use the random.shuffle() function in Python to disrupt all available application problem events. Then, iterate through each application problem event in turn, selecting the first application problem event with a similarity to the actual application problem below a preset threshold.
threshold_0.25	The similarity threshold is <0.25. The negative sample selection is the same as the similarity threshold of 0.5, the only difference being that the threshold is set to 0.25.
k-means_2	Clustering. Using k-means (implemented in Scikit-Learn), all application problem events are clustered into two groups based on the cosine similarity of their USE vectors. For each action-problem pair event, we find the cluster to which the application problem event in the positive example belongs, and randomly select an application problem event from the other cluster as the negative example.
k-means_10	Clustering. The negative sample selection method is the same as k-means_2. The difference is that all application problem events are clustered into ten groups, and a problem event is randomly selected from the other nine clusters when generating the negative example.
k-means_100	Clustering. The negative sample selection method is the same as k-means_2. The difference is that all application problem events are clustered into one hundred groups and a problem event is randomly selected from another ninety-nine clusters when generating negative examples.
random	Completely random. No measures are taken and a random selection from the application problem is used as a negative sample.

where the parameters are the same as the forget gate.

In our case, the cell status memorizes the information of the tokens processed so far. As the time sequence t increases, the internal connection between action event and problem event is gradually extracted in the cell status. The hidden state h_t contains short-term strong memory information, which is significantly influenced by the token input at the current time. The hidden state h_t is expressed as

$$h_t = o_t \odot \tanh(c_t). \tag{10}$$

The hidden state significantly compresses the natural language and memorizes the internal relationship between action and problem events, which will be valuable for the subsequent classification task.

After extracting the forward information, we obtain the forward hidden state $\vec{h}_t$, and similarly, by extracting the reverse information, we obtain the reverse hidden state $\overleftarrow{h}_t$. Joining the two hidden states results in the final hidden state $H_t \in R^{n\times 2h}$, which is then fed to the output layer. Through the above steps, the Bi-LSTM network learns the respective textual features of the action-problem pairs and the inference relationship between them, laying the foundation for the subsequent prediction of event inference labels.

(4) Predictive Event Inference Label

We predict inference relationship labels for action-problem pairs in the test dataset by training an inference model on application review events. For each action-problem pair in the test dataset, our model predicts the probability of an inference relation and the probability of no inference relation, and uses the maximum probability as the criterion for determining the corresponding inference relation label, i.e. 0 or 1.

IV. EXPERIMENTAL SETUP

In this section, we will describe our datasets, baselines, evaluation metrics and training details.

A. Datasets

We utilized the dataset of application reviews event inference (Guo [13]) as the primary dataset. This dataset is created by amalgamating six distinct datasets that are based on various negative sampling strategies. The textual content of six datasets, namely threshold_0.5, threshold_0.25, k-means_2, k-means_10, k-means_100, and random datasets, is determined based on six dataset ids. In each dataset, an action-problem pair event includes two samples, i.e., a positive example and a negative example. The original action-problem pair, i.e., ordered event pair, is used as a positive example since it consists of actual user action follow-up events. On the other hand, a negative example (random event pair) is generated by merging user action and random application problem events using a negative sampling strategy. We selected 153,178 (90%) data samples from the dataset as the training set to train the model, and 17,020 (10%) data samples as the test set to assess the model's performance. Table III provides the detailed information about the datasets.

B. Baselines

Regarding performance comparison, we adopted the baseline methods used by Guo et al. [13], namely SVM [18] and USE+SVM [19]. We also employed their state-of-the-art method Bi-LSTM [13] as our baseline and compared /approach with these three baselines. For the methods for which the article provides source code (SVM, USE+SVM, and Bi-LSTM), we

replicated their methods on the dataset. Additionally, our work aimed to optimize Bi-LSTM by incorporating data denoising and data augmentation.

C. Evaluation Metrics

As per Guo's work [13], we evaluated the performance of EDA-AREI using the *Accuracy* metric. In a binary classification problem, instances are categorized into Positive or Negative classes. During actual classification, the following four cases may arise: TP denotes the number of samples where the predicted action-question pair has an inference relationship and also has an inference relationship in reality. TN represents the number of samples where the predicted action-question pair has no inference relationship, and it is also not present in reality. FP corresponds to the number of samples where the predicted action-question pair has an inference relationship but does not exist in reality. FN denotes the number of samples where the predicted action-question pair does not have an inference relationship, but it does exist in reality.

Accuracy is the number of samples whose labels are correctly predicted as a proportion of the total number of samples predicted. It is defined as follows:

$$Accuracy = \frac{TP + TN}{TP + FN + FP + TN}. \tag{11}$$

Furthermore, to comprehensively evaluate the effectiveness of the denoising and data augmentation components, which distinguish EDA-AREI from baseline approaches, we introduce additional metrics, including *Precision*, *Recall*, *F1-score*, and *AUC*, and analyze the experimental results in addressing RQ1.

Precision is the proportion of samples for which the positive label is correctly predicted, out of the total number of samples that are predicted to have a positive label. This can be mathematically expressed as follows:

$$Precision = \frac{TP}{TP + FP}. \tag{12}$$

Recall is the proportion of samples with a true positive label that were correctly predicted as positive, out of the total number of samples with a true positive label. It can be mathematically expressed as follows:

$$Recall = \frac{TP}{TP + FN}. \tag{13}$$

F1-score is a comprehensive metric that combines both *Precision* and *Recall*. It provides a more accurate evaluation of a model's accuracy in identifying relevant action-problem pairs. The *F1-score* is defined as follows:

$$F1 - score = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{14}$$

AUC represents the area under the ROC curve along the X-axis. The ROC curve is plotted on a coordinate system, with the X-axis representing the false positive (FP) ratio, and the Y-axis representing the true positive (TP) ratio. The ROC curve is generated by varying the threshold that determines whether a sample's score is high enough to be classified as positive. The *AUC* is calculated as the area under the ROC curve, with higher *AUC* scores indicating a greater likelihood that the model will correctly classify a positive sample with a higher score than a negative sample.

D. Training Details

This section presents the detailed parameter settings for training EDA-AREI. For the denoising stage, we adopted Guo et al.'s [13] method and utilized the Spacy pre-trained English statistical model en_core_web_lg [16] to vectorize each text token with a dimension of 300 dimensions. We used the Bi-LSTM from the CL denoising method to obtain the prediction probabilities and minimized the Sigmoid cross-entropy between the output and the target using the Adam optimizer. Moreover, we set other parameters as follows: the number of Bi-LSTM hidden layers is 256, the initial learning rate is 10^{-4}, the training batch size is 1, and we trained a total of 20 epochs. During the data augmentation stage, we set the SR ratio to 0.1. For the event inference prediction stage, we employed the denoised dataset mentioned in Section III-B and retrained the Bi-LSTM model using the same parameter settings as in the denoising stage. However, we added dropout to the last layer of the Bi-LSTM and set it to 0.3 to avoid overfitting of the model.

V. Experimental Results

In this section, we conduct eight experiments to evaluate EDA-AREI model.

A. Can EDA-AREI Predict Event Inference Relationships?

Motivation. The objective of this experiment is to verify the efficacy of EDA-AREI in inferring relational predictions for application review events, and to compare its performance against that of baseline approaches.

Method. To validate the effectiveness of EDA-AREI, we compare its performance with that of SVM [18], USE+SVM [19], and Bi-LSTM [13] in terms of *Accuracy*, *Precision*, *Recall*, *F1-score*, and *AUC*. For the baseline approach, we replicate its experiments and experimental setup. All experiments are trained and tested on the dataset mentioned in Section IV-A.

Results. The findings are presented in Table IV. EDA-AREI attains superior performance in terms of *Accuracy*, *F1-score*, and *AUC* on most datasets. *Accuracy* measures the percentage of correctly predicted samples among all samples. EDA-AREI emerges as the top-performing model on five datasets, with *Accuracy* values of 71.19%, 79.14%, 69.05%, 69.02%, and 68.24%, respectively, which surpass the best baseline by 2.09%, 1.25%, 1.54%, 0.83%, and 1.28%, respectively. This indicates that EDA-AREI is proficient in predicting action-problem pairs accurately. *F1-score* is a comprehensive metric for evaluating binary classifiers that takes into account the numbers of true positives (TP), false positives (FP), and false negatives (FN). A model can achieve a relatively greater *F1-score* value, approaching 100%, when the number of TP is substantially higher than that of FP and FN. In terms of this metric, EDA-AREI achieves *F1-score* values of 71.30%, 69.93%, and 68.76% on the threshold_0.5, k-means_2, and random datasets,

TABLE IV
COMPARISON OF EDA-AREI AND BASELINE MODELS IN TERMS OF ACCURACY, F1-SCORE AND AUC

	threshold_0.5			threshold_0.25			k-means_2		
Models	acc	f1-score	auc	acc	f1-score	auc	acc	f1-score	auc
SVM	60.70%	58.43%	58.27%	72.90%	75.31%	75.72%	58.50%	66.19%	54.78%
USE+SVM	68.10%	69.48%	69.13%	**82.80%**	79.42%	79.34%	67.80%	65.76%	65.11%
Bi-LSTM	69.10%	67.28%	74.51%	79.60%	**79.62%**	**85.03%**	67.80%	68.24%	72.97%
EDA-AREI	**71.19%**	**71.30%**	**75.66%**	79.14%	79.27%	83.55%	**69.05%**	**69.93%**	**73.37%**
	k-means_10			k-means_100			random		
	acc	f1-score	auc	acc	f1-score	auc	acc	f1-score	auc
SVM	46.55%	52.13%	46.55%	46.61%	50.26%	46.61%	55.30%	49.99%	46.32%
USE+SVM	65.81%	66.35%	65.81%	65.49%	65.87%	65.49%	66.00%	65.70%	65.36%
Bi-LSTM	67.48%	**69.02%**	**73.20%**	67.41%	**69.64%**	**72.63%**	67.20%	66.22%	**72.03%**
EDA-AREI	**69.02%**	68.68%	72.58%	**68.24%**	68.13%	72.14%	**68.48%**	**68.76%**	71.98%

respectively. This indicates that EDA-AREI is capable of predicting relevant action-problem pairs accurately. *AUC* takes into account both the classification ability of positive and negative samples and provides a reasonable assessment without any restriction on dataset distribution. EDA-AREI achieves *AUC* values of 75.66% and 73.37% on the threshold_0.5 and k-means_2 datasets, respectively, which exceed the best baseline model by 1.15% and 0.40%, respectively.

