Received January 2, 2022, accepted February 25, 2022, date of publication May 2, 2022, date of current version May 12, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3171579

Prediction of Application Usage on Smartphones via Deep Learning

ABDULRAHMAN ALRUBAN
Department of Information Technology, College of Computer Sciences and Information Technology, Majmaah University, Majmaah 11952, Saudi Arabia
National Center for Artificial Intelligence (NCAI), Saudi Data and Artificial Intelligence Authority (SDAIA), Riyadh 11543, Saudi Arabia
e-mail: a.alruban@mu.edu.sa

This work was supported by the Deanship of Scientific Research at Majmaah University under Project 5453.

ABSTRACT Smartphones have proven to be a transformative tool that helps users perform various tasks such as online banking, chatting, sending an email or SMS, and online shopping. However, with the growing number of available applications and people downloading new applications at a high rate, managing the performance of such a large number of applications will increasingly become a concern, which makes managing smartphones' screens and folders complicated which led the user to spend time finding those application to perform actions such as chatting or sending a message or even finding his favorite game at evening time may take few seconds to reach. By letting the smartphone learns the user's behavior and their interactions, to predict which app the user looking for at a specific time after a certain sequence of actions. This saves users time and increases the level of usability. This paper investigates to what extent the usage of those applications can be predicted. The proposed methodology utilizes a deep learning algorithm (long short-term memory) to accurately predict the probability of a given application to be used by the smartphone user after a sequence of applications usage. The experimental result shows that forecasting those applications' usage performance can be correct with an achieved accuracy of approximately 80%.

INDEX TERMS Applications usage, deep learning, GRU, LSTM smartphone, usability.

I. INTRODUCTION

The use of smartphones in our daily lives has grown steadily, especially with hardware improvement and effective programming capabilities. More specifically, smartphones are used to perform activities, such as sending emails, transferring money via mobile Internet banking, making calls, texting, surfing the Internet, viewing documents, storing medical, confidential, and personal information, shopping online, and playing games. For instance, Deloitte Mobile Consumer Survey (2018) demonstrates that 95% of the owner of smartphones use their devices daily in the UK. Furthermore, the UK market penetration by device type, which showed clearly that mobile, is gradually increased compared with a laptop (Deloitte Mobile Consumer Survey, 2018). In addition, with the advent of the increasing features of smartphones, mobile applications have been evolved and become ubiquitous and widely used by smartphone owners. According to Statista (2019), app downloads will surpass 258.2 billion in 2022. Furthermore, the number of available apps in the Google Play app store was about 2 million in 2018, while 1.962.576 in Apple. Interestingly, Statista (2019), in 2020, mobile apps are projected to generate 188.9 billion U.S. dollars in revenues via app stores. Using genuine heuristics is highly relevant as it can potentially reveal more specific usability matters related to the application's domain (Inostroza *et al.*, 2016).

The growth in smartphone usage has led to increased user concerns regarding privacy and security (Lamiche *et al.*, 2018). More specifically, the traditional authentication mechanisms for smartphones such as PIN, Pattern, and Password are suffered from some issues; for instance, secret knowledge-based systems are vulnerable to be sharable, forgotten, easy to guess, and trying to remember and manage a significant number of different accounts as well (Mahfouz *et al.*, 2017). In addition, after the point of entry, using techniques such as a PIN or password, the device user can perform almost all tasks, of different risk levels, without having to re-authenticate periodically to re-validate the user's identity. Furthermore, the current point-of-entry authentication mechanisms consider all the applications on a mobile device to have the same level of importance and so do not apply any further access control rules (Ledermuller & Clarke,

The associate editor coordinating the review of this manuscript and approving it for publication was Li He.

2011). As a result, with the rapid growth of smartphones for use in daily life, securing the sensitive data stored upon them makes authentication paramount.

In this context, behavioral profiling (service utilization) attempts to identify and discriminate users based upon how they interact with applications and services, specifically, which applications they access, time of day, and how long (Clarke, 2011). The advantages of using this mechanism are gathering user data inthe background without requiring any dedicated activity by regularly checking user behavior to provide continuous monitoring for smartphone protection.

