2021 36th IEEE/ACM International Conference on Automated Software Engineering Workshops (ASEW) | 978-1-6654-3583-3/21/$31.00 ©2021 IEEE | DOI: 10.1109/ASEW52652.2021.00053

Learning Sentiment Analysis for Accessibility User Reviews

Wajdi Aljedaani*, Furqan Rustam†, Stephanie Ludi*, Ali Ouni‡, and Mohamed Wiem Mkaouer§
*University of North Texas. Email{wajdi.aljedaani, Stephanie.Ludi@unt.edu}
†KFUEIT University. Email{furqan.rustam1@gmail.com}
‡ETS Montreal, University of Quebec. Email{ali.ouni@etsmtl.ca}
§Rochester Institute of Technology. Email{mwmvse@rit.edu}

***Abstract*—Nowadays, people use different ways to express emotions and sentiments such as facial expressions, gestures, speech, and text. With the exponentially growing popularity of mobile applications (apps), accessibility apps have gained importance in recent years as it allows users with specific needs to use an app without many limitations. User reviews provide insightful information that helps for app evolution. Previously, work has been done on analyzing the accessibility in mobile applications using machine learning approaches. However, to the best of our knowledge, there is no work done using sentiment analysis approaches to understand better how users feel about accessibility in mobile apps. To address this gap, we propose a new approach on an accessibility reviews dataset, where we use two sentiment analyzers, *i.e.*, TextBlob and VADER along with Term Frequency—Inverse Document Frequency (TF-IDF) and Bag-of-words (BoW) features for detecting the sentiment polarity of accessibility app reviews. We also applied six classifiers including, Logistic Regression, Support Vector, Extra Tree, Gaussian Naive Bayes, Gradient Boosting, and Ada Boost on both sentiments analyzers. Four statistical measures namely accuracy, precision, recall, and F1-score were used for evaluation. Our experimental evaluation shows that the TextBlob approach using BoW features achieves better results with accuracy of 0.86 than the VADER approach with accuracy of 0.82.**

***Index Terms*—Mobile Applications, User Reviews, Accessibility, Sentiment Analysis, Machine Learning.**

I. INTRODUCTION

Web and mobile applications are common means of engaging with information and services. It is also crucial for these technologies to be accessible to have equal access to people with different abilities. However, in most mobile applications, there is little attention given to accessibility which results into several difficulties to appropriately utilize such applications by people with disabilities [6], [5]. Software application stores like Google Play, App Store and Amazon are available for searching and downloading mobile apps. Most of these platforms freely provide features for user reviews where users can write a review and/or give a star rating. Users' experience provides a valuable knowledge and can be studied by developers, designers, and analysts for the identification of issues in the applications with the help of user reviews [37], [13]. User reviews can be related to requests for features, troubleshooting, compliments, complaints, and dissatisfaction. Reviews can be categorized at higher levels, e.g., how good or bad a feature is. In addition, this division of the high level of app reviews can also be related to the design and usability aspects. Reviews comment on accessibility as well, i.e., the accessibility of apps to the disabled people on their mobile devices [56].

While several studies have addressed various problems related to user reviews [37], [53], [55], [44], [46], [34], user reviews related to accessibility in mobile applications are under-studied [16]. Although the growth of mobile app development is substantial, there is still a lack of mobile-based accessibility-related research, and associated guidelines as compared to web-based accessibility [47]. The significance of mobile app development raises several challenges for a deeper understanding of user reviews that are focused on accessibility concerns. Multiple challenges are associated with the study of user reviews related to accessibility. Prior work has analyzed an enormous amount of reviews and analysts with little impact on the field. For example, there is the possibility of bias in manually identifying the accessibility reviews.

The Internet has become an effective tool through which people communicate their feelings, emotions, and ideas [21]. Business analysts use this data for monitoring people's perceptions and opinions about their products. Natural Language Processing (NLP) based methods have been widely used for the automatic detection of data contents from the text [12]. Artificial Intelligence (AI) based approaches have gained prominence for the development of sentiments or emotion-based systems [38]. In state-of-the-art Sentiment Analysis techniques, the issue is that they access the response in the context of positive or negative aspects but not the specific feelings of the customer and the intensity of their response. To deal with these issues, we present a sentiment analysis based method for identifying accessibility-related problems in mobile apps. The proposed approach is based on machine learning and NLP methods for Emotion detection and automatically tags user reviews as positive, negative, or neutral. A supervised learning method is employed for the annotation of the corpus. After performing annotation of the corpus, the labeled corpus is fed into the model for detecting emotions/sentiments from the user reviews. The proposed system comprises different stages, including data pre-processing, extraction of features, and the prediction model.

Following are the key contributions of our research:

- Use of sentiment analysis for tagging the accessibility-related user reviews.

978-1-6654-3583-3/21/$31.00 ©2021 IEEE
DOI 10.1109/ASEW52652.2021.00053

- Improvement of prediction accuracy of user-reviews tagging.
- We perform a comparison of TextBlob and VADER sentiment analyzers.
- A replication package of the dataset for extension purposes [1].

Paper organization. Section II discusses the related works. The study methodology describes in Section III. Following by Section IV, which presents study results, and Section V discusses the results. In Section VI we discuss the threats to validity. Finally, we conclude the paper in Section VII.

