第 48卷第 6期
七月 2023
Diqiu Kexue - Zhongguo Dizhi Daxue Xuebao/Earth Science
Journal of China University of Geosciences
https://dqkx-periodicals.com
Vol. 48 No. 6
December 2023

https://doi.org/10.3799/dqkx.2023.8163

User Reviews Classification in Play Store Applications Using Deep Learning: An Empirical Study

Mehrdad Razavi Dehkordi[a], Hamid Rastegari [a,b], Akbar Nabiollahi-Najafabadi [a,c], Taghi Javdani Gandomani[d]

Corresponding author : Dr Hamid Rastegari at Faculty of Computer Engineering, Najafabad branch, Islamic Azad University, Najafabad, Iran.
Email : rastegari@iaun.ac.ir

Abstract: Since developing mobile applications, users usually express their requirements as reviews under applications residing in the Google Play Store or Apple AppStore. Some methods were proposed for automatically classifying user reviews, most of which employ old methods and databases or provide poor accuracy. In this article, a model named DARCLSTM is proposed to improve the process of user reviews classification. In the proposed model, the New Kaggle dataset containing 51,000 reviews related to the year 2021 is given to the deep learning system with LSTM architecture to train the model after pre-processing, removing noisy data and data cleaning along with the reviews lables. Then the trained model is used to classify reviews into three groups bug reports, feature requests, and information_giving while using an app. Then, the model is compared to other methods (other proposed models using machine learning or deep learning), indicating the outperformance of proposed model against previous studies. The F-measure (11%) and accuracy (97%) parameters have significantly improved in the proposed model.

Keywords: *Mobile application, user reviews, google play store, classification, deep learning, machine learning.*

[a] Faculty of Computer Engineering, Najafabad branch, Islamic Azad University, Najafabad, Iran, mehrdad.razavi68@gmail.com
[b] Big Data Research Center, Najafabad branch, Islamic Azad University, Najafabad, Iran, rastegari@iaun.ac.ir
[c] Big Data Research Center, Najafabad branch, Islamic Azad University, Najafabad, Iran, a.nabi@pco.iaun.ac.ir
[d] Computer Science Department, ShahreKord University, ShahreKord, Iran, javdani@sku.ac.ir

1. Introduction

According to the IDC official website[5], a large portion of today's smartphone market belongs to the Android OS. *Google play Store*, as the main android store, hosts apps from developers [6]. Statistics indicate the Google Play Store and iOS Apple Store include about 4 million apps in total[7].

The Google Play has enabled users to download the existing apps from it and place their reviews[1, 2]. Studies revealed that the reviews include important information. This information includes bug reports, feature requests, and user experience while using an app [3, 4, 5]. Previous studies indicate user reviews may lead the software development process and result in improved future versions of software [6, 7]. Moreover, recent studies revealed important information in reviews crucial for program designers and analyzers [4, 5, 8]. Considering the large volume of reviews and their vital embedded information, their manual classification is a time-consuming and tedious task for app developers [9]. An automated review classification system comes in handy for developers when debugging the app to improve it and saves time in review classification/analysis.

So far, many different methods are proposed for automatically classifying user reviews, most of which face challenges like employing old methods and datasets or providing poor accuracy. Some methods that used deep learning (DL) in training their model provided low accuracy and used an old dataset. The 2021 Kaggle dataset and LSTM architecture-based DL – with a desired performance in classification [10] – are used to train the proposed model in this article.

The proposed model initially inputs the dataset and performs the data-cleaning process, including removing noisy, numbered data and non-English sentences. Then, the DL system's parameters are set, and model training begins with the LSTM architecture. The proposed model is named DARCLSTM — Deep learning-based App Reviews Classifier with LSTM. It is compared to the other old models in terms of their evaluation parameters and resolved the aforementioned challenges; furthermore, it yielded acceptable results compared to the best-related studies.

The rest of this paper is organized as follows: Section 1 provides DL-related notions and includes the most important related works, and the proposed reviews classification model is provided in Section 2. The proposed method is compared against other similar studies in Section 3. Finally, Sections 4 include the discussion and conclusion, respectively.

[5] IDC - Smartphone Market Share - Market Share." https://www.idc.com/promo/smartphone-market-share

[6] Android Apps on Google Play." https://play.google.com/store/apps

[7] Biggest app stores in the world 2022 | Statista." https://www.statista.com/statistics/276623/number-of-apps-available-in-leading-app-stores/

1.1. Background

Artificial neural networks are vastly distributed processors comprised of small side-by-side processors responsible for performing calculations and holding knowledge [11]. Neural networks are suitable for training models due to the following benefits: a) their data-driven and self-adaptive nature reduces errors and enables them to adapt themselves to models; b) they can be generalized and enabled to infer unseen (new) data even at the presence of noise, c) they are nonlinear, and d) they are trainable with little samples [12, 13, 14]. However, sometimes neural networks are stuck into local minima [16], and lower layers are not trained because of the reduced related slopes while employing back-propagation error learning [15]; therefore, these networks have lost their popularity.

DL is based on a set of machine learning algorithms in which high-level data abstract models are modeled by multiple nonlinear transformations. This technology works on top of artificial neural networks, capable of improving their performance using learning algorithms and enhancing the volume of data. DL is comprised of multiple layers nonlinearly used for transformation and feature identification [17, 18], providing the system with the most accuracy and data understanding. Among DL applications are image recognition, speech recognition, voice recognition, text classification, medical diseases analysis, drug recognition, bioinformatics, and mobile advertisement [17, 18, 19].