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

- Provides a review of the most recent literature in predicting smartphone applications usage.
- To introduce our approach and build the prediction model, we propose a predictable model that can predict the probability of a given application used by the smartphone user after a sequence of application usage.
- To introduce how we do our evaluation and experimentation, which resulted in the forecasting of those applications usage performance can be correct with an achieved accuracy of approximately 80%.

The remainder of the paper is structured as follows. The following section presents related work and the state of the art of smartphone behavior profiling biometrics. This is followed by an outline of a novel prediction subsequent application usage by smartphone users, including the data collection phase and experimental methodology in section 3. Section 4 presents the experimental results, and section 5 concludes the paper.

II. RELATED WORK

With the rising usage of machine learning and Artificial intelligence (AI), the researchers explore the new ear of predicting the next app using machine learning algorithms and contextual models. This section will review the related work of smartphone app prediction.

A. APP PREDICATION USAGE

In the literature, the researchers have done explicit work on app prediction among users by studying their behavior while using their smartphones. This section will emphasize prediction using different methods of app prediction. Shin *et al.* (2012) collected a dataset from 23 users who use Android System; the researchers proposed a context model for app prediction based on Naïve Bayes Model (NB). Shin *et al.* identified that several factors such as an hour of the day, last application, and cell ID had an essential effect on the prediction. This model provided the best prediction results for up to 10 apps "in terms of accuracy and the number of users impacted" (Shin *et al.*, p. 182). At the same time, Zou *et al.* (2013) used three Bayesian models relying on time and history context by implementing the most frequently used model (MFU) and Latest Used Model (LU). Zou *et al.* employed 80 users' records with 5589 records for each user, so NB reached the highest accuracy among MFU and LU by 85%.

Furthermore, Huang *et al.* (2013) conducted a study similar to Shin *et al.* study (2012) by predicting the user's last app usage, location, and time. Huang *et al.* concluded that the pattern of the users' app could be learned through contextual information. Moreover, there is a strong relationship between prediction accuracy and used apps. Finally, Linear Model is more efficient than NB in the app prediction, where it reached 79%, 75%, respectively.

From a different perspective, Parate *et al.* (2013) introduced a method to predict app usage among users to predict their subsequent app usage in Android smartphones. Parate *et al.* reached 95% accuracy using two models. The first model is App Prediction by Partial Match (APPM), which learns the app probability distribution to predict the location. The second model is Time Till Usage (TTU), "which utilizes the learned distribution of time spent before app use to estimate appropriate time for prefetch in a bandwidth cost-aware manner" (Parate *et al.*, 2013, p. 276). The model used in Parate *et al.*, the study was unique compared to other studies mentioned before due to the usage of APPM.

On the contrary, Baeza-Yates *et al.* (2015) studied the prediction of the following app comprehensively as they conducted a large experiment depending on the information recorded into the log form. Baeza-Yates *et al.* (2015) have 60 million log samples extracted from 200,000 anonymized users; The researchers collected a total of 70,000 different apps in January 2014. Baeza-Yates *et al.*, (2015) achieved a high precision of 90.2% using Parallel Tree Augmented Naïve Bayesian Network (PTAN) and Tree Augmented Naïve Bayes (TAN) in-app prediction. From our point of view, we recommend Baeza-Yates *et al.*, (2015), among other studies discussed in this paper, due to the large dataset collected in their research that leads to the high accuracy of the app prediction.

Furthermore, Baeza-Yates *et al.* proposed a unique method by merging TAN and PTAN to have an efficient app prediction model. However, Rahnamoun *et al.* (2016) ascertained the Learning Automata (LA) model, which learns and predicts users' app usage; it improves the prediction by updating the probability of each App. LA defined as, "Formally, Learning Automata (LA) is defined by a quadruple (O, R, Q, F), where O is a set of outputs or actions that are chosen by the automaton in time instant, R is a set of reinforcement values that may be discrete or continuous, Q is the set of the internal state of automaton, and F: $Q \times O \times R \rightarrow Q$ is the learning algorithm and also a mapping that updates internal state of an automaton" (Rahnamoun *et al.*, 2016, p. 3). Rahnamoun *et al.* used Finite Action-set Learning Automata (FALA). It is also defined as quadruple like LA, a variable structure type.