II. RELATED WORK

Accessibility in user reviews. Despite the fact that user reviews can significantly help in improving the accessibility of even well-established apps [17], [58], approximately 98.76% of users do not give feedback on accessibility problems to app stores [16]. Shockingly, roughly 1% of mobile app users give reviews on accessibility to help in future app improvements. The research was done by Eler et al. [16] to identify accessibility feedback using 214,053 mobile app reviews. Our study uses the mentioned dataset. However, after conducting the manual inspection, we uncovered only 2,663 mobile app reviews from the research focused on accessibility, and those were the ones that we used in this work. Our study is one of the few that have used sentiment analysis to analyze the preliminary dataset developed by Eler et al.

Text documents classification. Different researchers have used various taxonomies to classify their reviews, which depends on their objectives [13], [15], [27], [39], [45], [46]. For example, some studies classify their reviews into categories such as complaints, bug reports, and future feature requests. However, many of them do not focus or even mention accessibility.

The utilization of predefined keywords to categorize documents has been used in many previous studies, which is a deviation from automatic classification approaches. In a study by Eler et al. [16], the researchers used 213 keywords to investigate user reviews, while Ratzinger et al. [49] used 13 keywords. In our study, contrary to those done previously, we are using sentiment analysis to understand the opinions of app users on the accessibility of the apps so that we can understand the users' emotions (positive, negative, or neutral) when they are reporting about accessibility.

A similar study to ours was done by AlOmar et al. [5], which used automated machine learning to explore accessibility user reviews. In this study, we use sentiment analysis to evaluate accessibility user reviews in a selected database. As far as we know, this is the first study to use such an approach.

III. STUDY DESIGN

The main goal of our study is to automatically identify user reviews related to accessibility from the application reviews dataset. Reviews are provided as an input to our proposed approach, and then it performs sentiment analysis on the reviews, i.e., whether the review is positive, neutral, or negative. For this purpose, we generate the classification features using bag of words, extracted using TF-IDF, similarly to previous studies processing user reviews [36], [22]. We build our classification model using corpus reviews and current classification techniques. We then utilized the classification model to predict the types of new app reviews. The overview of the whole process is depicted in Figure 1. The key steps of our proposed approach are as follows:

Step (1) - Data Collection: For training, the dataset including the app reviews and their categories are identified through manual inspection [16].

Step (2) - Data Preprocessing: To improve the reviews of the proposed learning algorithms, data cleansing and pre-processing techniques, i.e., tokenizing, lemmatizing, stop words removal, and capitalization removal, are utilized [3], [2].

Step (3) - Sentiment Analysis: To tag the user reviews, we used two sentiment analyzers TextBlob [35] and VADER [26].

Step (4) - Feature Engineering: To create a structured feature space, TF-IDF and BoW [60] techniques are used on preprocessed review text.

Step (5) - Model Selection: We used six classification models for performance evaluation of the proposed prediction model, *i.e.*, LR, SVC, ETC, GNB, GBM, and ADA. The algorithms that are most commonly used for text classification were selected [41], [28]. The performance of the model is validated after training and evaluating the model. We have followed the approach provided by Kowsari et al. [30] which discusses state-of-the-art techniques and algorithms similar to [5] since the app reviews are in plain text.

Step (6) - Model Evaluation: We evaluated the performance of our selected models based on four parameters: accuracy, precision, recall, and F-score [19], [59].

A. Step 1: Data Collection

In our approach we use a dataset that contains 2,663 manually verified reviews related to accessibility by Eler et al. [16], as shown in Table I. The collected reviews have been extracted from 701 applications which fall under 15 different categories. Eler et al. [16] first collected 214,053 app reviews, then used 213 keywords for string matching and filtering down the reviews and kept only those reviews that contain accessibility-related information. 54 the British Broadcasting Corporation (BBC) recommendations [8] for accessibility are used for derivation of keywords. After this step, 5,076 potential accessibility reviews were selected after performing string matching.

TABLE I: Statistics of the dataset.

Number of Apps	701
App Categories	15
All Reviews	214,053
Accessibility Reviews	2,663

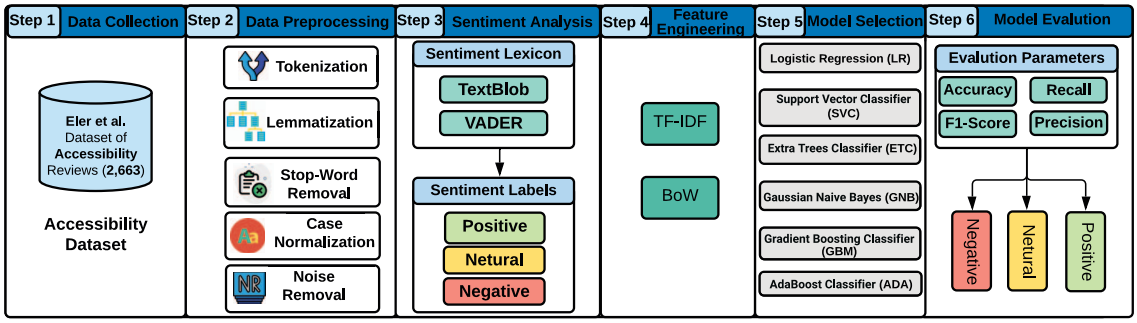

Fig. 1: Overview Approach of Our Study.