Upon reviewing the results, it is evident that SVM and USE+SVM exhibit poor performance, particularly with respect to *F1-score* and *AUC* on the k-means_2 and random datasets. The negative composition of these datasets implies the existence of a greater number of noisy data points compared to the threshold_0.5 or threshold_0.25 datasets. One of the primary reasons for this unsatisfactory outcome is the large number of noisy data points in these two datasets, which disrupts the learning process of the model, thereby compromising its effectiveness. In contrast, EDA-AREI demonstrates superior accuracy over Bi-LSTM across most datasets and the three performance metrics evaluated. Bi-LSTM's performance can be inconsistent, it is sometimes inferior to USE+SVM in terms of *F1-score* on the threshold_0.5 dataset and inferior to USE+SVM in terms of *Accuracy* on the k-means_2 dataset. In contrast, EDA-AREI exhibits more consistent and stable performance, effectively predicting unknown action-problem pairs with a high degree of *accuracy*. Despite the fact that EDA-AREI employs the same deep learning model as Bi-LSTM, it eliminates noisy data and enhances the textual diversity of the data through synonym replacement. With less harmful data and a broader range of textual expressions, EDA-AREI acquires a more precise understanding of the relationship between the training data, rendering it highly effective when dealing with unknown data. The *Precision* and *Recall* experimental results are presented in the Appendix (https://github.com/lhr024/EDA-AREI/blob/master/Tables/APPENDIX.pdf).

Conclusion. EDA-AREI often outperforms the baseline approaches, achieving superior results in terms of *Accuracy*, *F1-score*, and *AUC* in the majority of cases. EDA-AREI not only takes into account the labeling and representation noise limitations of action-problem event pairs, but also the limitations of model robustness and generalization, thus providing better performance than the baseline approaches.

B. What Is the Impact of Different Threshold Calculation Methods on Model Performance in EDA-AREI?

Motivation. To tackle the issue of a substantial amount of noise in action-problem event pairs, we employ the denoising module of EDA-AREI to eliminate labeling and characterizing noise. Within the denoising module, we employ the confidence level to indicate the predicted probability that an action-problem event pair belongs to its assigned label, and the threshold value to represent the anticipated confidence level for each class. Various approaches for calculating thresholds directly influence the joint estimation of true and noisy labels, and consequently, the detection of labelling errors and characterizing noise. Hence, in this RQ, we examine the influence of diverse methods of calculating thresholds on the performance of the model.

Method. To investigate the impact of different threshold calculation methods on the performance of the EDA-AREI model, we set the thresholds in the denoising component to the lower quartile, median, and upper quartile of the confidence level distribution to denoise the training set for the six different sets of action-problem events mentioned in Section IV-A. Subsequently, the denoised data are used to retrain the Bi-LSTM in the event inference prediction component and the performance of the EDA-AREI model is evaluated using the *Accuracy* metric.

Results. The results of the experiments investigating the effect of different threshold calculation methods in the denoising component on the model performance are presented in Table V. Overall, for the evaluation metric *Accuracy*, the median threshold calculation method outperforms the other two methods in the threshold_0.25, k-means_2, and k-means_10 datasets, achieving accuracies of 79.85%, 68.59%, and 68.93% respectively, compared to accuracies of 79.26%, 68.33%, and 68.39% achieved using the larger quartile threshold calculation method. In these datasets, the median threshold calculation method improves performance by 0.67%, 0.49%, and 1.2% over the larger quartile threshold calculation method. On the other hand, the larger quartile threshold calculation method performs better in the threshold_0.5, k-means_100, and random datasets, achieving accuracies of 70.61%, 68.21%, and 68.12% respectively, which are 0.14%, 0.65%, and 0.51% better than the accuracies achieved using the median threshold calculation

TABLE V
COMPARISON OF DIFFERENT THRESHOLDS ON *ACCURACY*

Methods	threshold_0.5	threshold_0.25	k-means_2	k-means_10	k-means_100	random
lower quartile	70.50%	79.85%	68.59%	68.93%	68.08%	68.02%
median	70.47%	**79.93%**	**68.82%**	**69.59%**	67.56%	67.61%
larger quartile	**70.61%**	79.26%	68.33%	68.39%	**68.21%**	**68.12%**

TABLE VI
COMPARISON OF DIFFERENT DATA AUGMENTATION METHODS ON *ACCURACY*

Methods	threshold_0.5	threshold_0.25	k-means_2	k-means_10	k-means_100	random
TF-IDF	70.42%	**79.86%**	68.45%	68.75%	**68.59%**	67.97%
EDA-AREI	**71.19%**	79.14%	**69.05%**	**69.02%**	68.24%	**68.48%**

method. The different results obtained for the six datasets can be attributed to the fact that although the positive samples are the same, the negative samples are randomly selected based on different similarity thresholds and clustering methods, leading to more variability between the negative samples. Despite these differences, the median threshold calculation method is still effective in improving the performance of the EDA-AREI model and is not significantly different from the larger quartile threshold calculation method in the threshold_0.5, k-means_100, and random datasets.

Conclusion. After analyzing and comparing the experimental results, it can be concluded that the median threshold calculation method is more effective in improving the performance of the model. This is because the joint distribution obtained by this method is closer to the true distribution, which facilitates the accurate identification of noisy labels. As a result, the EDA-AREI model performs better in predicting the inference of application review events.

C. What Is the Impact of Different Data Augmentation Methods on Model Performance in EDA-AREI?

Motivation. The data augmentation component in EDA-AREI is a method for enriching textual diversity. In order to test the difference in performance between EDA-AREI and other data augmentation methods for inference models of application review events, we will validate the different data augmentation methods.

Method. We employ EDA-AREI as well as unsupervised data augmentation methods based on Term Frequency-Inverse Document Frequency (TF-IDF) [20] synonym replacement to manipulate the application problem events in denoised action-problem event pairs, thereby increasing the diversity of the text representation. Subsequently, the enriched action-problem event pairs are used to retrain the Bi-LSTM event inference model and evaluate the model performance.

Results. The performance of different data augmentation methods on the event inference model is presented in Table VI. EDA-AREI outperforms TF-IDF on threshold_0.5, k-means_2, k-means_10, and random datasets with *Accuracy* rates of 71.19%, 69.05%, 69.02%, and 68.48%, respectively. EDA-AREI shows an improvement of 0.77%, 0.60%, 0.27%, and 0.51% over TF-IDF, respectively. This is because TF-IDF's approach of measuring the importance of a word using word frequency is not comprehensive enough, and textual diversification requires consideration of both keyword and non-keyword diversity. Conversely, EDA-AREI takes into account both types of diversity, leading to better event inference performance. However, TF-IDF performs better on threshold_0.25 and k-means_100 datasets, achieving accuracy rates of 79.86% and 68.59%, respectively. In comparison, EDA-AREI achieves *Accuracy* rates of 79.14% and 68.24%, respectively. The reason for TF-IDF's better performance in these datasets is that the negative samples selected using the similarity threshold less than 0.25 and clustering 100 are more different from the original samples. Therefore, TF-IDF's keyword synonym replacement is more targeted than synonym replacement for all words, leading to more enriched text diversity and diverse data representations with more textual information.

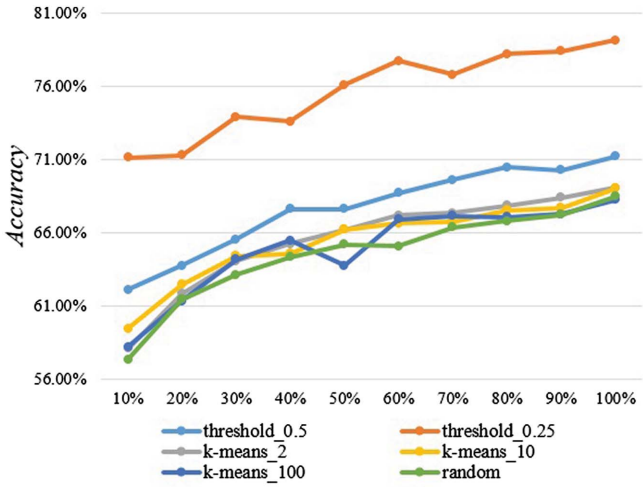


Fig. 3. Comparison of different training set sizes on *Accuracy*.

Conclusion. For the different data augmentation methods, EDA-AREI provides better performance improvements for the application reviews event inference model.

D. Effect of Different Training Set Sizes on EDA-AREI Performance

Motivation. When training a neural network, the size of the training dataset can significantly impact the effectiveness of the resulting model. In this research, we aim to analyze the impact of different training set sizes on the performance of EDA-AREI.

TABLE VII
COMPARISON OF DIFFERENT TYPES AND THE NUMBER OF INCORRECT PREDICTION SAMPLES ESTIMATED BY MODELS

Models	threshold_0.5		threshold_0.25		k-means_2	
	FPR	FNR	FPR	FNR	FPR	FNR
SVM	20.51%	20.90%	11.32%	12.94%	36.73%	**8.16%**
SVM+USE	15.53%	14.89%	**10.51%**	10.15%	18.37%	16.52%
Bi-LSTM	**14.46%**	15.93%	10.75%	9.84%	**15.23%**	16.90%
EDA-AREI	15.76%	**13.42%**	12.51%	**8.65%**	16.92%	14.33%
	k-means_10		k-means_100		Random	
	FPR	FNR	FPR	FNR	FPR	FNR
SVM	32.55%	20.90%	30.37%	23.02%	30.19%	23.45%
SVM+USE	17.89%	16.30%	17.80%	16.70%	17.82%	16.82%
Bi-LSTM	17.27%	**14.55%**	**16.27%**	16.46%	**15.66%**	17.50%
EDA-AREI	**15.88%**	14.83%	16.40%	**15.27%**	17.20%	**15.13%**

Method. We randomly select action-problem pairs from each of the training sets mentioned in Section IV-A to create nine new datasets consisting of 10%, 20%, 30%, 40%, 50%, 60%, 70%, 80% and 90% of the original dataset. Subsequently, we apply EDA-AREI to these new training sets of varying sizes and employ the *Accuracy* metric to assess the performance of the models.