Cao and Lin (2017) conducted a comprehensive literature review in Appin-app predication usage divided into two main contributions. The researchers explained the app usage patterns, explored the challenges around this topic,

TABLE 1. Studies summary.

Study	Method	Dataset	#Users	#Apps	Accuracy (%)
Shin 2012	NBM	n/a*	23	74	78
Liao 2013	NBM	n/a*	80	41	80
Shin et al., (2013)	BN	Android market	23	n/a*	79
Zou et al., (2013)	BN, MFU, & LU	5589	80	n/a*	85
Huang et al., (2013)	BM & LM	n/a*	28	43	79
Parate et al (2013)	APPM & TTU	S-controlled users	7630	n/a*	95
Srinivasan et al., (2014)	Tizen platform	n/a*	106	n/a*	80
Baeza-Yates et al., (2015)	PTAN & TAN	60 million	200K	70 K	90
Ricardo 2015	PTAN	n/a*	480	n/a*	90
Yu et al., (2017)	Adaboost	n/a*	182	n/a*	98
Fang et al., (2018)	NBM	30 million	1083	588	n/a*
Zhao, J., et al., (2018)	LM, BN, & LU	46,434,380	130	5000	85
Zhao, S. et al. (2018)	Sequence vector	46,434,380	10,360	5000	84.74

Note: n/a = not applicable

and presented some recommendations (Cao & Lin, 2017). Cao and Lin (2017) divided smartphone features into explicit and implicit. Implicit is the smartphone's statistics, clear includes the following information: (1) mode, e.g., airplane mode, (2) location, and (3) time-information, e.g., week of the month. On the other hand, Fang *et al.* (2018) recommended a unique method to predict the next app, incorporating sequential behavior and semantic information to predict the user's next app click. Fang *et al.* created a model using a crawler system which they obtained a descriptive text for each based on Scrappy and Redis. Furthermore, the researchers used the User-based Collaborative Filtering (UCF) method, which showed the best performance among all other ways – Singular Value Decomposition (SVD), Most Frequently Used (MFU), Most Recently Used (MRU), and Bayesian Network (BN). Fang *et al.* asserted that UCF had the best prediction with small k values. On the contrary, Yu *et al.* (2017) provided a unique method that predicts the user's next position by presuming the user's next activity. The researchers model the user activity pattern then indicates the following status (Yu *et al.*, 2017). Yu *et al.*, reached an accuracy of 98% using the AdaBoost model. The researchers also determined a relationship between the number of app clicks and prediction, so more clicks mean challenging to predict (Yu *et al.*, 2017).

Zhao, S. al., (2018) collected the Mobile Dataset Challenge dataset data, containing 200 features from the data collecting Campaign. They processed 5000 records extracted from 130 users. Zhao, S. al., predicted users' next app using the linear model (LM) and the Bayesian network (BN). The researchers asserted that the current app prediction is influenced by the users' last two apps, which leads to using social apps or calling friends (Zhao, S. al., 2018). However, Zhao, S. al., reached 85% using LM, which is less than the accuracy Yu *et al.*, reached in their study, which was 98%. Similarly, Zhao, J. *et al.*, (2018) created a novel model called AppUsage2Vec using the information of 10,360 users and 46,434,380 records. Zhao, J. *et al.* compared the recall of AppUsage2Vec with other approaches such as NB and Markov chain. The recall @ 5 for AppUsage2Vec, NB, and Markov chain is 84.47%, 33.54%, and 77.05% respectively. Thus, Zhao, J. *et al.*, the exclusive model had higher recall among other approaches as mentioned. Recently, Mahbub *et al.*, (2018) approach for app prediction is distinct from other approaches which use the top applications to authenticate. The researchers used a list of applications for verification purposes. Mahbub *et al.* (2018) applied a Modified Edit-Distance (M-ED) algorithm and Modeled the Person Authentication using Trace Histories (PATH). Mahbub *et al.* conducted their work on a large scale of data to investigate the authentication problem for the University of Maryland. Mobility Markov Chains or Hidden Markov Models (HMM) are usually employed for data verification. Still, the researchers employed the Markov Chain (MC)-based Verification, Marginally Smoothed HMM (MSHMM), and HMM with Laplacian Smoothing (HMMlap). Using these enhanced models will improve the usability of any unforeseen events (Mahbub *et al.*, 2018).