In this regard, Hochreiter and Schimdhumber devised LSTM with many different applications [20]. IBM used this method for speech recognition [17]. This method uses a memory unit named a cell that preserves its value for sufficient time, then uses it as its input function. This capability helps the unit to keep its last calculated value. Investigations showed that LSTM outperforms Multi-Layer Perceptron (MLP) networks in terms of scalability and classification [10, 21]. LSTM accesses more contextual information compared to RNN. The first step of LSTM involves keeping/removing information through its forget gate. The input of forget gate includes previous latent states and current input information. The operation outputs a value between 0 and 1 to the memory state cell; 0 means complete removal, and 1 means complete keeping. Then, the input gate updates information and keeps a new candidate vector on the memory cell. In the next step, i.e., cell update status, the forget gate determined information is added to the candidate vector. Finally, the cell state and output gate determine the output. Fig. 1 shows a cell from this architecture

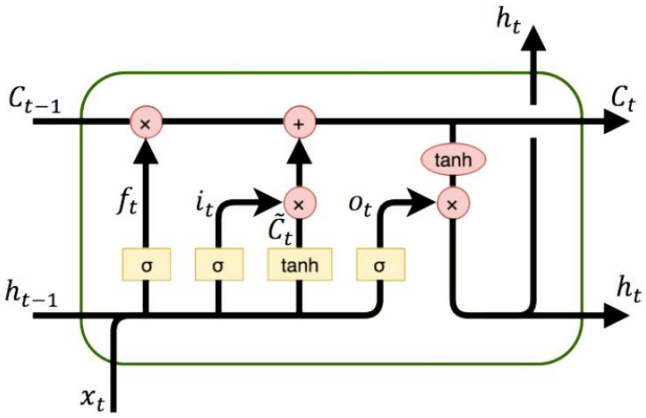

Figure 1.1. *LSTMCell [22]*

1.2. Related Work

Among numerous works done in the field of mobile applications' user reviews classification, the most important studies are as follows.

Chen et al. proposed a tool named AR-Miner for detecting Informative/nonInformative Reviews for the development team. The proposed model uses the reviews of Facebook and SwiftKey applications as well as Temple Run and Tapfish games for training using EMNB5, Top Modeling, algorithms; it could classify Informative /non- Informative reviews with a 0.7 hit rate [23].

Maalej and Nabil, in a study in 2016 [24], investigated the best method for user reviews classification in Play Store based on feature requests, bug reports, user experience, and rating. They developed a database of 1100 applications and 1.1 million reviews. They concluded the Bayes theory is the best classification method with a precision of 0.82. Moghaddam [25] investigated 50000 reviews from eBay AppStore in 2015 and managed to identify the existing patterns in user reviews using Part of Speech (PoS) Tagging. Then he used sentiment analysis, Support Vector Machine (SVM), and LDA feature reduction for classification to classify user reviews into classes of feature requests, bug reports, and user experience reports. The pattern-based method acquired a precision of 88% in reviews classification [25]. Johann et al. proposed a tool named SAFE in 2017 for extracting features in the Play Store user reviews. They manually extracted 18 PoS patterns and 5 sentence patterns. After investigating reviews from 10 applications, they acquired a precision of 0.55 and a recall of 0.43 [26].

Scalabrino et al. employed a machine learning method and a tool named CLAP for classifying reviews for clustering reviews and prioritizing clusters required for the development team to implement in the future. They named operational bug reports, proposing new feature and nonfunctional requirements. Finally, they acquired a precision of 0.87 using random tree and DBScan algorithms and employing tools database Ar-Miner [23].

Martens et al. studied fake reviews in the Apple AppStore in 2019; they consulted 43 fake review producers resulting in a database including 60000 fake reviews and 62 million reviews. After classifying the database using seven different classification tools, it turned out that Random Forest (RF) classified fake/real comments by recall 98% [28].

Aslam et al. in 2020 employed the 2016 database provided by Maalej et al. study [24] and extracted 4400 reviews from this database; they presented a review classification system using Convolutional Neural Networks (CNN) and Deep Learning. The proposed system had an F-measure of 94.7% in user reviews classification [29].

Qiao and Wang proposed a model for reviw classification in Apple AppStore applications based on DeepLearning with LSTM architecture. The model was built by investigating 18261515 comments from 4602 games. Then the reviews were labeled as

three classes feature request, bug report, and miscellaneous by 26 bachelor students. The F-measure of this DL-based tool was 0.769, and its recall rate was 0.768 [30].

2. LSTM Based User Reviews Classifier

In this article, a DL-based model is proposed to classify reviews. To train the model to classify reviews as feature requests, bug reports, and information giving, first, we should determine the DL parameters. These parameters include the number of classes, length of sentences, utilized architecture for the learning system (type of RNN cell), number of layers in each cell, a portion of training and test data, number of nodes in each network layer, optimizer function, number of input sentences at each step, and number of times to perform model training.