Manual inspection showed that 2,663 are true positive. The process of Levin et al. [33] is followed for verification of previous manually labeled reviews, and 243 out of 2,663, *i.e.*, we randomly selected 9% of sample reviews. This value is approximately equal to the sample size by 95% confidence level and 6 confidence interval. Then, 243 non-accessibility reviews were randomly added, and we had 486 total reviews. After that, their labeling was done by another researcher, and the data were kept confidential before. To avoid fatigue, 7 days were given to the review process, and the researcher was given the opportunity to perform online searching of keywords that they were unaware of during the labeling process. After completing the data labeling process, we validated them against the originally labeled reviews. Cohen's Kappa coefficient [14] was utilized to evaluate the categorical classes in terms of inter-rater agreement level, and an agreement level of "0.82" was achieved. The perfect agreement values are 0.81˘1.00 , and our agreement values are considered to have an almost perfect agreement according to Fleiss et al. [20]. The highest number of documents used in the related studies [33], [32] was approximately 2000. In contrast to the existing studies, we have selected 5,663 model creation and validation reviews as our aim was to provide sufficient reviews to the model that could signify all potential accessibility topics.

B. Step 2: Data Preprocessing

After completing the data collection process, we selected a text pre-processing approach from [30], which is similar to [5]. In order to perform text classification accurately by a model, it is necessary to clean and pre-process the document properly. For pre-processing the app reviews, NLP techniques using the Python natural language toolkit [54] have been used in our approach. These NLP based techniques include:

- **Tokenization:** In this process, the natural text is split into tokens that do not contain white space. The app reviews are tokenized by splitting them into a constituent set of words.
- **Lemmatization:** In this process, the suffix of a word is removed or replaced in order to get its basic form. It also reduces the count of unique occurrences of similar words. In the proposed approach, this technique is employed to pre-process the words in their canonical form to reduce the count of unique occurrences of similar text tokens.
- **Stop-Word Removal:** Words that do not contribute to the classification process, e.g., am, the, etc., are removed.
- **Case Normalization:** As the exact words with different font cases need to be treated similarly, e.g., Accessibility" and "accessibility", it is required to convert the whole text in lower case. It is generally known as a type of data cleansing which helps in avoiding repetition of the same features that differ only in terms of case sensitivity. In the context of accessibility-related reviews, a user can identify itself as "Deaf" with upper case 'D' for expressing his cultural identity in the reviews. Our classifier is binary; therefore, it will produce the same classification result for "Deaf" and "deaf" and the case normalization will be safe, and no overruling of users' expressions will be done.
- **Noise Removal:** Any noise that can deteriorate the classification performance and create confusion for the model while learning is removed in this step. Noise types that are removed in this step include numeric data, email id, special characters.

C. Step 3: Sentiment Analysis

We selected TextBlob and VADER tools in our study for the analysis. We used TextBlob because it is a higher accurate tool for sentiment analysis than other tools [50]. In comparison with the TextBlob, we used the VADER, a most use-able tool for sentiment analysis in recent studies [11]. We used both sentiment analysis tools for fair comparison analysis with multiple techniques.

1) TextBlob: is a widely used lexicon-based method[1] that performs different tasks related to natural language processing (NLP) on raw test [35]. TextBlob algorithm is implemented with a Python library named TextBlob that works as a programming interface for processing text data. With the help of TextBlob, different tasks can be performed, e.g., analysis of sentiments in text, creation of POS tags, extraction of noun phrases, etc. [54]. There are several built-in functions in TextBlob that help in performing different language processing tasks. TextBlob can work in different languages, e.g., Spanish, English, etc. According to research, [42], TextBlob helps in sentiment analysis of tweet data with positive, negative, or neutral polarity. TextBlob library works on top of Natural

[1]https://github.com/sloria/TextBlob

Language Toolkit NLTK, and its algorithm for sentiment analysis works together with NLTK and pattern processing [31]. Its dictionary includes around 2918 lexicons. In TextBlob, polarity calculation is done on two bases, i.e., objectivity (facts) or subjectivity (personal opinions). The sentiment analyzer returns a sentiment score that comprises polarity and subjectivity score. The sentiment scoring range of TextBlob is shown below:

TABLE II: TextBlob sentiment score range

Negative	Polarity score < 0
Neutral	Polarity score = 0
Positive	Polarity score > 0

For subjectivity, the facts-based sentiments have scores below 0.0, while the personal opinion-based sentiments have scores above 1.0.

2) VADER: (Valence Aware Dictionary for Sentiment Reasoning) is a lexicon-based approach[2] that works on gold standard heuristics. It has English language-based sentiment lexicons and is scored and validated by a human. To improve the performance of sentiment analyzer, these lexicons utilize qualitative methods. KirliA et al. [29] suggested that scoring done by VADER sentiment analyzer and human raters hold equal results. Multiple datasets are combined in VADER's corpus. Compared to the previous corpus that focuses on sentiment polarity, VADER also includes the intensity of the polarity score. Slang words and abbreviations that collectively make more than 7500 lexicons are present in its corpus. Scores range between -4.0 to +4.0. The score below -4 specifies the sentiment as negative, whereas the score above +4 indicates the sentiment as negative. Vader's output is depicted in different terms, i.e., neg, neu, pos, compound. Compound output is the aggregation of lexicon scores of complete sentences or a text and ranges from -1.0 to +1.0. The sentiment scoring range of VADER is shown below:

TABLE III: VADER sentiment score range

Negative	compound score <= -0.05
Neutral	compound score $>$ -0.05 to compound score $<$ 0.05
Positive	compound score >= 0.05

To represent the sentiment intensity and polarity, the VADER algorithm includes a sentiment lexicon approach and grammatical rules and syntactic conventions. The VADER lexicon approach contains different lexical features that include acronyms and emoticons. Hence around 7,500 sentiment features are present in its dictionary. To determine the sentiment intensity of a word, grammatical rules are considered, which can cause variation in the sentiment score of a word.