Results. The impact of varying training set sizes on the performance of the event inference model is presented in Fig. 3, where nine new datasets, each consisting of randomly selected action-problem pairs, are created from the original dataset, ranging in size from 10% to 90%. The resulting models are evaluated using *Accuracy*. The results indicate that as the size of the training set increases, the accuracy of the model gradually improves. Specifically, when the training set size is 10% of the original dataset, *Accuracy* values of 62.1%, 71.11%, 58.09%, 59.46%, 58.21%, and 57.35% are achieved for threshold_0.5, threshold_0.25, k-means_2, k-means_10, k-means_100, and random datasets, respectively. As the size of the training set increases to that of the original dataset, the values of *Accuracy* increase to 71.19%, 79.14%, 69.05%, 69.02%, 68.24%, and 68.48%. It can be observed that with every 10% increase in the dataset, the final EDA-AREI model achieves an improvement in *Accuracy* of 9.09%, 8.03%, 10.96%, 9.56%, 10.03%, and 11.13% for threshold_0.5, threshold_0.25, k-means_2, k-means_10, k-means_100, and random datasets, respectively.

The experimental results clearly demonstrate that the effectiveness of EDA-AREI is positively correlated with the size of the training set. These findings suggest that a small number of training samples may result in incomplete feature learning and poor model performance. As the size of the training set gradually increases, the learned features become progressively more diverse and comprehensive, leading to improved model performance. However, once the training set reaches a certain size, such as training on 90% of the original training set samples, the model accuracy improvement becomes marginal and gradually slows down.

Conclusion. EDA-AREI consistently improves performance across datasets of varying sizes. The efficacy of EDA-AREI increases with larger datasets.

E. Analyzing the Types and Quantity of Samples That Are Predicted Incorrectly by EDA-AREI and Baselines

Motivation. Although the quantitative experimental results of EDA-AREI are not significantly better than the state-of-the-art baseline, the removal of noise data and the enhancement of the model's robustness are crucial. Therefore, in this study, we adopt a qualitative perspective to analyze the types and number of incorrect prediction samples estimated by each model, specifically focusing on the denoising component and data augmentation component. Our analysis demonstrates the importance of these components in predicting related event pairs.

Method. We counted the number of false positives (FP) and false negatives (FN) predicted by each model (i.e. SVM [18], USE+SVM [19], Bi-LSTM, and EDA-AREI) in order to analyze the type and the number of incorrect prediction samples. The models estimated 17,020 action-problem pairs in the test dataset and labeled them as positive or negative. We use *TOTAL* to present the number of event pairs in test dataset, i.e. 17,020. To present the experimental results visually, we used *FPR*, which is the proportion of event pairs incorrectly predicted as positive among all event pairs in the test dataset, to evaluate the models. *FPR* is expressed as:

$$FPR = \frac{FP}{TOTAL}. \tag{15}$$

Similarly, *FNR* represents the proportion of event pairs incorrectly predicted as negative among all event pairs, which is expressed as:

$$FNR = \frac{FN}{TOTAL}. \tag{16}$$

Results. The *FPR* and *FNR* results for different models on various datasets are presented in Table VII. The experimental findings demonstrate that EDA-AREI performs well in the case of positive event pairs, as it achieves lower FNR values than FPR on all six datasets. Moreover, EDA-AREI performs better than the best baseline on four datasets, except for the k-means_2 and k-means_10 dataset, namely, 13.42%, 8.65%, 15.27%, and 15.13%, respectively, where the values are lower than the best baseline by 1.47%, 1.19%, 1.19%, and 1.69%, respectively. On

TABLE VIII
COMPARISON OF DIFFERENT COMBINATIONS OF DENOISING COMPONENT AND DATA AUGMENTATION COMPONENT IN EDA-AREI

EDA-AREI	threshold_0.5			threshold_0.25			k-means_2		
	acc	f1-score	auc	acc	f1-score	auc	acc	f1-score	auc
Denoising-augmentation	**71.19%**	**71.30%**	75.66%	79.14%	79.27%	83.55%	**69.05%**	**69.93%**	**73.37%**
Augmentation-denoising	70.76%	70.61%	**75.83%**	78.28%	78.93%	83.42%	68.58%	69.09%	72.80%
Denoising-only	70.41%	69.73%	75.72%	**79.96%**	**80.09%**	84.72%	68.70%	69.47%	73.24%
Augmentation-only	69.49%	69.77%	75.09%	77.72%	78.91%	83.66%	66.92%	68.79%	71.47%
Bi-LSTM	69.10%	67.28%	74.51%	79.60%	79.62%	**85.03%**	67.80%	68.24%	72.97%
	k-means_10			k-means_100			random		
	acc	f1-score	auc	acc	f1-score	auc	acc	f1-score	auc
Denoising-augmentation	**69.02%**	68.68%	72.58%	**68.24%**	68.13%	72.14%	**68.48%**	**68.76%**	71.98%
Augmentation-denoising	68.65%	**69.23%**	73.11%	68.11%	68.97%	72.46%	67.56%	67.48%	71.65%
Denoising-only	68.69%	68.86%	**73.21%**	68.01%	68.07%	72.68%	67.37%	68.44%	71.79%
Augmentation-only	67.27%	68.17%	71.86%	66.84%	68.21%	71.29%	66.48%	67.07%	70.80%
Bi-LSTM	67.48%	69.02%	73.20%	67.41%	**69.64%**	**72.63%**	67.20%	66.22%	**72.03%**

the k-means_2 dataset, the *FNR* value of EDA-AREI is higher than that of SVM, indicating that SVM identifies more related event pairs but unfortunately predicts many non-related event pairs as related ones, with a really high *FPR* value. On the k-means_10 dataset, there is little difference between EDA-AREI and the best baseline. Therefore, it can be said that EDA-AREI achieves high precision in predicting positive action-event pairs. *FPR* is used to assess the proportion of event pairs incorrectly predicted as positive among all event pairs. For this metric, EDA-AREI only performs best on the k-means_10 dataset, with an *FPR* value of 15.88%, while Bi-LSTM achieves the best performance on four of the remaining five datasets. The reason why EDA-AREI predicts positive action-problem event pairs accurately, but not when dealing with negative pairs, may be that the denoising component removes more noise data with the true label of negative than positive, where negative sample strategies mainly introduce negative noise instances, which affects the knowledge learned by EDA-AREI, leading it to rely more on positive event pairs than negative ones. Additionally, the data augmentation component exacerbates this process, resulting in EDA-AREI being better at handling related samples. These experimental results stimulate a possible idea for improvement. We propose flipping the labels of noise data identified by the denoising component, rather than removing them. This approach balances the datasets after the denoising process and increases the size and diversity of the post-processed datasets, which would be a better strategy to train deep learning models.

Conclusion. EDA-AREI has shown effectiveness in identifying related event pairs in the case of positive pairs, outperforming comparative approaches significantly. However, its performance in identifying related event pairs in negative pairs is suboptimal.

F. The Impact of Separate Denoising or Data Augmentation on EDA-AREI

Motivation. As demonstrated in RQ1, the precision of EDA-AREI in predicting event pair relationships is enhanced by leveraging denoising and data augmentation techniques. The denoising component removes annotation and representation noise from the original datasets, while the data augmentation component enriches the diversity of problem text representations. In this RQ, we conduct an analysis to investigate the individual effects of the denoising and data augmentation components on the performance of EDA-AREI.

Method. We assess the efficacy of EDA-AREI by altering the combination of denoising and data augmentation components to Denoising-only, Augmentation-only, Denoising-augmentation (i.e., the current strategy of EDA-AREI), and Augmentation-denoising. These four combinations encompass all possible ways in which denoising and data augmentation components can impact EDA-AREI. We employ the metrics discussed in Section IV-C.

Results. The results of *Accuracy*, *F1-score*, and *AUC* for different combinations of the denoising component and data augmentation component are presented in Table VIII. The experimental results indicate that the models incorporating both denoising and augmentation components simultaneously achieve the best performance on most datasets in terms of the metrics of *Accuracy*, *F1-score*, and *AUC*. For example, on the threshold_0.5 dataset, the three metric values of the Denoising-augmentation strategy are 71.19%, 71.30%, and 75.66%, respectively, which outperform the Denoising-only, Augmentation-only, and Bi-LSTM strategies, exceeding the performance of the original Bi-LSTM model by 2.09%, 4.02%, and 1.15%, respectively. The same trend is observed for the Augmentation-denoising strategy. The denoising component reduces noisy data that interferes with EDA-AREI's ability to learn the correct relationship between action and problem events, while the data augmentation component improves the diversity and richness of the application problem text representation, providing more available textual features for EDA-AREI. With more textual features representing the correct relationship between action and problem events, EDA-AREI achieves the greatest performance improvement. Furthermore, the experimental results show that the denoising component usually promotes the improvement of the Bi-LSTM model separately, but the augmentation component does not. For instance, on the k-means_2 dataset, the Denoising-only model achieves