Because the text is unstructured and it is difficult to extract features from it, using Glove Word Embedding, the words in the Reviews are converted into a vector containing numbers with 300-dimensional space. Considering that RNN networks work well with ordinal data and text is also a type of ordinal data, using RNN is the best method to solve problems related to text. Each RNN cell contains input words in vector format and hidden layer contains it's past knowledge. But because the RNN cell has a short-term memory and only remember the surrounding words, they are not suitable for cases that require words far away from them.

In Figure 3, to train the model, each word of the sentence is given as a vector to an LSTM cell separately, after remembering the location of the words and the relationship of the words with each other, a general representation of the sentence is made in an encoded form. For each LSTM cell, an input weight for the word and an input weight from the previous cell are entered into the new cell and the weights are combined. Then the weights along with the bias are given to the forgetting gate and at this stage it is decided which part is kept and which part is forgotten. Next, if the context of the text has changed, the cell has to receive new information. This new information is added to the cell in the addition gate with Addition Operation. Next, all the vectors are given to the last layer and according to the weights along with the hidden layers, the category in which the review is placed is determined.

Table 1. *Parameters tuned for training model based on Deep Learning*

Parameter	***Value or Type***	**Parameter**	***Value or Type***
A. Max number of Words in each Sentence	300	EmbeddingSize	300
B. Number of Words to Use	1000	Number of Cell Layers	1
C. Train Size	70%	Loss Function	Categorical Cross Entropy
D. Test Size	30%	RNN Cell Architecture	LSTM
Units	128	WordEmbedding	Glove
		Epochs	10
is Bidirectional	False	Number of Classes	3
Optimizer	Adam	DropOut Rate	0.4
BatchSize	64		

As you can see in Table 1, 70% of the data is assigned for training and 30% for testing the model. The LSTM architecture is used for training the model using deep networks. The model is trained ten times so that the accuracy be maximized. Fig. 2 shows the LSTM-based DL architecture along with its input and output.

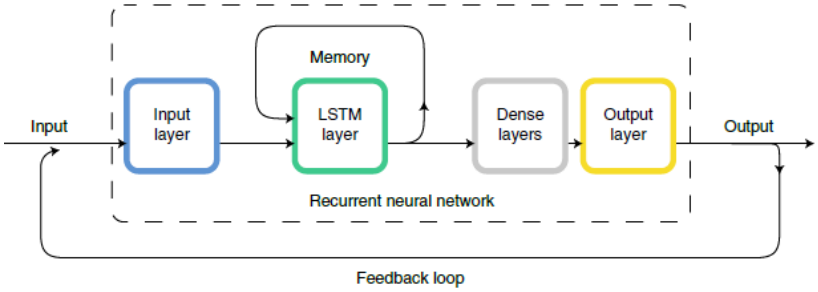

Figure 2.1 *- Schematic of the RNN architecture used, showing the input layer, the LSTM recurrent layer, and two hidden (dense) layers[32]*

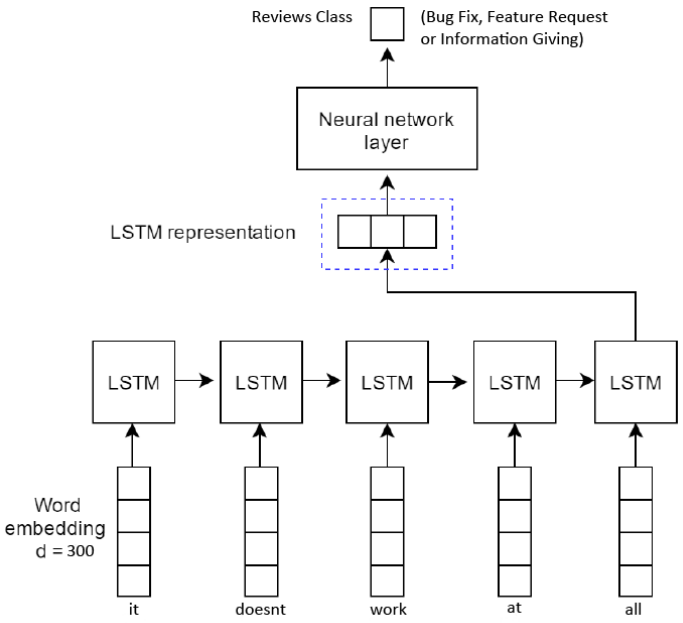

Figure 2.2*- Illustration of our LSTM model for User Reviews classification*[33]

According to Figure. 2.2, the LSTM-based network comprises input, output, and middle layers. Figure. 2.3 shows the architecture of the model with more details during classification. The input is the review expressed by the user. It is transformed into a vector space of 300 members by Word Embedding and is sent to LSTM, then to the neural network layer; finally, the category to which the reviews belongs is determined in the output. Fig. 4 shows the input/output details of the DL system used in the proposed model. The input includes 51000 reviews along with three labels, including the bugfix, feature request, and information_giving, which are used for training the DL-based model after the data cleaning step.