D. Step 4: Feature Engineering

Feature Engineering is a method of discovering significant characteristics from data to efficiently train machine learning algorithms or develop features from the main features [10], [4]. These features are being used to enhance the performance of machine learning algorithms [23]. This study used two methods of feature engineering as follows:

[2]https://github.com/cjhutto/vaderSentiment

1) TF-IDF: For information retrieval and summarization is one of the most used scoring metrics is TF-IDF. It is used for the representation of the term's significance in each text. TF and IDF are given as input in the extraction function of TF-IDF. Tokens that are infrequent in the dataset are given by TF-IDF. The significance of unusual words increases if it appears in two documents.

$$tfidf_{t,d,D} = tf_{t,d} \cdot idf_{t,D} \quad (1)$$

where terms are indicated by ***t***; each document by ***d***; set of documents by ***D***. Along with TF-IDF, the "n-gram range" parameter is employed. Words' weight that provides the weights of corpus for any word is calculated with the help of TF-IDF. The output is the word matrix being weighted. An increase in meaning is proportional to the count with the TF-IDF vectorizer, but the word frequency within the corpus helps to manage it. The TF technique is often considered for features extraction and is commonly used for the purpose of text classification. The terms' incidence frequency is employed as a parameter for classifier training. Unlike TF-IDF, in which less weight is given to more common terms, the TF function does not consider if a word is popular or not.

2) BoW: One of the methods used for the simple representation of Natural Language Processing (NLP) and information retrieval is Bag-of-Words, commonly known as BoW. It is the easiest and flexible way for obtaining the features of a document. In BoW, the histogram of the words in the text is looked at. To train the set, the words' frequency is used as a function. In this research, the CountVectorizer function using the pythons' Scikit-learn library is utilized to implement the BoW method. Vectorization is the process of converting a set of textual data into numerical vectors. Words' frequency helps in the operation of CountVectorizer, and it shows that counting of tokens is done and generation of the limited token matrix is completed [18]. The BoW is a list of features and terms that allocates a significance to each attribute which reflects the particular features' frequency [25].

E. Step 5: Model Selection

In our study, we considered the following learning algorithms to build our model:

- **Logistic Regression (LR):** It is a statistical model that is based on the concept of probability and is akin to linear regression. LR performs prediction of the outcomes' probability by fitting the data to a logistic function [7].
- **Support Vector Classifier (SVC):** It is a popular ML classifier for solving linear and non-linear problems. It works well for several practical applications [57], [51]. A line or hyperplane is generated by SVC, which separated the data into a section. Low-dimensional input space is transformed with the help of its Kernel function into higher dimensional space. This transformation means that non-separable issues are converted into separable ones. It mainly helps in solving non-linear differential problems. SVC separates the data based on labels and performs complex data transformations [9].

- **Extra Tree Classifier (ETC):** A combination of classification algorithms teaching approach in which outcomes of several de-correlated random forests collected in a "forest" are combined for generation of identifications' outcome, is Extra Tree Classifier (ETC). In principle, it is very similar to a Random Forest Classifier but differs from it in other ways, like a decision tree algorithm is built throughout the forest. In ETC, the Previous training dataset in the ET Forest is used for the creation of decision models [52].
- **Gaussian Naïve Bayes (GNB):** It employs the Gaussian distributions for the handling of continuous attributes in the Naive Bayes classification and represents the features' likelihood based on the classes [40]. GNB assigns each data point to its nearest class. It considers the distance from the mean point as well as performs its comparison to the class variance [24]. Moreover, GNB exhibits faster performance as compared to other algorithms [48].
- **Gradient Boosting Classifier (GBC):** Decision trees are widely utilized for performing gradient boosting. As they have shown significant results in the classification of the large system, GB frameworks have gained importance in machine learning [43].
- **Ada Boost Classifier (ADA):** uses a linear combination of "weak" classifiers for constructing a "strong" classifier, like GBC. The "weak" classifier can be considered a simple threshold operation on a specific feature category. Weak classifiers' training process is known as "Weak-Learn". Ada Boost consumes less memory and has fewer computational requirements [61].

F. Step 6: Model Evaluation

The performance of our selected models is measured using the four measurement aspects listed in Table IV below where:

- Positive Predictions labeled correctly by the classifier is determined by the True Positive (TP) parameter.
- Negative Predictions labeled correctly by the classifier is determined by the True Negative (TP) parameter.
- Number of negative instances mistakenly presumed as positive instances by the classifier are determined by the parameter False Positive (FP).
- Number of positive instances mistakenly presumed as negative instances by the classifier is determined by the parameter False Negative (FP).

TABLE IV: Summary of performance measures, formulas, and definitions.