TABLE IX
COMPARISON OF DIFFERENT HIDDEN LAYERS IN EDA-AREI ON ACCURACY, F1-SCORE, AUC

	threshold_0.5			threshold_0.25			k-means_2		
Hiddenlayers	acc	f1-score	auc	acc	f1-score	auc	acc	f1-score	auc
64	69.56%	71.07%	74.45%	77.58%	78.73%	82.59%	67.50%	68.86%	71.42%
128	70.48%	71.28%	75.30%	78.97%	79.11%	**83.67%**	68.01%	67.57%	72.40%
256	71.19%	71.30%	75.66%	**79.14%**	79.27%	83.55%	**69.05%**	**69.93%**	**73.37%**
512	**71.23%**	**71.84%**	**75.57%**	78.61%	**79.34%**	82.93%	68.69%	69.08%	72.97%
	k-means_10			k-means_100			random		
	acc	f1-score	auc	acc	f1-score	auc	acc	f1-score	auc
64	67.70%	68.12%	72.11%	66.77%	67.46%	70.69%	66.51%	66.74%	70.59%
128	68.14%	67.91%	72.22%	67.95%	68.43%	72.16%	67.30%	67.52%	71.51%
256	**69.02%**	68.68%	72.58%	68.24%	68.13%	72.14%	**68.48%**	**68.76%**	71.98%
512	68.94%	**69.48%**	**73.20%**	**68.50%**	**68.86%**	**72.55%**	68.19%	68.68%	**72.09%**

68.70%, 69.47%, and 73.24%, respectively, outperforming Bi-LSTM on the three metrics by 0.90%, 1.23%, and 0.27%, while the Augmentation-only model is poorer than Bi-LSTM on *Accuracy* and *AUC*. The denoising component works because we have identified the correct noise data by the technique mentioned in Section III-B. The Denoising-only strategy improves performance by correctly removing noise data. The failure of the Augmentation-only strategy is also attributed to noise data. The data augmentation component improves not only the textual diversity of labeled correct instances but also the noise data, which exacerbates the damage of noise data to the model. Based on the experimental analysis, we conclude that the performance of the model can only be improved to the maximum extent by combining both denoising and data augmentation components simultaneously. Therefore, we use the Denoising-augmentation strategy in EDA-AREI. The *Precision* and *Recall* experimental results are presented in the Appendix.

Conclusion. The results suggest that the separate denoising component improves the performance of EDA-AREI, while the augmentation component does not. The optimal performance is achieved when both denoising and data augmentation techniques are used together in EDA-AREI.

G. The Impact of Different Parameters on the Accuracy of EDA-AREI

Motivation. As discussed in Section IV-D, EDA-AREI involves several parameters. EDA-AREI is based on a Bi-LSTM model; hence, the number of hidden layers affects the complexity of the Bi-LSTM model, which may further affect the model's ability to extract textual information. The dropout rate determines the number of neurons that fail, and is used to mitigate overfitting during model learning. A moderate dropout rate can improve performance, whereas an excessive value can hinder learning and result in poor performance. The Data Augmentation component is a crucial aspect of the EDA-AREI model, and the SR ratio determines the number of words replaced in each sentence by the synonym replacement operation, which can greatly affect the diversity of language expression in the enhanced datasets. In this RQ, we analyze the impact of the number of hidden layers, dropout rate, and SR ratio on event inference prediction, and attempt to identify the optimal values for these three parameters.

Method. Regarding the hidden layers, we evaluated the effectiveness of EDA-AREI by setting the number of hidden layers as 64, 128, 256, and 512, respectively. We did not evaluate EDA-AREI with a larger number of hidden layers for two reasons. Firstly, as the number of hidden layers increases, the complexity of EDA-AREI and the training time of the model significantly increase. Secondly, using a large number of hidden layers can cause serious overfitting, resulting in poor learning and inaccurate prediction of action-problem pairs. In terms of the dropout rate, we evaluated the effectiveness of EDA-AREI by varying the value of the dropout rate from 0.1 to 0.5, with a step of 0.1. Finally, we set the SR ratio as 0.1, 0.2, 0.3, 0.4, and 0.5, respectively.

Results. Table IX presents the results of *Accuracy*, *F1-score*, and *AUC* for different numbers of hidden layers. The table shows that EDA-AREI achieves the best performance when the number of hidden layers is set to 256 or 512. For the k-means_2 dataset, the setting of 256 hidden layers outperforms that of 512 in terms of the three metrics. For the threshold_0.5, threshold_0.25, k-means_10, and random datasets, the two models have little differences in terms of *Accuracy*, *F1-score*, and *AUC*. On the k-means_100 dataset, hidden layers of 512 outperform those of 256 in terms of the three metrics. Hidden layers of 64 and 128 do not achieve sufficiently good results, as the number of neurons is too small to fully extract textual information from natural sentences. Moreover, increasing the number of hidden layers significantly increases the consumption of computing power and training time. Therefore, we consider the optimal number of hidden layers to be 256 in our experiments.

Table X shows the results of different dropout rates, indicating that the dropout rate is not the main factor affecting the performance of EDA-AREI. A higher dropout rate leads to only a slight improvement. The moderate value of 0.3 achieves relatively good performance, with the best *Accuracy* on threshold_0.5, threshold_0.25, and random datasets. In terms of *F1-score*, dropout 0.3 outperforms the other settings on threshold_0.5, k-means_2, and random datasets by 0.05% to 0.47%, and 0.53% to 1.15%, respectively. In terms of *AUC*, dropout 0.3 is slightly better than the other settings on threshold_0.25

TABLE X
COMPARISON OF DIFFERENT DROPOUT RATE IN EDA-AREI ON ACCURACY, F1-SCORE, AUC

	threshold_0.5			threshold_0.25			k-means_2		
Dropout	acc	f1-score	auc	acc	f1-score	auc	acc	f1-score	auc
0.1	70.91%	71.25%	75.72%	78.75%	79.71%	83.08%	68.68%	68.80%	72.52%
0.2	70.81%	71.24%	75.57%	78.46%	**79.67%**	83.08%	68.48%	69.40%	72.57%
0.3	**71.19%**	**71.30%**	**75.66%**	**79.14%**	79.27%	**83.55%**	**69.05%**	**69.93%**	**73.37%**
0.4	71.03%	71.13%	75.45%	78.41%	79.47%	83.14%	68.34%	68.78%	72.52%
0.5	70.48%	70.83%	75.64%	78.61%	79.91%	83.42%	68.88%	69.08%	73.37%
	k-means_10			k-means_100			Random		
	acc	f1-score	auc	acc	f1-score	auc	acc	f1-score	auc
0.1	68.79%	69.19%	72.75%	68.02%	68.45%	72.26%	68.01%	68.51%	**72.36%**
0.2	68.73%	**69.20%**	72.68%	**68.61%**	68.55%	72.60%	68.17%	68.67%	71.90%
0.3	**69.02%**	68.68%	72.58%	68.24%	68.13%	72.14%	**68.48%**	**68.76%**	71.98%
0.4	68.99%	68.70%	72.69%	68.14%	68.49%	72.46%	67.62%	67.12S	72.06%
0.5	68.74%	69.08%	**73.11%**	68.42%	**68.81%**	**72.92%**	67.52%	68.30%	72.00%

TABLE XI
COMPARISON OF DIFFERENT SR RATIO IN DATA AUGMENTATION COMPONENT IN TERMS OF ACCURACY, F1-SCORE, AUC

	threshold_0.5			threshold_0.25			k-means_2		
SR ratio	acc	f1-score	auc	acc	f1-score	auc	acc	f1-score	auc
0.1	**71.19%**	71.30%	75.66%	**79.14%**	79.27%	83.55%	**69.05%**	**69.93%**	**73.37%**
0.2	71.04%	71.30%	**75.74%**	78.78%	79.29%	83.24%	68.68%	69.23%	72.95%
0.3	70.84%	**71.83%**	75.56%	78.64%	79.37%	**83.66%**	68.44%	68.30%	72.41%
0.4	70.49%	71.43%	75.33%	78.57%	**79.52%**	83.29%	68.25%	68.30%	72.64%
0.5	70.53%	71.34%	75.63%	78.52%	79.37%	83.45%	68.20%	68.57%	72.56%
	k-means_10			k-means_100			random		
	acc	f1-score	auc	acc	f1-score	auc	acc	f1-score	auc
0.1	**69.02%**	68.68%	72.58%	68.24%	68.13%	72.14%	**68.48%**	**68.76%**	**71.98%**
0.2	68.15%	69.06%	72.64%	68.32%	68.58%	72.41%	67.72%	68.34%	71.89%
0.3	68.89%	69.75%	**73.03%**	68.22%	**69.77%**	72.19%	67.80%	68.19%	71.95%
0.4	68.62%	**69.85%**	72.71%	**68.50%**	69.21%	**72.60%**	67.80%	68.23%	71.71%
0.5	68.54%	69.54%	72.74%	68.35%	68.45%	72.61%	67.36%	67.66%	71.55%

and k-means_2 datasets. Dropout is used to alleviate overfitting and improve the effectiveness of EDA-AREI to some extent. A moderate value of dropout can effectively reduce overfitting and keep the model's learning ability. Based on the experimental results, we conclude that the optimal dropout rate for EDA-AREI is 0.3.

Table XI presents the results of *Accuracy*, *F1-score*, and *AUC* for different values of the SR ratio. The experimental results indicate that a larger value of the SR ratio does not improve the accuracy of EDA-AREI. The value of 0.1 achieves the best performance in terms of *Accuracy* and outperforms the other settings on five datasets, falling short only on the k-means_100 dataset by less than 0.5%. Regarding *F1-score* and *AUC*, the settings from 0.1 to 0.5 have almost the same probability of being the best. The SR ratio determines the number of words replaced in each sentence. As the SR ratio increases, more words are replaced. However, if too many words are replaced, the semantic meaning of the sentence may change significantly, altering the relationship between the action event and the problem event. Based on the experimental results, we have determined the optimal SR ratio value to be 0.1 in our experiments. The *Precision* and *Recall* results are presented in the Appendix.

Conclusion. EDA-AREI achieves the best performance when the number of hidden layers was set to 256, the dropout rate was set to 0.3, and the SR ratio was set to 0.1.

H. How Does EDA-AREI Effectively Predict the Relationship of Event Pairs Compared With Bi-LSTM in the Statistical Context?

Motivation. The apparent absence of a substantial enhancement in terms of Accuracy by EDA-AREI raises a valid concern that demands thorough exploration. To substantiate the efficacy of EDA-AREI in distinguishing pertinent action-event pairs in comparison to Bi-LSTM, a comprehensive statistical analysis has been undertaken.