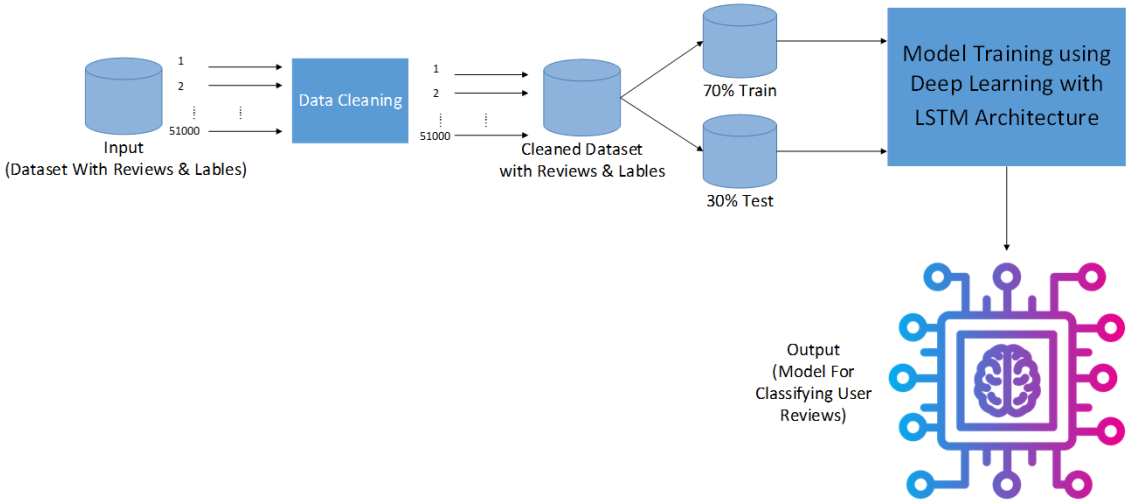

***Figure 2.3**- Details of input & output of Deep Learning System*

3. Findings

3.1. Utilized dataset, process, and data cleaning

The labeled Kaggle dataset is used for training the model. Tables 2 and 3 show the full information about the dataset and its features.

***Table 2.** Used Dataset with details*

Provider	Number of Reviews	Number of Apps	Number of Categories
Kaggle	51000	10842	32

***Table 3.** Features available for each application in the used dataset*

Number	Feature	Number	Feature
1	App Name	2	Category
3	Average app rating (0 to 5)	4	Number of Reviews
5	App Size in MB	6	Number of Installations
7	Free or Paid	8	Price of an app in case of not free
9	Age limit for using an app	10	Date of last app update
11	Last Version of app	12	Minimum Android version required for installing app
13	Reviews for app in text format	14	Reviews Lable (Feature Request,BugFix, and Information Giving)

Figure 3.1 summarizes application frequency in each category. According to Figure 3.1, the most number of applications respectively belongs to the categories Game with 1214

apps, Tools with 1682 apps, and Family with 1972 apps; the least number of applications respectively belongs to categories Art & Design with 65 items and Auto & Vehicles with 85 items. The non-equality of each category's application frequency is because of different popularity among different categories.

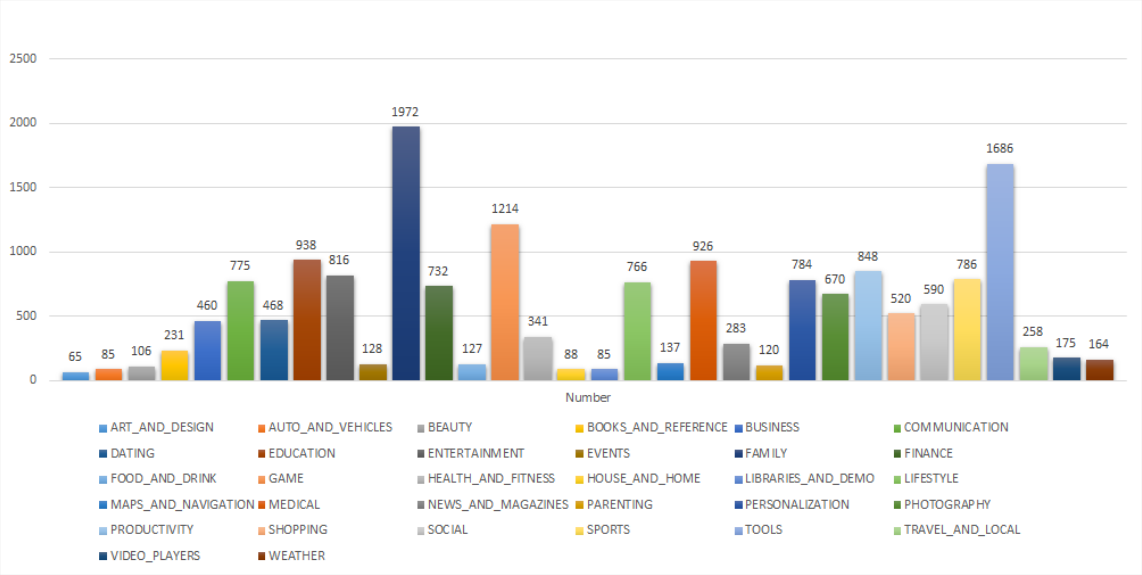

Figure 3.1- *Number of apps in each category of Google Play Store*

The employed Kaggle database is asymmetric; various experiments have proved that data symmetricity or asymmetrically (skewed or non-skewed data) does not affect the accuracy of the model in ML- or DL-based classifying systems [34].

3.2. Data preprocessing and cleaning

The working dataset includes noisy data such as non-English user reviews, useless numbers, stop words, and upper/lower case letters. Therefore, it should be cleaned and preprocessed so that cleaned data enters the DL-based learning system and the accuracy of the system is improved. Figure 3.2 shows the preprocessing in total.