Piferrer Torres (2018) emphasized that continuous authentication uses behavioral biometrics, which performs once the user controls the device. Piferrer Torres considers such prediction usage to evaluate and identify the current user. Piferrer Torres analyzed sequences of application usage to identify a user to prevent non-legitimate access. The researchers analyzed app usage patterns using Recurrent Neural Networks (RNN). Applying RNN methodology allowed the study to evaluate a long application sequence. Also, the researcher identified the legitimate or non-legitimate user, which was unique because it placed the user from their activity information using their devices. Piferrer Torres (2018) followed previous work on app prediction performed by Alzubaidi (2017). Alzubaidi studied application usage patterns by focusing on continuous authentication. Alzubaidi has developed a new dataset of Android users; the author derived several features from their app usage. Alzubaidi has achieved outstanding results in user recognition. Alzubaidi constructed a library of machine learning algorithms for data mining tasks and Weka. Alzubaidi calculated the impact factor for each app/user. Alzubaidi examined legitimate user and user recognition for short and long training time. Alzubaidils used a Market-basket analysis and Apriori algorithm.

Han *et al.* (2018) focused on studying the case of the cold start in the mobile phone app and benefit from prediction to enhance user experience on these apps. Han *et al.* designed a collaborative filtering algorithm (CF), which provides a forecast for developing a collaborative filtering algorithm (CF) that predicts app cold start for new users using mobile phones. It integrated both App preferences and App usage via the conditional combination. Han *et al.* propose a more appropriate App launching than another traditional method. Han's model provided better performance results than other methods. However, Han suggested combining both cold start and warm start app. Han's provides a beneficial app prediction model because it supports new users with a flexible prosperous app cold start. It also provides a more accurate prediction because it avoids Long Tail by using the weight optimization strategy (Han *et al.*, 2018). It had high-performance predictions using the cold start app, especially for new users (Han *et al.*, 2018).

Furthermore, Xu *et al.* (2018) created the subsequent app usage by developing a model based on Long Short-term Memory (LSTM). LSTM is an extension of the recurrent neural network. The researchers collected data from 42 students who used Android systems. The precision Xu *et al.* achieved was 0.80 in-app usage prediction; further investigation needs to be conducted to increase the performance of the model.

To conclude, this study found that most of the researchers used NB (Shin *et al.*, 2012; Zou *et al.*, 2013; Fang *et al.*, 2018) or MFU (Shin *et al.*, 2012; Zou *et al.*, 2013; Fang *et al.*, 2018) for app prediction. Besides, other methods for app prediction, i.e., APPM, TTU, or AdaBoost. However, the performance of a classifier is not stable and fluctuant, which may depend heavily on the used dataset. For example, Baeza-Yates *et al.* (2015) reached 90% accuracy while Zou *et al.* (2013) reached 85% due to the size of the dataset and method used. Thus, different techniques used in prediction produced variance accuracy. In the next part, we summarize all the datasets used on this topic.

B. PREVIOUS STUDIES DATASET ON NEXT APP PREDICTION

Those researchers summarised the studies of smartphone app prediction using different types of datasets, users, and apps. Android was the most OS used in-app projection because it does not require any jailbreak such as iOS system in iPhone (Cao & Lin, 2017). This research comprehends the methods used, the number of users, predicted applications, datasets used, and accuracy/performance achieved.