Method. To compare the performance of EDA-AREI and Bi-LSTM, we conducted a total of six training runs for each model on the threshold_0.5 dataset. We collected three evaluation metrics, namely *Accuracy*, *F1-score*, and *AUC*, resulting in 12 values for each metric. Due to time constraints, our focus was

TABLE XII
THE STATISTICAL ANALYSIS OF PERFORMANCE BETWEEN EDA-AREI AND BI-LSTM IN TERMS OF ACCURACY, F1-SCORE, AND AUC

Models	Accuracy			
	AVG	Variance	P-value	Effect-size
EDA-AREI	**70.85%**	**0.03**		
Bi-LSTM	69.84%	0.39	0.0034	0.7400
	F1-score			
	AVG	Variance	P-value	Effect-size
EDA-AREI	**71.38%**	**0.10**		
Bi-LSTM	69.71%	2.57	0.0310	0.5865
	AUC			
	AVG	Variance	P-value	Effect-size
EDA-AREI	75.53%	**0.09**		
Bi-LSTM	75.53%	0.29	0.9949	0.0019

on the threshold_0.5 dataset, which exhibits a moderate noise level, making it more representative of real-world conditions. We performed both univariate analysis of variance and effect size measurements between EDA-AREI and Bi-LSTM using the collected data from the threshold_0.5 dataset. We used a 95% confidence interval for *Accuracy*, *F1-score*, and *AUC* to ensure the statistical significance of our findings. Univariate analysis of variance is a statistical method that employs F-tests to analyze differences between two groups of samples. From the analysis, we obtained three key metrics, *AVG*, *Variance*, and *P-value*, which are used to assess the differences between EDA-AREI and Bi-LSTM and to determine the superior model. *AVG* represents the average performance effectiveness of the model in terms of *Accuracy*, *F1-score*, or *AUC*. *Variance* measures the stability within a set of data, where lower *Variance* indicates higher stability. *P-value* is a quantitative measure of the differences between the two groups of data. A *P-value* below 0.05 suggests a significant difference between the two models, while a *P-value* above 0.05 indicates that the differences between the two models are not significant. Effect size measurement provides a quantitative assessment of the factual differences between the two models. When the value of *Effect-size* is greater than 0.5, it indicates significant practical differences between the models. A value higher than 0.3 indicates moderate performance differences, and a value just higher than 0.1 suggests slight differences between the two models. These effect size measurements allow us to better understand the magnitude of the distinctions between EDA-AREI and Bi-LSTM. By conducting these comprehensive statistical analyses, we aim to provide a robust and objective evaluation of the relative performance of EDA-AREI and Bi-LSTM on the threshold_0.5 dataset.

Results. The experimental results, shown in Table XII, demonstrate the differences between EDA-AREI and Bi-LSTM in terms of *Accuracy*, *F1-score*, and *AUC*. The *P-values* for the three metrics are 0.0034, 0.0310, and 0.9949, respectively. The *P-values* for *Accuracy* and *F1-score* are lower than 0.05, indicating significant differences in the effectiveness of EDA-AREI and Bi-LSTM in identifying relevant action-problem events. EDA-AREI exhibits stronger ability to precisely predict the relationship between action and problem events. However, the *P-value* for *AUC* is 0.9949, indicating little difference between EDA-AREI and Bi-LSTM on the *AUC* metric. EDA-AREI has the same ability as Bi-LSTM to give a higher score for positive instances. Regarding the *Effect-size* values, they are 0.7400, 0.5865, and 0.0019 for *Accuracy*, *F1-score*, and *AUC*, respectively. The *Effect-size* values for *Accuracy* and *F1-score* are both higher than 0.5, indicating a significant difference between EDA-AREI and the state-of-the-art method. EDA-AREI excels in practically identifying application problems in response to user actions. The *AVG* metric shows that EDA-AREI outperforms Bi-LSTM by 1.01% and 1.67% on *Accuracy* and *F1-score*, respectively, and performs the same as Bi-LSTM on *AUC* with a value of 75.53%. These results suggest that EDA-AREI can determine the relationship between action events and problem events more accurately than Bi-LSTM. Furthermore, *Variance* is a crucial metric reflecting the stability of the models when dealing with the same instances. The results in Table XII indicate that EDA-AREI is a much more stable method compared to Bi-LSTM, achieving *Variance* values of 0.03, 0.10, and 0.09 on the three metrics, which are lower than Bi-LSTM by 0.36, 2.47, and 0.20, respectively. This implies that EDA-AREI has less bias when confronted with the same dataset, making it more effective and stable in practical applications. In practical applications, developers face massive amounts of user review information with varying language expression habits. A predictive model with high stability and strong generalization ability can better handle diverse natural language expressions and provide consistent decisions on the same application problem, understanding the user's true needs. A model with poor stability may vacillate on certain samples, predicting them sometimes as positive while sometimes as negative, which greatly affects development efficiency. Therefore, EDA-AREI's stability and effectiveness in identifying related action-problem pairs make it a better choice than Bi-LSTM in practical applications.

Conclusion. The effectiveness of EDA-AREI in identifying related action-problem pairs has shown a significant improvement compared to Bi-LSTM in terms of *Accuracy* and *F1-score*. In addition, EDA-AREI's significant improvement in stability underscores the importance of its complex structure.

I. How Does EDA-AREI Effectively Remove Noisy Data?

Motivation. To validate the denoising performance of EDA-AREI against noise in the application reviews event inference data, a manual evaluation experiment was conducted to manually analyze noisy action-problem pairs instances screened by EDA-AREI.

Method. We extracted the noise data identified by the denoising component of EDA-AREI and evaluated the quality of the denoised data through manual analysis. We invited a cohort of ten Master's and Ph.D. students specializing in computer science, who were independent of the co-authorship and possessed three to five years of experience in app development,

TABLE XIII
EXAMPLE OF LABELLING NOISE

ID	Examples	
1	Review	When I open the app a black screen pops up but nothing else shows up. I uninstalled and then reinstalled the app but the same thing happens. The app was working fine for me until yesterday
	Events	I open the app a black screen pops up
		nothing else shows up
		I uninstalled (user action)
		reinstalled the app
		the same thing happens (application problem)
2	Review	I liked this app until the latest update. What a mess. FB became slow to load or crashed. It is now gone and I am accessing FB through Safari.
	Events	I had to delete it
		I tried to download it from the cloud (user action)
		it kept timing out
		It is now gone I am accessing FB through Safari (app problem)

TABLE XIV
EXAMPLE OF CHARACTERIZING NOISE

ID	Examples	
1	Review	Facebook deciding what I want to see first is annoying. Facebook keeps force closing when I go to public Facebook pages
	Events	I go to public Facebook pages (user action)
		Nothing will work (application problem)
		Facebook keeps force closing
2	Review	I have been encountering a problem with my iPhone 5c on Facebook where when I click on certain links for articles or videos the page completely freezes and it Please fix!!!
	Events	I click on certain links for articles or videos (user action)
		However , it always sticks there (application problem)
		I have been encountering a problem with my iPhone 5c on Facebook where the page completely freezes

to participate in our survey. In this survey, we systematically selected 2000 instances of noise event pairs, evenly distributed across five distinct groups. Corresponding to each group, we formulated five distinct questionnaires, each containing 400 pairs of action-problem events. The participants were allocated into these five groups. These questionnaires were then distributed to the respective groups, with each participant in a given group independently evaluating the same questionnaire. For every entry in the questionnaire, the participants gauged the relevance of the two events and determined whether the instance was indicative of a positive or negative case. The accuracy of the denoising component was established by contrasting the responses of the five questionnaire evaluations with the original labeling of the instances in the dataset. In cases where two participants within a group expressed discordance regarding an entry in the questionnaire, the entry was reviewed by all ten participants. The final decision was determined by the majority consensus. We judged the authenticity of the noise labels and the true labels, demonstrating the effectiveness of EDA-AREI in removing the noise data.

Results. As a result, it's notable that approximately 8.5% of the noise data received differing evaluations by the two participants within a group, prompting all ten participants to review these specific questionnaires. Consequently, our investigation revealed that more than 90% of the noise data identified by EDA-AREI was detrimental to model training. The denoising performance of EDA-AREI and the results of the manual evaluation experiment are presented in Tables XIII and XIV. Four examples are provided to demonstrate the effectiveness of EDA-AREI in removing noisy data. The examples in Table XIII are instances of labelling noise. In both examples, the red-labelled events are the noisy data extracted by EDA-AREI. In the first example, five events are extracted from the original application reviews, and analysis shows that the events "I open the app a black screen pops up" and "nothing else shows up" are the actual application problems, while the noise data "the same thing happens" merely refers to the two aforementioned events without providing a specific description of the application problems. Therefore, labelling noise is present in this example, and "I uninstalled - the same thing happens" has no inferential relationship, and its true label is 0. In the second example, analysis of the four events extracted from the original application reviews shows that "it kept timing out" is the actual follow-up to "I tried to download it from the cloud," and there is an inferential relationship between them. However, the noise data "It is now gone I am accessing FB through Safari" merely describes the user's subsequent actions and not the application problem. These examples illustrate that EDA-AREI can effectively remove annotation noise from noisy data and improve the quality of the dataset.

The examples presented in Table XIV demonstrate the effectiveness of EDA-AREI in removing representational noise. Both examples have a noise label of 0, indicating the absence of an inference relationship between the user action and the application problem. The events highlighted in red correspond to the noise data, while those in green represent the correct application problem for the user action.

In the first example, the noisy data "nothing will work" and the real data "Facebook keeps force closing" express the same semantic information, i.e., the problem of the Facebook page not responding. In the second example, the noisy data "however, it always sticks there" and the real data "I have been encountering a problem with my iPhone 5c on Facebook where the page completely freezes" both convey the same meaning of the page being frozen, albeit with different phrasings. In summary, EDA-AREI effectively removes representational noise from the dataset.