Measures	Formula	Definition
Accuracy	$\frac{TP+TN}{TP+TN+FP+FN}$	Calculates the closeness of a measured value to the standard value.
Recall	$\frac{tp}{tp+fn}$	Calculates the exact number of positive predictions that are actually observed in the actual class.
Precision	$\frac{tp}{tp+fp}$	Calculates the exact no. of correct predictions out of all the input sample.
F1-score	$\frac{2 \cdot P \cdot R}{P+R}$	Calculates the accuracy from the precision and recall.

IV. STUDY RESULTS

This section discusses the results of our research work in light of the proposed research questions.

RQ$_1$: How do users express their sentiments in their accessibility app review?

As the expression of users' thoughts regarding the apps, reviews are used as a tool. If the accessibility features address the users' needs, the user reviews are written with positive sentiments On the other hand, if the accessibility features are not meeting user requirements, then attention is needed by the developers. These reviews reflect negative sentiments. Therefore, a review serves as a way to measure user satisfaction or dissatisfaction about the accessibility, and the negative reviews help identify accessibility topics that need to be fixed. In Figure 2, we present the comparison of sentiment analysis results between TextBlob and VADER. According to the results of TextBlob, 72.66% users have positive reviews, 15.88% have negative while the remaining 11.45% have the neutral review. On the other hand, results generated by the VADER approach show that 79.30% users have positive reviews, 10.75 are negative, and the remaining 9.95 are neutral reviews. Although there are some differences in the results of both sentient analyzers, they show a similar trend, *i.e.*, most of the users have positive reviews about the apps' accessibility, few users have negative reviews, while the least number of users have neutral views about.

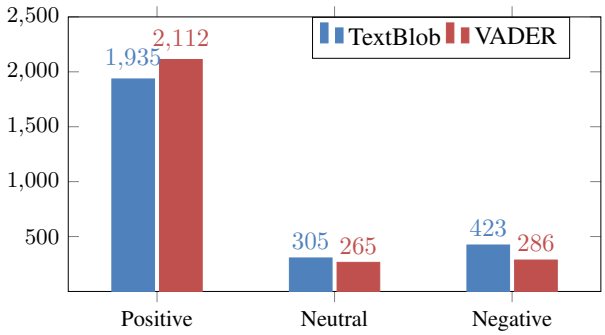

Fig. 2: Comparison of TextBlob and VADER sentiment analysis results.

RQ$_2$:How effective is our proposed sentiment analysis-based approach in the identification of accessibility reviews?

To analyze the sentiments of accessibility app users, we used two sentiments analyzers, i.e., TextBlob and VADER. Both sentiment analyzers help in the automatic prediction of emotions from user reviews. We also used six different machine learning models, i.e., SVC, GNB, GBM, LR, ADA, and ETC, along with TF-IDF and BoW features with the sentiment analyzers to categorize the sentiments based on the result of RQ1. Furthermore, we used four statistical measures for evaluating the proposed approach. We select the best hyperparameters setting using the hit and trial method. During tuning each time, we split the dataset and change the model's hyperparameters values. We have done this tuning between parameter values range such as for n_estimator in RF we start from 50, and we end up at 500 while our best value was 300.

The results of both sentiment analyzers, i.e., TextBlob and VADER, with TF-IDF and BoW in terms of accuracy,

TABLE V: Models performance comparison for TextBlob and VADER with TF-IDF and BoW features.

Sentiment Analyzer	Model	TF-IDF				BoW			
		Accuracy	Precision	Recall	F1-Score	Accuracy	Precision	Recall	F1-Score
TextBlob	**LR**	0.83	0.84	0.59	0.65	**0.86**	0.80	0.76	**0.77**
	SVC	0.81	0.80	0.54	0.60	**0.86**	0.76	**0.78**	**0.77**
	ETC	0.83	**0.88**	0.60	**0.66**	0.85	**0.86**	0.68	0.72
	GNB	0.65	0.55	0.48	0.50	0.65	0.55	0.47	0.50
	GBM	0.80	0.68	**0.66**	0.64	0.80	0.68	0.66	0.66
	ADA	0.71	0.57	0.64	0.59	0.73	0.61	0.69	0.64
Vader	**LR**	0.80	0.73	0.45	0.48	0.81	0.66	0.59	0.61
	SVC	0.80	0.72	0.43	0.46	0.82	0.65	0.65	0.65
	ETC	0.80	0.80	0.43	0.46	0.81	0.73	0.52	0.56
	GNB	0.68	0.45	0.42	0.43	0.68	0.44	0.42	0.43
	GBM	**0.84**	0.66	0.64	0.65	0.80	0.62	0.58	0.59
	ADA	0.74	0.46	0.42	0.44	0.72	0.49	0.49	0.49

precision, recall, and F-measure, are presented in Table V. We observe that when we used the TextBlob method with the TF-IDF technique, we found that LR and ETC exhibit the highest accuracy, i.e., 0.83. While for precision and F1-Score, ETC outperforms the remaining five classifiers. In terms of a recall measure, GBM performs better when compared to other techniques with TextBlob. For the BoW technique, TextBlob with LR and SVC achieved the highest accuracy (0.86) and F1-score (0.77). TextBlob with the ETC classifier attained 0.86 precision and recall of 0.78 by using the SVC classifier. Overall results show that TextBlob with the BoW method shows better accuracy, recall, and F1-score as compared to the TF-IDF method. On the other hand, the TF-IDF-based method outperforms BoW in terms of recall measure.