III. OUR PROPOSED APPROACH

A. DATASET

A real applications usage dataset is necessary to provide scientific rigor and a basis for evaluating the application usage pattern. Furthermore, it will identify whether the application usage pattern could maintain a reliable assumption for predicting the smartphone user's next application. Therefore, this study was recruited 76 participants (18 years or older) at the University of Plymouth from February to July 2017. Ethical approval for this research project was obtained from the university's Research Ethics Committee to fulfill University of Plymouth research ethics requirements. All the participants were 18 years or older and were asked to read and sign a consent form and information sheet regarding data collection before starting the experiment. In addition, the research and data were conducted and stored within the Centre for Security, Communications and Network Research at the university premises. Although the study collected applications' logs/metadata, no sensitive material was involved. Participants were asked to use their smartphones generally for at least one month.

After one month, since they were committed to the experiment, participants were asked to provide their devices to perform data extraction. In addition, the investigation was carried out on only those individuals who use Android-based mobile phones. Only applications access and actions metadata were collected. For the scope of this paper, only applications usage (access) is analyzed. A script code was developed to automate the extraction of log files from a backup file of participants' devices utilizing the Android Debug Bridge (ADB). ADB is a command-line tool that allows communication between a connected Android device and a computer (Android, 2018). For each examined application, the backup file was extracted, and then the developed script retrieved those metadata stored in each application's local database (SQLite), as illustrated in TABLE 2. A total of 3,015,339 actions were accumulated. In turn, this dataset leads to a better understanding of generalized outcomes, positively impacting the conclusions drawn from the proposed approach.

Some applications used by the individuals, such as Facebook, online mobile banking, and Chrome, are fully encrypted. There was no means of collecting user data without compromising the user's privacy by asking the participant to root their device. For this reason, to protect the users' privacy, only 12 applications were included in this study, these are Phone Calls, native SMS, Download, YouTube, WhatsApp, Android Browser, Google Play, Email, Viber, Google Photo, native Android Camera, and Yahoo mail.

B. DATA PRE-PROCESSING

After acquiring the raw application usage data, normalization and standardization transformation were examined for transforming the raw data. Transforming the raw data into a rescaled value makes training predictable algorithms faster and reduces the chances of being stuck in local optima (Jason Brownlee, 2019). As data, normalization ensures that each feature is treated equally when applying supervised learners. In machine learning, we can handle various types of data, e.g., audio signals and pixel values for image data, and this data can include multiple dimensions. Feature

TABLE 2. **Applications collected from users' mobile phones.**

Id.	Application	Package name	Database
1	Phone Call	com.sec.android.provider.logs provider	logs
2	SMS	com.sec.android.provider.logs provider	logs
3	Downloading	com.android.providers.downloads	downloads
4	YouTube	com.google.android.youtube	history
5	WhatsApp	com.whatsapp	msgstore
6	Browser	com.sec.android.app.sbrowser	SBrowser_Tabs
7	Google Play	com.android.vending	localappstate
8	Email	com.android.email	EmailProvider
9	Viber	com.viber.voip	viber_data; viber_messages
10	Google Photo	com.google.android.apps.photos	gphotos0_local_media
11	Camera	com.android.providers.media	external_Images; external_video
12	Yahoo mail	com.yahoo.mobile.client.android.mail	mailsdk_messages

standardization makes the values of each feature in the data have zero-mean (when subtracting the mean in the numerator) and unit-variance. This method is widely used for normalization in machine learning algorithms (e.g., support vector machines, logistic regression, and artificial neural networks). The samples are normalized by scaling the input vectors individually to the unit norm (vector length). The other transformation approach standardizes the features by removing the mean and scaling to the unit variance.

C. NEXT APPLICATION CLICK PREDICTION MODEL

The predictable model aims to learn a machine-learning algorithm to predict the next application the user will click/use with high probability. As we have 12 applications to predict (one is predicted at a time), the investigated problem can be treated as a classification problem, predicting a category (class). Two main variations of LSTM are used, unidirectional and bidirectional. FIGURE 1 illustrates bidirectional RNN in which two independent RNNs are combined. The input sequence is fed in normal time order for one network and reverse time order for another. The outputs of the two networks are usually concatenated at each time step, though there are other options, e.g., summation. This structure allows the networks to have both backward and forward information about the sequence at every time step.