Conclusion. EDA-AREI effectively addresses the limitations caused by a large amount of annotation and representation noise in the dataset, ensuring dataset quality, and improving the performance of application comment event reasoning.

VI. Threats to Validity

In this section we will analyze the threats to validity in terms of internal validity, external validity and structural validity.

A. Internal Validity

Firstly, a potential threat to internal validity is the reproducibility and accuracy of the baseline approach. When reproducing the model, we ensure that the same parameter settings as the original article are maintained, based on the provided code. If the baseline model's article does not provide code, we mitigate this threat by reproducing the model based on the work of others.

Secondly, another threat to internal validity is the performance of EDA-AREI. To reduce errors in EDA-AREI, we verified the correctness of our code. After this manuscript is accepted, we will publish our code and dataset online to facilitate replication and extension of this work by other researchers.

Thirdly, there is a potential threat that the parameters used to obtain the sample prediction probabilities in the CL using Bi-LSTM are suboptimal. Different parameter settings may affect the results of data denoising and ultimately the performance of the model. Therefore, in the future, we plan to further optimize the performance of EDA-AREI model through experimental analysis to identify the optimal parameters for the denoising component.

B. External Validity

The primary external validity threat is that our experiments were only conducted on one-star reviews from the Apple App Store, which may not be generalizable to all App review data. Moreover, although EDA-AREI is effective in addressing the event inference performance for App reviews on IOS-based systems, we have not validated App reviews from other system platforms such as Android releases. Nonetheless, we do not claim to have covered all platforms in this field, and there are many other platforms and systems that we have not considered in our research. Thus, in the future, we aim to collect more data from diverse open-source platforms and assess EDA-AREI in multiple real-life development scenarios.

C. Construct Validity

One potential threat to construct validity is the possibility that certain application problems in the dataset may occur repeatedly, or that different application actions may correspond to the same application problem. To address this, we plan to set appropriate limits on the proportion of recurring application problem data in future work. This will ensure that the dataset considers both cases where multiple application actions may generate the same application problem, as well as cases where additional sample data is added to enrich the dataset information. Another threat to construct validity is the applicability of our evaluation measures, particularly the use of the *Accuracy* evaluation metric. However, since this metric is commonly used in software engineering tasks, we believe that it poses little threat to construct validity in our study.

VII. Related Work

A. Event Inference

Application reviews provide insight into the user's interaction with an application and their overall experience. From a natural language processing perspective, these reviews can be viewed as a sequence of events. The study of event inference involves understanding the relationships between events. Previous research has focused on extracting temporal relationships between events. For example, Mani et al. [21] inferred temporal relationships between events using rules and axioms that incorporate temporal markers like "before" and "after". Mirroshandel and Ghassem-Sani [22] extracted temporal relationships from news articles by using basic features of events such as tense, polarity, and mood, as well as additional features such as prepositional phrases. Ning et al. [23] extracted temporal relations by utilizing a knowledge base of temporal relations collected from other textual sources such as news articles.

Numerous research endeavors have aimed to extract causal relationships from the temporal order of events. Beamer and Girju [24] introduced the notion of causal potential to gauge the strength of causal relationships between pairs of events. The causal relationship between two events is stronger when they occur in a particular order than in the opposite order. Hu and Walker [25] extracted the temporal relationships of actions from movie scripts and inferred their causal relationships. Hu et al. [26] extracted and inferred fine-grained pairs of events that were causally related from blogs and movie descriptions. Zellers et al. [27] provided a dataset of multiple-choice problems consisting of event pairs extracted from video subtitles: SWAG. The primary task is to select the second event from four choices given the first event, based on common sense reasoning. The Story Completion Quiz [28] also utilized an inference model to determine whether the last event is the end of the first four events, based on a collection of five events extracted from a personal blog.

With the widespread adoption of deep learning techniques in the field of natural language processing, LSTM networks have emerged as a special kind of recurrent neural network that can handle long sequences of data with superior performance on textual data. Srinivasan et al. [29] targeted the story cloze test and used a Bi-LSTM model to determine whether an event was random or a correct story ending. Building upon their work, Guo et al. [13] constructed and trained an "action-problem" event pair inference model to determine whether there was a relationship between an unknown application problem and a known action. They mainly achieved two baseline tasks, SVM [18] and USE+SVM [19]. SVM used TF-IDF [20] to convert each event into a vector, concatenated the vectors of two events into an event pair, and trained an SVM classifier on the concatenated vector. USE+SVM changed the vector

conversion method by using USE to convert each event into a vector.

In contrast to previous studies, we adopt a dataset quality perspective by removing a substantial amount of labeling and characterizing noise through confidence learning. Furthermore, we employ synonym substitution to enhance the data and enrich it with diverse representations, resulting in the generation of a high-quality dataset of application reviews.

B. Noise-Robust Learning

In supervised learning using neural networks, the quality of training data labels is critical for achieving good learning outcomes. If the label data used for learning is incorrect, it is impossible to train an effective predictive model. Currently, methods for learning from noisy label data are mainly divided into two categories: one is to directly train a model that is robust to noise, and the other is to first identify noisy data and then train the model based on cleaned data.

In recent years, several noise-robust models have been proposed. Patrini et al. [30] proposed a loss correction method for training deep neural networks with noisy labels. Shu et al. [31] introduced a weighting function that is learned automatically from data, with the function parameters continuously updated during classifier training. Han et al. [32] proposed a "Co-teaching" approach, which trains two deep neural networks simultaneously and each network feeds its own small loss instances into the other network to achieve cross-training. Wang et al. [33] identified the problem of category bias in noisy data learning with cross-entropy loss functions, and proposed a symmetric cross-entropy learning method based on the commonly used cross-entropy loss function to address this problem.

However, these methods all rely on introducing novel models or modifying the loss function during the deep learning training process. This model-centric approach does not address the problem of mislabeling within the dataset itself. Therefore, efforts have emerged to identify noisy data. Mariya et al. [34] proposed a method to identify mislabeling based on forgotten events. During model training, a sample may be correctly classified by the model, but as the model parameters are updated, the sample is misclassified again. This process is called a forgetting event for the sample. Studies have shown that noisy samples tend to experience more forgetting events during model training than normal samples. Based on this heuristic rule, tracking the total number of forgetting events experienced by each sample can help identify possible noisy labeled data. Huang et al. [35] proposed a method to identify mislabeling based on the loss value of samples during training. By observing the training process, it is reported that from underfitting to overfitting, the mean and variance of the loss values of noisy samples were larger than those of clean samples. Therefore, the mean and variance of the individual sample loss values were computed, and the larger the mean and variance, the greater the probability that the sample belongs to a noisy sample. In the work of Northcutt et al. [14], following a data-centric approach, they proposed the Confident Learning (CL) framework, which focuses on label quality and identifies mislabeled data by estimating the joint distribution of noisy and true labels, as well as ranking and pruning principles.

Diverging from previous studies, our approach begins with the dataset itself, employing Bi-LSTM networks and CL to eliminate noisy data.

C. Data Augmentation

Data augmentation is a frequently used technique for expanding datasets. It is often used in computer vision, as operations on images do not alter their semantics and help to train more robust models. However, text data augmentation techniques have not been extensively studied due to the challenging nature of general rules for language transformation.

There have been several studies on data augmentation techniques in NLP. Zhang et al. [36] randomly selected a word from a sentence and replaced it with a synonym using a thesaurus dictionary. Mueller et al. [37] used a similar strategy to generate an additional 10K training samples for their sentence similarity model. Wei et al. [15] also employed the synonym replacement technique as one of the four random augmentation sets in their model.

Wang et al. [38] proposed a word vector-based replacement technique, utilizing pre-trained word embeddings to substitute some words in a sentence with the nearest adjacent words in the embedding space. Jiao et al. [39] also employed the word vector replacement technique to enhance the generalization ability of their language models in downstream tasks. Wei et al. [15] proposed a set of general data augmentation techniques, EDA, which included methods such as synonym replacement, random deletion, random insertion, and random swap to augment the original sentences. Xie et al. [40] applied data augmentation from supervised learning to semi-supervised learning, augmenting training sentences through back-translation and TF-IDF based word replacement. Coulombe et al. [41] augmented textual data through pattern matching transformations applied by regular expressions, converting words from short to full forms. Additionally, a method of manipulating the syntax tree was proposed, which converted the dependent syntax tree parsed into the original sentence by certain rules to generate a paraphrase sentence. Guo et al. [42] proposed using Mixup for data augmentation, which combined the word embeddings of two sentences proportionally to generate new embeddings passed to the downstream text classification task. Xie et al. [43] proposed using placeholder tokens to randomly replace words. Luque et al. [44] proposed instance cross augmentation, which generated new text by randomly swapping half of the tweets with the same polarity, maintaining the new text polarity. Kumar et al. [45] employed the Transformer model to augment the training data, splicing the labels and data into the Transformer training, and then used the fine-tuning task to generate new samples.

In this article, we utilize natural language processing data augmentation techniques to perform synonym replacement on problem events in action-problem event pairs, thereby generating new problems that preserve the original semantic information. By doing so, we can effectively expand the dataset and increase the diversity and richness of the data samples.

VIII. Conclusions and Future Work

This study empirically investigates the benefits of using EDA-AREI for event inference in App reviews, which comprises a denoising component, a data augmentation component, and an event inference prediction component. The denoising component utilizes Bi-LSTM and confidence learning techniques to improve the quality of the App review event inference dataset by removing noisy data. The data augmentation component then performs synonym substitution on the application problem events in each event pair to increase the variety of text representation for data augmentation purposes. Finally, the event inference prediction component performs inference prediction of labels based on the denoised and enhanced data. The comprehensive experimental results demonstrate that EDA-AREI outperforms the baseline models in terms of both *Accuracy* evaluation metrics. Compared to the Bi-LSTM baseline, EDA-AREI achieves an improvement of 2.09% on the threshold_0.5 dataset, 1.25%, 1.54%, and 0.83% on the k-means_2, k-means_10, and k-means_100 datasets, respectively, and 1.28% on the random dataset.