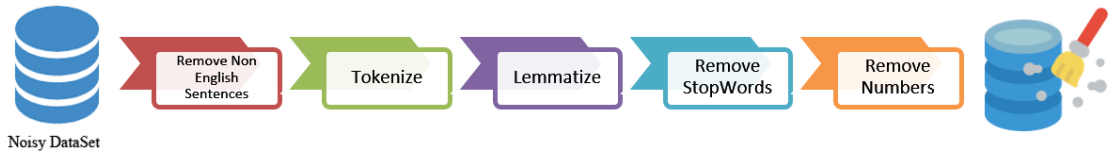

Figure 3.2- *Steps to remove the noisy from dataset*

Numerous articles indicate that DL-based networks readily get along with noisy data [35, 36]. To investigate this and its impact on system accuracy, we train the model both with noisy and cleaned data.

The data cleaning and noise removal steps in this study include removing non-English reviews and numbers, word lemmatizing, and lowercasing uppercase letters or words, which is performed using Python.

Then, the proposed model DARCLSTM (Deep Learning-based App Reviews Classifier With LSTM) is compared to other proposed tools, frameworks, and models considering their respective parameters.

3.3. Experiment and training environment

Training is done using Python v3.1.11 in Visual Studio Code v1.78.2. This environment is selected due to the simplicity of debugging and its similarity to Microsoft Visual Studio.

A personal computer with the following specifications is used for the processes: AMD Ryzen 2400G processor, 16 GB of RAM DDR4, no Graphics card, and Windows 10 OS.

3.4. Research Questions

We investigate the following research questions for the evaluation of DARCLSTM model:

RQ1: Can DARCLSTM outperform the state of the art in the classification of reviews? If yes, to what extent?

RQ2: Does data preprocessing/cleaning affect the accuracy of the model?

3.4. Experimental design

As can be seen in Figure 2.3, 70% of the reviews in the Kaggle dataset, along with their labels, are selected as training data and 30% as testing data. After performing the training process, the accuracy and loss parameters are measured. The training process is repeated because the test/train data are selected randomly, and again the accuracy and loss parameters are measured. This operation is performed ten times, in which each time accuracy is enhanced, and loss is decreased. In the end, the average value of accuracy and loss is calculated. Then, the trained model is uploaded to a location to be compared to other models/tools.

RQ1: To address this question, the proposed model is compared to the following models: Chen et al. (Ar-Miner tool) [23], Maalej et al. [25], Scalabrino et al. (CLAP tool) [27], and Aslam et al. (DCAR tool) [29]. For comparison purposes, the labeling of the Kaggle dataset used in DARCLSTM is modified according to the classification performed in each tool, and the training process is repeated. Then, the dataset used to test tools or frameworks provided by others is given to the new model. Finally, the results are compared in terms of the authors' intended parameters.

RQ2: To address the extent of data preprocessing (data cleaning) effect on the accuracy of the classification model, the database is entered into the DL-based learning system with and without data cleaning. The accuracy is measured in each step, and the results are finally compared.

3.5. Experiment results

RQ1: Comparing with other proposed tools/frameworks:

to address this question, as mentioned before, the DARCLSTM model is compared to other proposed tools/frameworks.

3.5.1 Comparing the proposed model to the Ar-Miner proposed by Chen et al. [23]

They categorized the reviews as Informative and non-Informative. Informative reviews include a functional flaw that produces incorrect or unexpected results, a functional flaw that degrades the performance of the application, add/modify feature requests, remove advertisements/notifications requests, and permission removal. Non-Informative reviews include pure user emotional reviews, Descriptions of (apps, features, actions, etc.), overall description regarding a bug or a fault, and questions and inquiries [23].

Then our used dataset is orchestrated with the intended labels of Chen et al. They employed reviews of Facebook and Swiftkey as well as TempleRun2 and Tapfish games. The apps and games had labeled and unlabeled reviews to be evaluated. The model is trained using our used dataset with Chen et al labels; Table 4 shows the comparison results for the apps and games.

Table 4. *Comparison of the proposed model with ar-miner*

AppName	Number of Reviews	Ar-Miner	DARCLSTM
Facebook	2000	0.877	0.961
Swiftkey	2000	0.764	0.958
TempleRun2	2000	0.797	0.93
TapFish	2000	0.761	0.912

As can be seen in Table 4, the Ar-Miner tool classifies Informative and non-Informative reviews with an average F-measure of 0.799, while this metric is 0.94 for the proposed DARCLSTM model; that is, the proposed model outperforms the Ar-Miner tool by 12.1%.

3.5.2 Comparing the proposed model to the Maalej et al model [25]

Table 5 shows the working categories of Maalej and the keywords categorizing each app. Their categories include bug reports, feature requests, user application experience, and user ratings.

Table 5*. Categories used by maalej et al with Keywords*

Review Type	KeyWord
Facebook	Bug, fix, problem, issue, defect,…
Swiftkey	Add, please, could, would,…
TempleRun2	Help, support, assist,…
TapFish	Great, good, nice,…

The dataset used by Maalej et al includes 370 reviews in the bug report category and 296 reviews in the feature request category. Next model training was done using our dataset and Maalej labels. Table 6 shows the comparison results

Table 6. *Comparison of the Maalej model with the proposed model*

Number of Reviews	Class	Maalej	DARCLSTM
370	Bug Report	0.78	0.963
296	Feature Request	0.76	0.945

According to Table 6, their method's average F-measure is 0.77 in classifying reviews into BugReport and Feature Request categories. However, the average F-measure is 0.954 for the proposed DARCLSTM tool. That is, the proposed model outperforms Maalej's method by 15.9%.