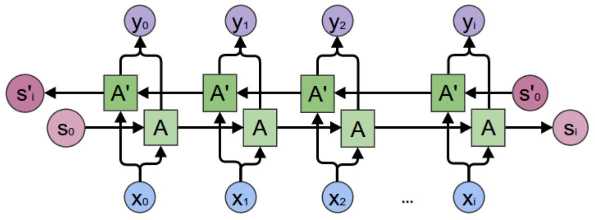


FIGURE 1. **Generic bidirectional recurrent neural network.**

Each network unit (represented as A and A') is a recurrent unit. A slightly more dramatic variation on the LSTM is the Gated Recurrent Unit, or GRU, introduced by Cho, *et al.* (2014). It combines the forget and input gates into a single "update gate." It also merges the cell and hidden states and makes other changes, as illustrated in FIGURE 2.

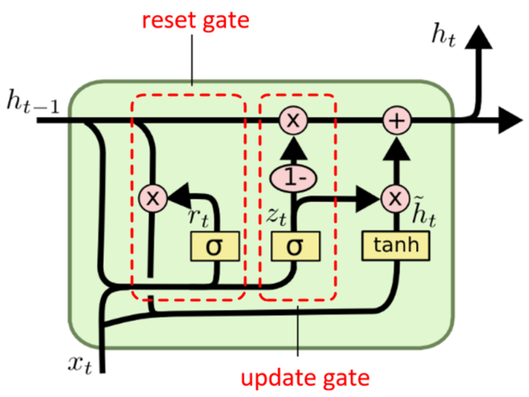


FIGURE 2. **GRU recurrent unit.**

FIGURE 3 explains the shapes and arrow types that control the data flow in the GRU unit. Each line carries an entire vector, from the output of one node to the inputs of others. The pink circles represent pointwise operations, like vector addition, while the yellow boxes are learned neural network layers. Lines merging denote concatenation, while a line forking denotes its content being copied and the copies going to different locations (Olah, 2015).

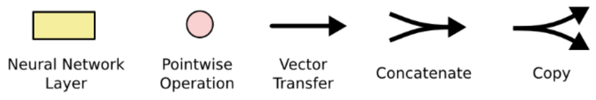


FIGURE 3. **Shapes and arrow types of a GRU unit.**

As GRU is a recurrent model, the current time prediction depends on all past time inputs. For each layer, the GRU processes at time *t* by computing the following equations:

$$z_t = \sigma\left(W_z[h_{t-1}, x_t]\right) \quad (1)$$

$$r_t = \sigma\left(W_r[h_{t-1}, x_t]\right) \quad (2)$$

$$\tilde{h}_t = \tanh\left(W[r_t * h_{t-1}, x_t]\right) \quad (3)$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t \quad (4)$$

where σ is the sigmoid function, z and r are gates, while h are hidden states. Unidirectional LSTM (GRU) only uses past information. BiLSTM can also take advantage of future information. In each BiLSTM layer, there are a forward pass and a backward pass.

The model architecture used in this study consists of five layers, as illustrated in FIGURE 4. The first layer has five inputs as the sequence length is five, in which by giving five consecutive application usage, the model should predict the sixth one. This layer is followed by a dropout layer with a ratio of 0.20. Dropout removes some of the hidden nodes according to the predefined ratio Srivastava *et al.* (2011). It is found that it helps make the neural network-based models generalize better. The network output layer activation is processed with a softmax function. Softmax assigns decimal probabilities to each class in a multi-class problem. Those decimal probabilities must add up to 1.0. This additional constraint helps training converge more quickly than it otherwise would Janocha *et al.* (2017). The application with the highest probability will be determined as the predicted class among other applications.