In future work, we plan to impose constraints on the proportion of repetitive application problem data to prevent redundancy caused by duplicated data. Moreover, we intend to apply our approach to event inference for application reviews on different system platforms to enhance the generality of EDA-AREI.

References

[1] D. Pagano and W. Maalej, "User feedback in the appstore: An empirical study," in *Proc. 21st IEEE Int. Requirements Eng. Conf. (RE)*, Rio de Janeiro, Brazil. Los Alamitos, CA, USA: IEEE Comput. Soc., Jul. 15–19, 2013, pp. 125–134, doi: 10.1109/RE.2013.6636712.

[2] E. Guzman and W. Maalej, "How do users like this feature? A fine grained sentiment analysis of app reviews," in *Proc. IEEE 22nd Int. Requirements Eng. Conf. (RE)*, Karlskrona, Sweden, T. Gorschek and R. R. Lutz, Eds. Los Alamitos, CA, USA: IEEE Comput. Soc., Aug. 25–29, 2014, pp. 153–162, doi: 10.1109/RE.2014.6912257.

[3] "SwiftKey feedback forums." Bugzilla. Accessed: Aug. 24, 2021. [Online]. Available: http://www.bugzilla.org/

[4] A. Bacchelli, T. D. Sasso, M. D'Ambros, and M. Lanza, "Content classification of development emails," in *Proc. 34th Int. Conf. Softw. Eng. (ICSE)*, Zurich, Switzerland, M. Glinz, G. C. Murphy, and M. Pezzè, Eds. Los Alamitos, CA, USA: IEEE Comput. Soc., Jun. 2–9, 2012, pp. 375–385, doi: 10.1109/ICSE.2012.6227177.

[5] Bugzilla. Accessed: Sep. 25, 2023. [Online]. Available: https://www.bugzilla.org/

[6] Y. Dang, R. Wu, H. Zhang, D. Zhang, and P. Nobel, "Rebucket: A method for clustering duplicate crash reports based on call stack similarity," in *Proc. 34th Int. Conf. Softw. Eng. (ICSE)*, Zurich, Switzerland, M. Glinz, G. C. Murphy, and M. Pezzè, Eds. Los Alamitos, CA, USA: IEEE Comput. Soc., Jun. 2–9, 2012, pp. 1084–1093, doi: 10.1109/ICSE.2012.6227111.

[7] L. V. G. Carreño and K. Winbladh, "Analysis of user comments: An approach for software requirements evolution," in *Proc. 35th Int. Conf. Softw. Eng. (ICSE)*, San Francisco, CA, USA, D. Notkin, B. H. C. Cheng, and K. Pohl, Eds. Los Alamitos, CA, USA: IEEE Comput. Soc., May 18–26, 2013, pp. 582–591, doi: 10.1109/ICSE.2013.6606604.

[8] N. Chen, J. Lin, S. C. H. Hoi, X. Xiao, and B. Zhang, "AR-miner: Mining informative reviews for developers from mobile app marketplace," in *Proc. 36th Int. Conf. Softw. Eng. (ICSE)*, Hyderabad, India, P. Jalote, L. C. Briand, and A. van der Hoek, Eds. New York, NY, USA: ACM, May 31–Jun. 7, 2014, pp. 767–778, doi: 10.1145/2568225.2568263.

[9] A. D. Sorbo et al., "What would users change in my app? Summarizing app reviews for recommending software changes," in *Proc. 24th ACM SIGSOFT Int. Symp. Found. Softw. Eng. (FSE)*, Seattle, WA, USA, T. Zimmermann, J. Cleland-Huang, and Z. Su, Eds. New York, NY, USA: ACM, Nov. 13–18, 2016, pp. 499–510, doi: 10.1145/2950290.2950299.

[10] Z. Kurtanovic and W. Maalej, "Mining user rationale from software reviews," in *Proc. 25th IEEE Int. Requirements Eng. Conf. (RE)*, Lisbon, Portugal, A. Moreira, J. Araújo, J. Hayes, and B. Paech, Eds. Los Alamitos, CA, USA: IEEE Comput. Soc., Sep. 4–8, 2017, pp. 61–70, doi: 10.1109/RE.2017.86.

[11] W. Maalej and H. Nabil, "Bug report, feature request, or simply praise? On automatically classifying app reviews," in *Proc. 23rd IEEE Int. Requirements Eng. Conf. (RE)*, Ottawa, ON, Canada, D. Zowghi, V. Gervasi, and D. Amyot, Eds. Los Alamitos, CA, USA: IEEE Comput. Soc., Aug. 24–28, 2015, pp. 116–125, doi: 10.1109/RE.2015.7320414.

[12] S. Panichella, A. D. Sorbo, E. Guzman, C. A. Visaggio, G. Canfora, and H. C. Gall, "How can I improve my app? Classifying user reviews for software maintenance and evolution," in *Proc. IEEE Int. Conf. Softw. Maintenance Evolution (ICSME)*, Bremen, Germany, R. Koschke, J. Krinke, and M. P. Robillard, Eds. Los Alamitos, CA, USA: IEEE Comput. Soc., Sep. 29–Oct. 1, 2015, pp. 281–290, doi: 10.1109/ICSM.2015.7332474.

[13] H. Guo and M. P. Singh, "Caspar: Extracting and synthesizing user stories of problems from app reviews," in *Proc. 42nd Int. Conf. Softw. Eng. (ICSE)*, Seoul, South Korea, G. Rothermel and D. Bae, Eds. New York, NY, USA: ACM, Jun. 27–Jul. 19, 2020, pp. 628–640, doi: 10.1145/3377811.3380924.

[14] C. G. Northcutt, L. Jiang, and I. L. Chuang, "Confident learning: Estimating uncertainty in dataset labels," *J. Artif. Intell. Res.*, vol. 70, pp. 1373–1411, 2021, doi: 10.1613/jair.1.12125.

[15] J. W. Wei and K. Zou, "EDA: Easy data augmentation techniques for boosting performance on text classification tasks," in *Proc. Conf. Empirical Methods Natural Lang. Process. 9th Int. Joint Conf. Natural Lang. Process. (EMNLP-IJCNLP)*, Hong Kong, China, K. Inui, J. Jiang, V. Ng, and X. Wan, Eds. New York, NY, USA: ACM, Nov. 3–7, 2019, pp. 6381–6387, doi: 10.18653/v1/D19-1670.

[16] Spacy. Accessed: Dec. 20, 2021. [Online]. Available: https://github.com/explosion/spacy-models/releases?q=en_core_web_lg&expanded=true

[17] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997, doi: 10.1162/neco.1997.9.8.1735.

[18] S. Russell and P. Norvig, *Artificial Intelligence - A Modern Approach*, 2nd ed. Englewood Cliffs, NJ, USA: Prentice-Hall, 2003. [Online]. Available: https://www.worldcat.org/oclc/314283679

[19] D. Cer et al., "Universal sentence encoder," 2018. [Online]. Available: http://arxiv.org/abs/1803.11175

[20] G. Salton and M. McGill, *Introduction to Modern Information Retrieval*. New York, NY, USA: McGraw-Hill, 1984.

[21] I. Mani, M. Verhagen, B. Wellner, C. M. Lee, and J. Pustejovsky, "Machine learning of temporal relations," in *Proc. 21st Int. Conf. Comput. Linguistics 44th Annu. Meeting Assoc. Comput. Linguistics (ACL)*, Sydney, NSW, Australia, N. Calzolari, C. Cardie, and P. Isabelle, Eds. New York, NY, USA: ACM, Jul. 17–21, 2006. [Online]. Available: https://aclanthology.org/P06-1095/

[22] S. A. Mirroshandel and G. Ghassem-Sani, "Towards unsupervised learning of temporal relations between events," *J. Artif. Intell. Res.*, vol. 45, no. 1, pp. 125–163, 2012, doi: 10.1613/jair.3693.

[23] Q. Ning, H. Wu, H. Peng, and D. Roth, "Improving temporal relation extraction with a globally acquired statistical resource," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol. (NAACL-HLT)*, New Orleans, LA, USA, June 1–6, 2018, Volume 1 (Long Papers), M. A. Walker, H. Ji, and A. Stent, Eds. New York, NY, USA: ACM, Jun. 1–6, 2018, pp. 841–851, doi: 10.18653/v1/n18-1077.

[24] B. Beamer and R. Girju, "Using a bigram event model to predict causal potential," in *Proc. Comput. Linguistics Intell. Text Process., 10th Int. Conf. (CICLing)*, Mexico City, Mexico, A. F. Gelbukh, Ed., vol. 5449. Berlin, German: Springer, Mar. 1–7, 2009, pp. 430–441, doi: 10.1007/978-3-642-00382-0_35.

[25] Z. Hu and M. A. Walker, "Inferring narrative causality between event pairs in films," in *Proc. 18th Annu. SIGdial Meeting Discourse Dialogue*, Saarbrücken, Germany, K. Jokinen, M. Stede, D. DeVault, and A. Louis, Eds. New York, NY, USA: ACM, Aug. 15–17, 2017, pp. 342–351, doi: 10.18653/v1/w17-5540.

[26] Z. Hu, E. Rahimtoroghi, and M. A. Walker, "Inference of fine-grained event causality from blogs and films," in *Proc. Events Stories News Workshop@ACL*, Vancouver, BC, Canada, T. Caselli et al., Eds. New York, NY, USA: ACM, Aug. 4, 2017, pp. 52–58, doi: 10.18653/v1/w17-2708.

[27] R. Zellers, Y. Bisk, R. Schwartz, and Y. Choi, "SWAG: A large-scale adversarial dataset for grounded commonsense inference," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, Brussels, Belgium, E. Riloff, D. Chiang, J. Hockenmaier, and J. Tsujii, Eds. New York, NY, USA: ACM, Oct. 31–Nov. 4, 2018, pp. 93–104, doi: 10.18653/v1/d18-1009.