3.5.3 *Comparing the proposed model to the CLAP tool [27]*

CLAP's used dataset includes 3000 reviews from Chen's database – the Ar-Miner tool [23]. Next model training was done using our dataset and CLAP labels. Table 7 shows the comparison results.

Table 7. *Comparison of the Maalej model with the proposed model*

Number of Reviews	Average Classification F-Measure in CLAP	Average Classification F-Measure in DARCLSTM
200	0.86	0.97

According to Table 7, CLAP's average F-measure is 0.86 in classifying reviews into the mentioned categories. However, the average F-measure is 0.93 for DARCLSTM. That is, the proposed model outperforms CLAP by 7%.

3.5.4 *Comparing the proposed model to the Aslam et al. tool (DCAR) [29]*

Aslam et al. used Maalej's database [24]. The only difference is that they used deep learning with convolutional neural network(CNN) architecture for the learning process. Table 8 shows the comparison results

Table 8. *Comparison of the Maalej model with the proposed model*

Number of Reviews	Class	DCAR	DARCLSTM
370	Bug Report	0.9435	0.971
296	Feature Request	0.947	0.975

According to Table 8, the average F-measure for DARCLSTM is 0.963, which outperforms DCAR by 1.8%. Furthermore, it should be considered that DCAR is trained using an old 2016 dataset, while our proposed model is trained using a 2021 dataset.

In Table 9, all of the presented tools and methods by others are compared with the proposed DARCLSTM model in terms of the average F-Measure

Table 9. ***Categories used by maalej et al with Keywords***

Author/ Tool	Average Classification F-Measure
Chen et al/Ar-Miner	0.799
Maalej et al	0.77
Scalabrino/Clap	0.86
Aslam/DCAR	0.945
Proposed Methodology (DARCLSTM)	**0.953**

According to Table 9, the accuracy is increased since 2014, considering the advent of new methods for training user reviews classification models. Furthermore, the F-measure of the proposed method outperforms all state-of-the-art methods & models.

RQ2: The effect of data preprocessing & data cleaning on the accuracy of the classification model:

To address this question, both cleaned and noisy datasets are used to train the model. Once, the cleaned dataset is entered into the system, the learning operation is repeated ten times, and the average accuracy is calculated. Once again, the noisy dataset underwent the exact same operations. Then, their accuracy is compared, which is shown in Table 10.

Table 10. *Comparison of accuracy in user reviews classification using clean and noisy dataset*

Type of DataSet	Classification Accuracy
Cleaned Dataset	97.21%
Noisy Dataset	97.15%

As shown in Table 10, the model performs slightly better with cleaned data, yet data cleaning has no significant effect on accuracy.

4. Discussion and Conclusion

4.1. Discussion

Symmetric & Asymmetric Data: symmetric distributions occur when variable values appear in regular frequencies, and the mean, median, and mode all occur in one state. Graphically, symmetric distribution may appear as a normal distribution (bell diagram). This distribution is a key concept in technical analysis of trading because price corresponds to a symmetric distribution curve over time. Symmetric distribution usually

contradicts asymmetric distribution, which is directly related to skewness and other irregularities [34].

- The working dataset in this article is derived from Kaggle and, therefore, is asymmetric and, as mentioned, has no significant effect on accuracy.
- Considering the high volume of the unstructured text reviews expressed in the applications, extracting features from them is tedious and ML algorithms present poor accuracy in this regard.
- According to Table 9, the proposed model's F-measure is higher than models trained with CNN. The reason is that the training process is performed using a more updated dataset, and the LSTM method has a better performance compared to CNN.
- According to Table 9, the proposed model's F-measure is also higher than models trained with ML algorithms. The reason is that the DL-based methods perform better in vector spaces [38] with many features.
- In addition, because the new database was used to train the model, LSTM was more accurate than other methods in categorizing user reviews.

According to the above analysis, it could be concluded that the proposed user reviews classification model outperforms other methods.

4.2. Conclusion & Future Work

Considering the large volume of user reviews and their time-consuming and tedious manual classification, an automated reviews classification system is required. Such a system helps developers in improving and debugging their applications as well as saving time. A new Deep Learning-based method named DARCLSTM is proposed for classifying user reviews into three categories bug report, feature request, and information_giving. The proposed model utilizes an authorized, new 2021 dataset and provides more accuracy in classifying reviews compared to other proposed tools/methods. Furthermore, using the proposed model, developers receive faster analyses and readily access classified reviews.

For future work, one may use transformers to adjust and train BERT-based models more precisely with the provided dataset to achieve more accuracy. The proposed model could be trained by newer datasets or more data to increase its accuracy. Furthermore, in the presence of non-English reviews, one can train the model in other languages, providing reviews classification in non-English languages. The trained model could be trained for classification in other mobile platforms, such as Apple and BlackBerry, if any.

5. References

[1] M. R. Dehkordi, H. Seifzadeh, G. Beydoun, and M. H. Nadimi-Shahraki, “Success prediction of android applications in a novel repository using neural networks,” *Complex Intell. Syst.*, vol. 6, no. 3, pp. 573–590, 2020, doi: 10.1007/s40747-020-00154-3.