To measure the efficiency of TextBlob, which is a lexicon-based technique, we used the VADER technique on the same dataset. Based on the subjectivity and polarity, TextBlob performs assignment of the polarity score to each word ranging from -1 and 1. While VADER's performance depends on the mapping of lexicon features into sentiment scores done by a dictionary [26]. For the given dataset, the best accuracy (0.84), recall (0.64), and F1-score (0.65) for VADER with TF-IDF is achieved by GMB while ETC outperform in terms of precision (0.80). For VADER with BoW features, SVC outperform other models in accuracy (0.82), recall (0.65), and F1-score (0.65). VADER with ETC exhibits the similar trend in precision but with a lower value as compared to TF-IDF features. Overall, results show that TextBlob performs better than VADER with BoW as well as TF-IDF features.

V. Discussion

In this study, we applied sentiment analysis to identify the emotions of reviewed users towards the accessibility of apps. To facilitate the sentiment analysis, we used TextBlob and VADER, which are both popular lexicon-based methods. We wanted to know whether the two techniques could detect users' feelings towards accessibility in their apps based on machine learning techniques. The results of this study showed that sentiment analysis was crucial in identifying users' reviews towards the accessibility of apps, especially those that are disabled.

For many persons with disabilities (such as those who are deaf or blind), expressing their reviews towards various apps can be challenging. However, they can express their emotions (positive, neutral, or negative) towards an app, which may help developers understand whether it is accessible or not. We felt that disabled persons were not given much attention when collecting app reviews, probably because of the complexity involved in getting and analyzing their feedback. This study has shown that sentiment analysis could be the solution for determining the emotions of people with disabilities towards the accessibility of mobile devices. The findings of this study are important for software developers because it enables them to know whether disabled persons are satisfied or dissatisfied with the accessibility of their apps. Thus, developers can make any necessary changes to facilitate the use of the apps by persons with disabilities.

VI. Threats to Validity

Limitation of selected dataset. We performed training and testing of our approach on the previously collected dataset. However, this dataset contains a collection of accessibility reviews for Android open-source applications only. Hence, it cannot be generalized for all of the mobile applications available on the app store. Additionally, we have studied the English language-based reviews of open-source mobile applications only, so the results cannot be generalized for commercially developed projects or reviews are written in other languages.

VII. Conclusion

In this study, we presented an automated sentiment analysis-based approach for the classification of accessibility-related app reviews to help the developers detect these issues and improve their app's performance in light of user's reviews. We employed an existing dataset that is composed of manually validated accessibility reviews. Two sentiment analyzers, namely TextBlob and VADER using TF-IDF and BoW, are utilized with TF-IDF and BoW features. Both of the analyzers are coupled with six classifiers, namely LR, SVC, ETC, GNB, GBM, and ADA. Evaluation is done using four measures, i.e., Accuracy, Recall, Precision, and F1-Score, and the results show that TextBlob outperforms VADER in the classification of app reviews. Overall, results show that the ETC classifier performed best in TF-IDF features while svc is most efficient in BoW features. Sentiment analysis results also show that most of the users have given positive reviews about the accessibility of an app.

REFERENCES

[1] Accessibility dataset. https://doi.org/10.5281/zenodo.5540624, 2021.

[2] W. Aljedaani, Y. Javed, and M. Alenezi. Lda categorization of security bug reports in chromium projects. In *Proceedings of the 2020 European Symposium on Software Engineering*, pages 154–161, 2020.

[3] W. Aljedaani, M. Nagappan, B. Adams, and M. Godfrey. A comparison of bugs across the ios and android platforms of two open source cross platform browser apps. In *2019 IEEE/ACM 6th International Conference on Mobile Software Engineering and Systems (MOBILESoft)*, pages 76–86. IEEE, 2019.

[4] B. Alkhazi, A. DiStasi, W. Aljedaani, H. Alrubaye, X. Ye, and M. W. Mkaouer. Learning to rank developers for bug report assignment. *Applied Soft Computing*, 95:106667, 2020.

[5] E. A. AlOmar, W. Aljedaani, M. Tamjeed, M. W. Mkaouer, and Y. N. El-Glaly. Finding the needle in a haystack: On the automatic identification of accessibility user reviews. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, pages 1–15, 2021.

[6] A. Alshayban, I. Ahmed, and S. Malek. Accessibility issues in android apps: state of affairs, sentiments, and ways forward. In *2020 IEEE/ACM 42nd International Conference on Software Engineering (ICSE)*, pages 1323–1334. IEEE, 2020.

[7] G. Andrew and J. Gao. Scalable training of l 1-regularized log-linear models. In *Proceedings of the 24th international conference on Machine learning*, pages 33–40, 2007.

[8] BBC. The bbc standards and guidelines for mobile accessibility. https://www.bbc.co.uk/guidelines/futuremedia/accessibility/mobile, June 2021.

[9] K. P. Bennett and C. Campbell. Support vector machines: hype or hallelujah? *ACM SIGKDD explorations newsletter*, 2(2):1–13, 2000.

[10] F. F. Bocca and L. H. A. Rodrigues. The effect of tuning, feature engineering, and feature selection in data mining applied to rainfed sugarcane yield modelling. *Computers and electronics in agriculture*, 128:67–76, 2016.