[28] N. Mostafazadeh et al., "A corpus and cloze evaluation for deeper understanding of commonsense stories," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol. (NAACL HLT)*, San Diego, CA, USA, K. Knight, A. Nenkova, and O. Rambow, Eds. New York, NY, USA: ACM, Jun. 12–17, 2016, pp. 839–849, doi: 10.18653/v1/n16-1098.

[29] S. Srinivasan, R. Arora, and M. O. Riedl, "A simple and effective approach to the story cloze test," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol. (NAACL-HLT)*, New Orleans, LA, USA, vol. 2, M. A. Walker, H. Ji, and A. Stent, Eds. New York, NY, USA: ACM, Jun. 1–6, 2018, pp. 92–96, doi: 10.18653/v1/n18-2015.

[30] G. Patrini, A. Rozza, A. K. Menon, R. Nock, and L. Qu, "Making deep neural networks robust to label noise: A loss correction approach," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit. (CVPR)*, Honolulu, HI, USA. Los Alamitos, CA, USA: IEEE Comput. Soc., Jul. 21–26, 2017, pp. 2233–2241, doi: 10.1109/CVPR.2017.240.

[31] J. Shu et al., "Meta-weight-net: Learning an explicit mapping for sample weighting," in *Proc. Adv. Neural Inf. Process. Syst. 32, Annu. Conf. Neural Inf. Process. Syst. (NeurIPS)*, Vancouver, BC, Canada, H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. B. Fox, and R. Garnett, Eds., Dec. 8–14, 2019, pp. 1917–1928. [Online]. Available: https://proceedings.neurips.cc/paper/2019/hash/e58cc5ca94270acaceed13bc82dfedf7-Abstract.html

[32] B. Han et al., "Co-teaching: Robust training of deep neural networks with extremely noisy labels," in *Proc. Adv. Neural Inf. Process. Syst. 31, Annu. Conf. Neural Inf. Process. Syst. (NeurIPS)*, Montréal, QC, Canada, S. Bengio, H. M. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds., Red Hook, NY, USA: Curran Associates Inc., Dec. 3–8, 2018, pp. 8536–8546. [Online]. Available: https://proceedings.neurips.cc/paper/2018/hash/a19744e268754fb0148b017647355b7b-Abstract.html

[33] Y. Wang, X. Ma, Z. Chen, Y. Luo, J. Yi, and J. Bailey, "Symmetric cross entropy for robust learning with noisy labels," in *Proc. IEEE/CVF Int. Conf. Comput. Vision (ICCV)*, Seoul, South Korea. Piscataway, NJ, USA: IEEE, Oct. 27–Nov. 2, 2019, pp. 322–330, doi: 10.1109/ICCV.2019.00041.

[34] M. Toneva, A. Sordoni, R. T. des Combes, A. Trischler, Y. Bengio, and G. J. Gordon, "An empirical study of example forgetting during deep neural network learning," in *Proc. 7th Int. Conf. Learn. Representations (ICLR)*, New Orleans, LA, USA, May 6–9, 2019. [Online]. Available: https://openreview.net/forum?id=BJlxm30cKm

[35] J. Huang, L. Qu, R. Jia, and B. Zhao, "O2u-net: A simple noisy label detection approach for deep neural networks," in *Proc. IEEE/CVF Int. Conf. Comput. Vision (ICCV)*, Seoul, South Korea. Piscataway, NJ, USA: IEEE, Nov. 2, 2019, pp. 3325–3333, doi: 10.1109/ICCV.2019.00342.

[36] X. Zhang, J. J. Zhao, and Y. LeCun, "Character-level convolutional networks for text classification," in *Proc. Adv. Neural Inf. Process. Syst. 28, Annu. Conf. Neural Inf. Process. Syst.*, Montreal, QC, Canada, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, Eds., Cambridge, MA, USA: MIT Press, Dec. 7–12, 2015, pp. 649–657. [Online]. Available: https://proceedings.neurips.cc/paper/2015/hash/250cf8b51c773f3f8dc8b4be867a9a02-Abstract.html

[37] J. Mueller and A. Thyagarajan, "Siamese recurrent architectures for learning sentence similarity," in *Proc. 30th AAAI Conf. Artif. Intell.*, Phoenix, AZ, USA, D. Schuurmans and M. P. Wellman, Eds. Palo Alto, CA, USA: AAAI Press, Feb. 12–17, 2016, pp. 2786–2792. [Online]. Available: http://www.aaai.org/ocs/index.php/AAAI/AAAI16/paper/view/12195

[38] W. Y. Wang and D. Yang, "That's so annoying!!!: A lexical and frame-semantic embedding based data augmentation approach to automatic categorization of annoying behaviors using #petpeeve tweets," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, Lisbon, Portugal, L. Màrquez, C. Callison-Burch, J. Su, D. Pighin, and Y. Marton, Eds. New York, NY, USA: ACM, Sep. 17–21, 2015, pp. 2557–2563, doi: 10.18653/v1/d15-1306.

[39] X. Jiao et al., "TinyBERT: Distilling BERT for natural language understanding," in *Proc. Findings Assoc. Comput. Linguistics (EMNLP)*, T. Cohn, Y. He, and Y. Liu, Eds., vol. EMNLP,. New York, NY, USA: ACM, Nov. 16–20, 2020, pp. 4163–4174, doi: 10.18653/v1/2020.findings-emnlp.372.

[40] Q. Xie, Z. Dai, E. H. Hovy, T. Luong, and Q. Le, "Unsupervised data augmentation for consistency training," in *Proc. Adv. Neural Inf. Process. Syst. 33, Annu. Conf. Neural Inf. Process. Syst. (NeurIPS)*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., Red Hook, NY, USA: Curran Associates Inc., Dec. 6–12, 2020. [Online]. Available: https://proceedings.neurips.cc/paper/2020/hash/44feb0096faa8326192570788b38c1d1-Abstract.html

[41] C. Coulombe, "Text data augmentation made simple by leveraging NLP cloud APIS," 2018. [Online]. Available: http://arxiv.org/abs/1812.04718

[42] H. Guo, Y. Mao, and R. Zhang, "Augmenting data with mixup for sentence classification: An empirical study," 2019. [Online]. Available: http://arxiv.org/abs/1905.08941

[43] Z. Xie et al., "Data noising as smoothing in neural network language models," in *Proc. 5th Int. Conf. Learn. Representations (ICLR)*, Toulon, France, Apr. 24–26, 2017. [Online]. Available: https://openreview.net/forum?id=H1VyHY9gg

[44] F. M. Luque, "Atalaya at TASS 2019: Data augmentation and robust embeddings for sentiment analysis," in *Proc. Iberian Lang. Eval. Forum Co-located 35th Conf. Spanish Soc. Natural Lang. Process. (IberLEF@SEPLN) CEUR Workshop*, Bilbao, Spain, M. Á. G. Cumbreras et al., Eds., vol. 2421. Kigali, Rwanda: CEUR-WS.org, Sep. 24, 2019, pp. 561–570. [Online]. Available: http://ceur-ws.org/Vol-2421/TASS_paper_1.pdf

[45] V. Kumar, A. Choudhary, and E. Cho, "Data augmentation using pre-trained transformer models," 2020. [Online]. Available: https://arxiv.org/abs/2003.02245

Shikai Guo received the B.S. degree in computer science and technology from Liaoning University, Shenyang, China, in 2011, and the Ph.D. degree in computer applications technology from Dalian Maritime University, Dalian, China, in 2018. He is currently an Associate Professor with the College of Information Science and Technology, Dalian Maritime University, Dalian, China. His current research interests include intelligent software engineering, software testing, and program analysis techniques.

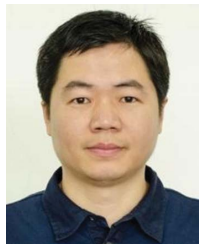
Haorui Lin is currently working toward the bachelor's degree with the School of Information Science and Technology, Dalian Maritime University, having plans to graduate in 2024 and to pursue advanced studies. His current research interests include intelligent software engineering and machine learning.

Jiaoru Zhao received the master's degree from Dalian Maritime University, in 2023. She is currently working as a Software Developer with Chongqing Changan Automobile Co. Ltd., China. Her research interests during the master's degree include intelligent software engineering, machine learning, and software testing.

Hui Li received the Ph.D. degree in computer architecture from Northeastern University, Shenyang, China in 2013. He is an Associate Professor with the School of Information Science and Technology, Dalian Maritime University. His current research interests include intelligent software engineering, mining software repositories, and recommendation system.

Rong Chen (Member, IEEE) received the M.S. and Ph.D. degrees in computer software and theory from Jilin University, Changchun, China, in 1997 and 2000, respectively. He is currently a Professor with the College of Information Science and Technology, Dalian Maritime University, Dalian, China. He was previously with Sun Yat-sen University, Guangzhou, China. His research interests include software diagnosis, collective intelligence, activity recognition, and Internet and mobile computing.

Xiaochen Li received the Ph.D. degree in software engineering from Dalian University of Technology, China, in 2019, under the supervision of Professor He Jiang. He is an Associate Professor with the School of Software, Dalian University of Technology. He was a Research Associate with the Interdisciplinary Centre for Security, Reliability and Trust (SnT), University of Luxembourg, headed by Professor Lionel Briand. His current research interests include intelligent software engineering and software testing. His work has been published at premier venues like IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, *ACM Transactions on Software Engineering and Methodology*, and International Conference on Software Engineering. For more information, please visit https://xiaochen-li.github.io.



He Jiang (Member, IEEE) received the Ph.D. degree in computer science from the University of Science and Technology of China, Hefei, China. He is currently a Professor with the Dalian University of Technology, Dalian, China. He is one of the ten supervisors for the Outstanding Doctoral Dissertation of the CCF in 2014. His current research interests include intelligent software engineering, software testing with focus on system software, and search-based software engineering (SBSE). His work has been published at premier venues like International Conference on Software Engineering, *Foundations of Software Engineering*, and ASE, as well as in major IEEE Transactions like IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, IEEE TRANSACTIONS ON CYBERNETICS, and IEEE TRANSACTIONS ON SERVICES COMPUTING.