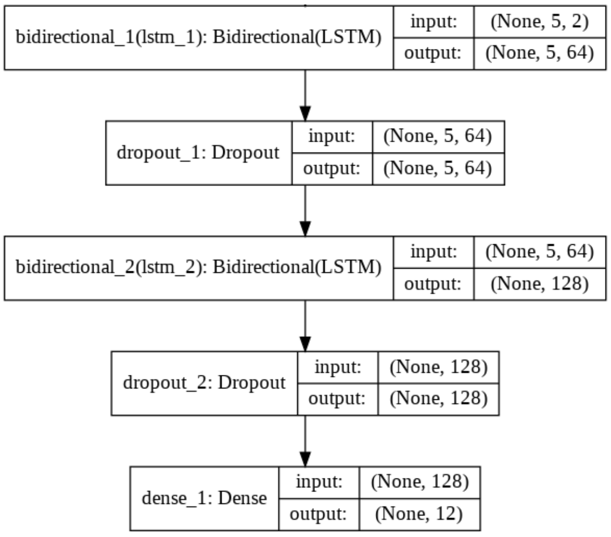


FIGURE 4. **Model architecture.**

D. HYPER-PARAMETERS

Batch size is 100. Adam optimization algorithm is used instead of the traditional stochastic gradient descent procedure to update network weights iterative based in training data. An early stop is applied for a total of 100 steps' training. The initial learning rate is 0.01. When it converges, the training will stop. We use the checkpoint of the best validation accuracy to evaluate the test accuracy (Zeng *et al.*, 2019).

IV. PERFORMANCE ANALYSIS

A. EVALUATION METRICS

Using only an accuracy metric does not fully reveal overlapping and false-positive rates among the classes or the predicted application names, as it computes the ratio of correct predicted labels to the total examined sample, which becomes insensitive to unbalanced classes. Therefore, an F score is computed, which is interpreted as the weighted mean of precision and recall. An F score of 1.0 is the highest, and the lowest score of 0.0 is the lowest. It worth mentioning that it is common to use F score for binary classification problems; however, adapting the metric for a multiclass problem is achieved using one label versus all other labels. In which, the relative contribution of precision and recall to the F score is equal. Equation 1 Equation 2 and Equation 3 explains how the precision, recall and F score are calculated respectively.

Equation 1: Illustrates how precision is calculated using true positive and false positive values.

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive} \tag{5}$$

Equation 2: Illustrates how the recall is calculated using the true positive and false negative values.

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative} \tag{6}$$

Equation 3: Illustrates how the F score is calculated using the precision and recall values.

$$Fscore = 2 \cdot \frac{Precision.Recall}{Precision + Recall} \tag{7}$$

B. EXPERIMENTAL ENVIRONMENT AND TESTBED

The experiment and analysis of this study were mainly conducted using Google Colaboratory (Colab, 2019).

Colaboratory is a research tool for machine learning research projects. It is a Jupyter notebook environment that requires no setup to use. For modelling the deep learning network, Keras is utilised as it is a high-level neural networks API, written in Python and capable of running on top of TensorFlow 2.0 (TensorFlow, 2019). We strongly recommend researchers to leverage Google Colaboratory features as it developed with a focus on enabling fast experimentation and enable collaborations among researchers along with facilitates documenting the developed code boxes/functions on the go.

C. EXPERIMENTAL ANALYSIS AND RESULTS

FIGURE 5 illustrates the density of those applications for all users in the collected dataset. Email is mostly used at late morning-midday as it is tipping the peak usage around 10-13 o'clock. This is not surprising as most people are at work/university at such time in which they are checking/sending emails. In contrast, YouTube application seems mostly used at evening time as it peaks around 18-20 o'clock. Overall, most of these application usages are gradual increases from midday until bedtime (21-22) where the usage declines. However, not all users have the same pattern as some of which uses a number of the selected application while others use the 12 applications. To show how different users have different usage patterns, FIGURE 6 shows two users usages timelines among the 12 applications. User number 1 (a) uses all the examined applications while the second user does not use application 1, not 2. In addition, they vary in their timeline usage.