[11] V. Bonta and N. K. N. Janardhan. A comprehensive study on lexicon based approaches for sentiment analysis. *Asian Journal of Computer Science and Technology*, 8(S2):1–6, 2019.

[12] G. G. Chowdhury. Natural language processing. *Annual review of information science and technology*, 37(1):51–89, 2003.

[13] A. Ciurumelea, A. Schaufelbühl, S. Panichella, and H. C. Gall. Analyzing reviews and code of mobile apps for better release planning. In *2017 IEEE 24th International Conference on Software Analysis, Evolution and Reengineering (SANER)*, pages 91–102. IEEE, 2017.

[14] J. Cohen. A coefficient of agreement for nominal scales. *Educational and psychological measurement*, 20(1):37–46, 1960.

[15] A. Di Sorbo, S. Panichella, C. V. Alexandru, C. A. Visaggio, and G. Canfora. Surf: summarizer of user reviews feedback. In *2017 IEEE/ACM 39th International Conference on Software Engineering Companion (ICSE-C)*, pages 55–58. IEEE, 2017.

[16] M. M. Eler, L. Orlandin, and A. D. A. Oliveira. Do android app users care about accessibility? an analysis of user reviews on the google play store. In *Proceedings of the 18th Brazilian Symposium on Human Factors in Computing Systems*, pages 1–11, 2019.

[17] M. M. Eler, J. M. Rojas, Y. Ge, and G. Fraser. Automated accessibility testing of mobile apps. In *2018 IEEE 11th International Conference on Software Testing, Verification and Validation (ICST)*, pages 116–126. IEEE, 2018.

[18] S. C. Eshan and M. S. Hasan. An application of machine learning to detect abusive bengali text. In *2017 20th International Conference of Computer and Information Technology (ICCIT)*, pages 1–6. IEEE, 2017.

[19] F. Fang, J. Wu, Y. Li, X. Ye, W. Aljedaani, and M. W. Mkaouer. On the classification of bug reports to improve bug localization. *Soft Computing*, 25(11):7307–7323, 2021.

[20] J. L. Fleiss, B. Levin, M. C. Paik, et al. The measurement of interrater agreement. *Statistical methods for rates and proportions*, 2(212-236):22–23, 1981.

[21] B. Gaind, V. Syal, and S. Padgalwar. Emotion detection and analysis on social media. *arXiv preprint arXiv:1901.08458*, 2019.

[22] T. Hasan and A. Matin. Extract sentiment from customer reviews: A better approach of tf-idf and bow-based text classification using n-gram technique. In *Proceedings of International Joint Conference on Advances in Computational Intelligence*, pages 231–244. Springer, 2021.

[23] J. Heaton. An empirical analysis of feature engineering for predictive modeling. In *SoutheastCon 2016*, pages 1–6. IEEE, 2016.

[24] R. Herbrich, T. Graepel, and C. Campbell. Bayes point machines. *Journal of Machine Learning Research*, 1(Aug):245–279, 2001.

[25] X. Hu, J. S. Downie, and A. F. Ehmann. Lyric text mining in music mood classification. *American music*, 183(5,049):2–209, 2009.

[26] C. Hutto and E. Gilbert. Vader: A parsimonious rule-based model for sentiment analysis of social media text. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 8, 2014.

[27] C. Iacob and R. Harrison. Retrieving and analyzing mobile apps feature requests from online reviews. In *2013 10th working conference on mining software repositories (MSR)*, pages 41–44. IEEE, 2013.

[28] N. Jha and A. Mahmoud. Mining non-functional requirements from app store reviews. *Empirical Software Engineering*, 24(6):3659–3695, 2019.

[29] A. Kirlic and Z. Orhan. Measuring human and vader performance on sentiment analysis. *Measuring human and Vader performance on sentiment analysis*, 1:42–46, 2017.

[30] K. Kowsari, K. Jafari Meimandi, M. Heidarysafa, S. Mendu, L. Barnes, and D. Brown. Text classification algorithms: A survey. *Information*, 10(4):150, 2019.

[31] R. A. Laksono, K. R. Sungkono, R. Sarno, and C. S. Wahyuni. Sentiment analysis of restaurant customer reviews on tripadvisor using naïve bayes. In *2019 12th International Conference on Information & Communication Technology and System (ICTS)*, pages 49–54. IEEE, 2019.

[32] S. Levin and A. Yehudai. Boosting automatic commit classification into maintenance activities by utilizing source code changes. In *Proceedings of the 13th International Conference on Predictive Models and Data Analytics in Software Engineering*, pages 97–106, 2017.

[33] S. Levin and A. Yehudai. Towards software analytics: Modeling maintenance activities. *arXiv preprint arXiv:1903.04909*, 2019.

[34] X. Li, Z. Zhang, and K. Stefanidis. Mobile app evolution analysis based on user reviews. In *New Trends in Intelligent Software Methodologies, Tools and Techniques*, pages 773–786. IOS Press, 2018.

[35] S. Loria. textblob documentation. *Release 0.15*, 2:269, 2018.

[36] M. Lu and P. Liang. Automatic classification of non-functional requirements from augmented app user reviews. In *Proceedings of the 21st International Conference on Evaluation and Assessment in Software Engineering*, pages 344–353, 2017.