[2] W. Martin, F. Sarro, Y. Jia, Y. Zhang, and M. Harman, “A survey of app store analysis for software engineering,” *IEEE Trans. Softw. Eng.*, vol. 43, no. 9, pp. 817–847, 2017, doi: 10.1109/TSE.2016.2630689.

[3] E. Guzman and W. Maalej, “How Do Users Like This Feature? A Fine Grained Sentiment Analysis of App Reviews,” in *2014 IEEE 22nd International Requirements Engineering Conference (RE)*, Aug. 2014, pp. 153–162, doi: 10.1109/RE.2014.6912257.

[4] D. Pagano and W. Maalej, “User feedback in the appstore: An empirical study,” in *2013 21st IEEE International Requirements Engineering Conference (RE)*, Jul. 2013, pp. 125–134, doi: 10.1109/RE.2013.6636712.

[5] L. V. G. Carreno and K. Winbladh, “Analysis of user comments: An approach for software requirements evolution,” *Proc. - Int. Conf. Softw. Eng.*, pp. 582–591, 2013, doi: 10.1109/ICSE.2013.6606604.

[6] W. Maalej and D. Pagano, “On the socialness of software,” *Proc. - IEEE 9th Int. Conf. Dependable, Auton. Secur. Comput. DASC 2011*, pp. 864–871, 2011, doi: 10.1109/DASC.2011.146.

[7] N. Seyff, F. Graf, and N. Maiden, “Using mobile RE tools to give end-users their own voice,” *Proc. 2010 18th IEEE Int. Requir. Eng. Conf. RE2010*, pp. 37–46, 2010, doi: 10.1109/RE.2010.15.

[8] A. Al-Subaihin *et al.*, “App store mining and analysis,” in *Proceedings of the 3rd International Workshop on Software Development Lifecycle for Mobile*, Aug. 2015, pp. 1–2, doi: 10.1145/2804345.2804346.

[9] C. Stanik, “Requirements Intelligence: On the Analysis of User Feedback,” Hamburg, 2020.

[10] X. Yu, X. Wu, C. Luo, and P. Ren, “Deep learning in remote sensing scene classification: a data augmentation enhanced convolutional neural network framework,” *GIScience Remote Sens.*, vol. 54, no. 5, pp. 741–758, 2017, doi: 10.1080/15481603.2017.1323377.

[11] C.-H. Ko and M.-Y. Cheng, “Dynamic Prediction of Project Success Using Artificial Intelligence,” *J. Constr. Eng. Manag.*, vol. 133, no. 4, pp. 316–324, Apr. 2007, doi: 10.1061/(ASCE)0733-9364(2007)133:4(316).

[12] G. Zhang, B. Eddy Patuwo, and M. Y. Hu, “Forecasting with artificial neural networks:,” *Int. J. Forecast.*, vol. 14, no. 1, pp. 35–62, Mar. 1998, doi: 10.1016/S0169-2070(97)00044-7.

[13] H. S. Hippert, C. E. Pedreira, and R. C. Souza, “Neural networks for short-term load forecasting: a review and evaluation,” *IEEE Trans. Power Syst.*, vol. 16, no. 1, pp. 44–55, 2001, doi: 10.1109/59.910780.

[14] M. D. Odom and R. Sharda, “A neural network model for bankruptcy prediction,” *1990 IJCNN Int. Jt. Conf. Neural Networks*, pp. 163–168 vol.2, 1990, doi: 10.1109/IJCNN.1990.137710.

[15] X. Glorot and Y. Bengio, “Understanding the difficulty of training deep feedforward neural networks,” in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, 2010, vol. 9, pp. 249–256, [Online]. Available: https://proceedings.mlr.press/v9/glorot10a.html.

[16] Y. N. Dauphin, R. Pascanu, C. Gulcehre, K. Cho, S. Ganguli, and Y. Bengio, “Identifying and attacking the saddle point problem in high-dimensional non-convex optimization,” *Adv. Neural Inf. Process. Syst.*, vol. 4, no. January, pp. 2933–2941, 2014.

[17] S. Dargan, M. Kumar, M. R. Ayyagari, and G. Kumar, “A Survey of Deep Learning and Its Applications: A New Paradigm to Machine Learning,” *Arch. Comput. Methods Eng.*, vol. 27, no. 4, pp. 1071–1092, 2020, doi: 10.1007/s11831-019-09344-w.

[18] P. P. Brahma, D. Wu, and Y. She, “Why Deep Learning Works: A Manifold Disentanglement Perspective,” *IEEE Trans. Neural Networks Learn. Syst.*, vol. 27, no. 10, pp. 1997–2008, 2016, doi: 10.1109/TNNLS.2015.2496947.

[19] P. P. Shinde and S. Shah, “A Review of Machine Learning and Deep Learning Applications,” in *2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA)*, Aug. 2018, pp. 1–6, doi: 10.1109/ICCUBEA.2018.8697857.

[20] S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory,” *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997, doi: 10.1162/neco.1997.9.8.1735.

[21] J. Hernández, D. López, and N. Vera, “Primary user characterization for cognitive radio wireless networks using long short-term memory,” *Int. J. Distrib. Sens. Networks*, vol. 14, no. 11, p. 155014771881182, Nov. 2018, doi: 10.1177/1550147718811828.