[37] W. Maalej, H.-J. Happel, and A. Rashid. When users become collaborators: towards continuous and context-aware user input. In *Proceedings of the 24th ACM SIGPLAN conference companion on Object oriented programming systems languages and applications*, pages 981–990, 2009.

[38] J. Martınez-Miranda and A. Aldea. Emotions in human and artificial intelligence. *Computers in Human Behavior*, 21(2):323–341, 2005.

[39] S. McIlroy, N. Ali, H. Khalid, and A. E. Hassan. Analyzing and automatically labelling the types of user issues that are raised in mobile app reviews. *Empirical Software Engineering*, 21(3):1067–1106, 2016.

[40] T. M. Mitchell et al. Machine learning. 1997.

[41] D. Mukherjee and G. Ruhe. Analysis of compatibility in open source android mobile apps. In *2020 IEEE Seventh International Workshop on Artificial Intelligence for Requirements Engineering (AIRE)*, pages 70–78. IEEE, 2020.

[42] P. Munjal, M. Narula, S. Kumar, and H. Banati. Twitter sentiments based suggestive framework to predict trends. *Journal of Statistics and Management Systems*, 21(4):685–693, 2018.

[43] A. Natekin and A. Knoll. Gradient boosting machines, a tutorial. *Frontiers in neurorobotics*, 7:21, 2013.

[44] F. Palomba, M. Linares-Vásquez, G. Bavota, R. Oliveto, M. Di Penta, D. Poshyvanyk, and A. De Lucia. User reviews matter! tracking crowdsourced reviews to support evolution of successful apps. In *2015 IEEE international conference on software maintenance and evolution (ICSME)*, pages 291–300. IEEE, 2015.

[45] S. Panichella, A. Di 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 *2015 IEEE international conference on software maintenance and evolution (ICSME)*, pages 281–290. IEEE, 2015.

[46] L. Pelloni, G. Grano, A. Ciurumelea, S. Panichella, F. Palomba, and H. C. Gall. Becloma: Augmenting stack traces with user review information. In *2018 IEEE 25th International Conference on Software Analysis, Evolution and Reengineering (SANER)*, pages 522–526. IEEE, 2018.

[47] C. Power, A. Freire, H. Petrie, and D. Swallow. Guidelines are only half of the story: accessibility problems encountered by blind users on

the web. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pages 433–442, 2012.
[48] R. D. Raizada and Y.-S. Lee. Smoothness without smoothing: why gaussian naive bayes is not naive for multi-subject searchlight studies. *PloS one*, 8(7):e69566, 2013.
[49] J. Ratzinger, T. Sigmund, and H. C. Gall. On the relation of refactorings and software defect prediction. In *Proceedings of the 2008 international working conference on Mining software repositories*, pages 35–38, 2008.
[50] F. Rustam, M. Khalid, W. Aslam, V. Rupapara, A. Mehmood, and G. S. Choi. A performance comparison of supervised machine learning models for covid-19 tweets sentiment analysis. *Plos one*, 16(2):e0245909, 2021.
[51] N. Safdari, H. Alrubaye, W. Aljedaani, B. B. Baez, A. DiStasi, and M. W. Mkaouer. Learning to rank faulty source files for dependent bug reports. In *Big Data: Learning, Analytics, and Applications*, volume 10989, page 109890B. International Society for Optics and Photonics, 2019.
[52] A. L. Samuel. Some studies in machine learning using the game of checkers. *IBM Journal of research and development*, 3(3):210–229, 1959.
[53] N. Seyff, F. Graf, and N. Maiden. Using mobile re tools to give end-users their own voice. In *2010 18th IEEE International Requirements Engineering Conference*, pages 37–46. IEEE, 2010.
[54] S. Vijayarani, R. Janani, et al. Text mining: open source tokenization tools-an analysis. *Advanced Computational Intelligence: An International Journal (ACII)*, 3(1):37–47, 2016.
[55] P. M. Vu, T. T. Nguyen, H. V. Pham, and T. T. Nguyen. Mining user opinions in mobile app reviews: A keyword-based approach. *arXiv preprint arXiv:1505.04657*, 2015.
[56] W3C. Web content accessibility guidelines (wcag) 2.1. https://www.w3.org/TR/WCAG21/, June 2021.
[57] X. Wu, V. Kumar, J. R. Quinlan, J. Ghosh, Q. Yang, H. Motoda, G. J. McLachlan, A. Ng, B. Liu, S. Y. Philip, et al. Top 10 algorithms in data mining. *Knowledge and information systems*, 14(1):1–37, 2008.
[58] S. Yan and P. Ramachandran. The current status of accessibility in mobile apps. *ACM Transactions on Accessible Computing (TACCESS)*, 12(1):1–31, 2019.
[59] X. Ye, Y. Zheng, W. Aljedaani, and M. W. Mkaouer. Recommending pull request reviewers based on code changes. *Soft Computing*, 25(7):5619–5632, 2021.
[60] B. Yu. An evaluation of text classification methods for literary study. *Literary and Linguistic Computing*, 23(3):327–343, 2008.
[61] Y. Zhao, L. Gong, B. Zhou, Y. Huang, and C. Liu. Detecting tomatoes in greenhouse scenes by combining adaboost classifier and colour analysis. *Biosystems Engineering*, 148:127–137, 2016.