[22] A. M. and G. H. Alex Graves, “Speech Recognition with Deep Recurrent Neural Networks , Department of

Computer Science, University of Toronto," *Dep. Comput. Sci. Univ. Toronto*, vol. 3, no. 3, pp. 45–49, 2013, [Online]. Available:
https://ieeexplore.ieee.org/stampPDF/getPDF.jsp?tp=&arnumber=6638947&ref=aHR0cHM6Ly9pZWVleHBsb3JlLmllZWUub3JnL2Fic3RyYWN0L2RvY3VtZW50LzY2Mzg5NDc/Y2FzYV90b2tlbj1OQUo1VFJxWk5JRUFBQUFBOmtPZmdDbS00NGhqaGI2N3dMd2JrU3lSaEdJREhBWnpMSkxoT201Um5YMXR0S0poUDAtM2hkbT.

[23] 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 *Proceedings - International Conference on Software Engineering*, May 2014, no. 1, pp. 767–778, doi: 10.1145/2568225.2568263.

[24] W. Maalej, Z. Kurtanović, H. Nabil, and C. Stanik, "On the automatic classification of app reviews," *Requir. Eng.*, vol. 21, no. 3, pp. 311–331, Sep. 2016, doi: 10.1007/s00766-016-0251-9.

[25] S. Moghaddam, "Beyond Sentiment Analysis: Mining Defects and Improvements from Customer Feedback," 2015, pp. 400–410.

[26] T. Johann, C. Stanik, A. M. B. Alizadeh, and W. Maalej, "SAFE: A Simple Approach for Feature Extraction from App Descriptions and App Reviews," in *Proceedings - 2017 IEEE 25th International Requirements Engineering Conference, RE 2017*, Sep. 2017, pp. 21–30, doi: 10.1109/RE.2017.71.

[27] S. Scalabrino, G. Bavota, B. Russo, M. Di Penta, and R. Oliveto, "Listening to the Crowd for the Release Planning of Mobile Apps," *IEEE Trans. Softw. Eng.*, vol. 45, no. 1, pp. 68–86, 2019, doi: 10.1109/TSE.2017.2759112.

[28] D. Martens and W. Maalej, "Towards understanding and detecting fake reviews in app stores," *Empir. Softw. Eng.*, vol. 24, no. 6, pp. 3316–3355, Dec. 2019, doi: 10.1007/s10664-019-09706-9.

[29] N. Aslam, W. Y. Ramay, K. Xia, and N. Sarwar, "Convolutional neural network based classification of app reviews," *IEEE Access*, vol. 8. Institute of Electrical and Electronics Engineers Inc., pp. 185619–185628, 2020, doi: 10.1109/ACCESS.2020.3029634.

[30] Z. Qiao, A. Wang, A. Abrahams, and W. Fan, "Deep Learning-Based User Feedback Classification in Mobile App Reviews," 2020. [Online]. Available: https://aisel.aisnet.org/sigdsa2020.

[31] C. M. Suneera and J. Prakash, "Performance Analysis of Machine Learning and Deep Learning Models for Text Classification," *2020 IEEE 17th India Counc. Int. Conf. INDICON 2020*, 2020, doi: 10.1109/INDICON49873.2020.9342208.

[32] L. Salmela, N. Tsipinakis, A. Foi, C. Billet, J. M. Dudley, and G. Genty, "Predicting ultrafast nonlinear dynamics in fibre optics with a recurrent neural network," *Nat. Mach. Intell.*, vol. 3, no. 4, pp. 344–354, 2021, doi: 10.1038/s42256-021-00297-z.

[33] Q. H. Vo, H. T. Nguyen, B. Le, and M. Le Nguyen, "Multi-channel LSTM-CNN model for Vietnamese sentiment analysis," *Proc. - 2017 9th Int. Conf. Knowl. Syst. Eng. KSE 2017*, vol. 2017-January, pp. 24–29, 2017, doi: 10.1109/KSE.2017.8119429.

[34] A. Larasati, A. M. Hajji, and A. Dwiastuti, "The relationship between data skewness and accuracy of Artificial Neural Network predictive model," *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 523, no. 1, 2019, doi: 10.1088/1757-899X/523/1/012070.

[35] S. Sukhbaatar, J. Bruna, M. Paluri, L. Bourdev, and R. Fergus, "Training convolutional networks with noisy labels," *3rd Int. Conf. Learn. Represent. ICLR 2015 - Work. Track Proc.*, pp. 1–10, 2015.

[36] H. Song, M. Kim, D. Park, Y. Shin, and J.-G. Lee, "Learning From Noisy Labels With Deep Neural Networks: A Survey," *IEEE Trans. Neural Networks Learn. Syst.*, pp. 1–19, 2022, doi: 10.1109/TNNLS.2022.3152527.

[37] P. J. Khiabani, M. E. Basiri, and H. Rastegari, "An improved evidence-based aggregation method for sentiment analysis," *J. Inf. Sci.*, vol. 46, no. 3, pp. 340–360, 2020, doi: 10.1177/0165551519837187.

[38] B. Amini, R. Ibrahim, M. S. Othman, and H. Rastegari, "Incorporating scholar's background knowledge into recommender system for digital libraries," in *2011 Malaysian Conference in Software Engineering*, 2011, pp. 516–523, doi: 10.1109/MySEC.2011.6140721.