The proposed model is trained independently for each user. FIGURE 6 illustrates the model accuracy for both the train and validation sets in predicting the application the user will use—for a selected user (e.g. user 52). This specific user is selected randomly to show the train and validation sets accuracy over epochs. After epoch number 50, the model started to overfit. By leaving the learning process of the model to continue longer, this could lead to an overfitting issue–in

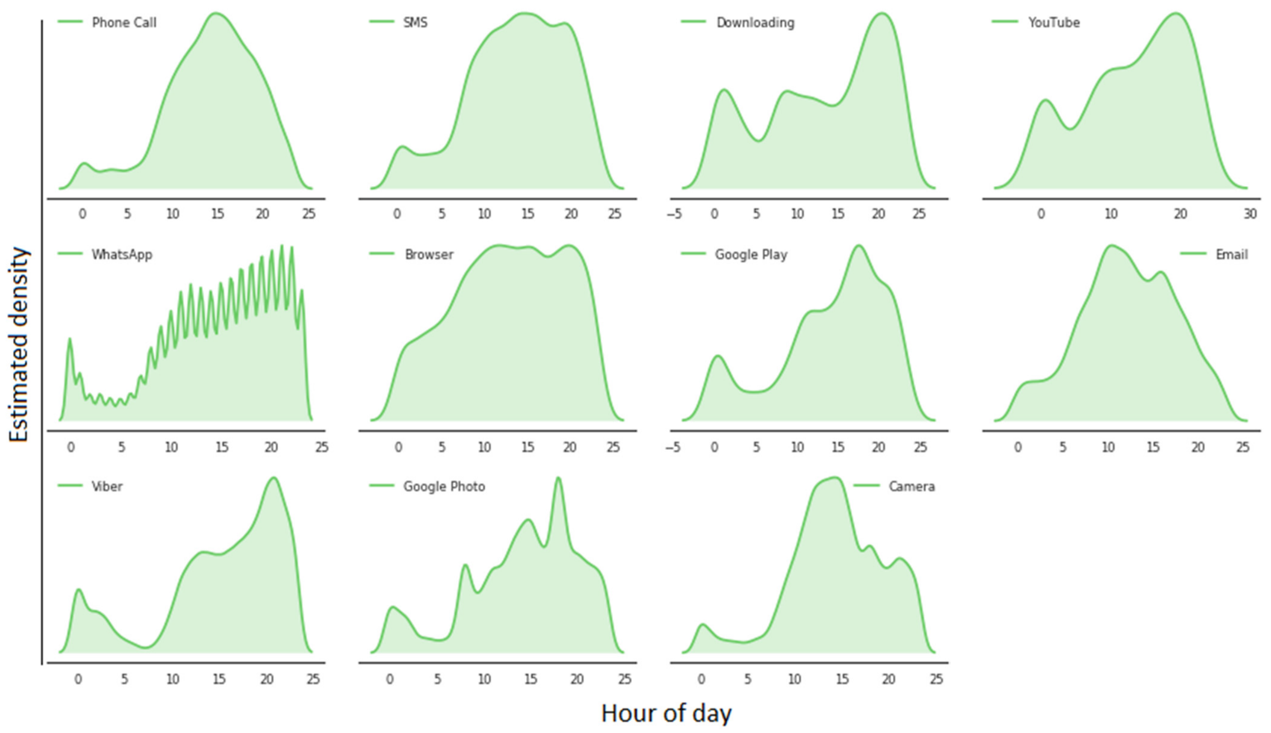
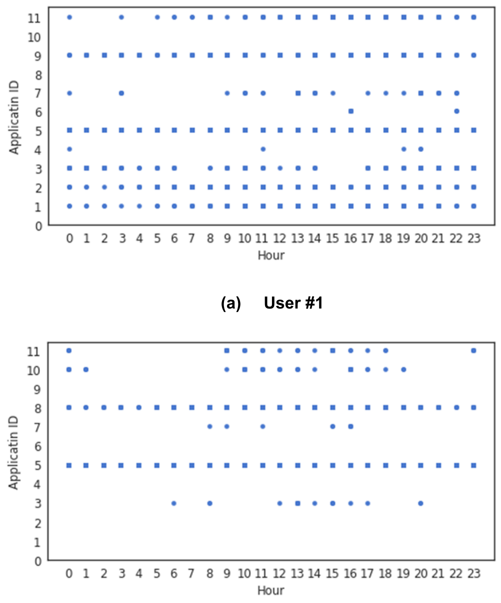


FIGURE 5. Time of day apps density estimation for all users dataset.



(b) User #2

FIGURE 6. Application timeline usage–(a) illustrates user number 1 in the dataset and (b) illustrates user number 2.

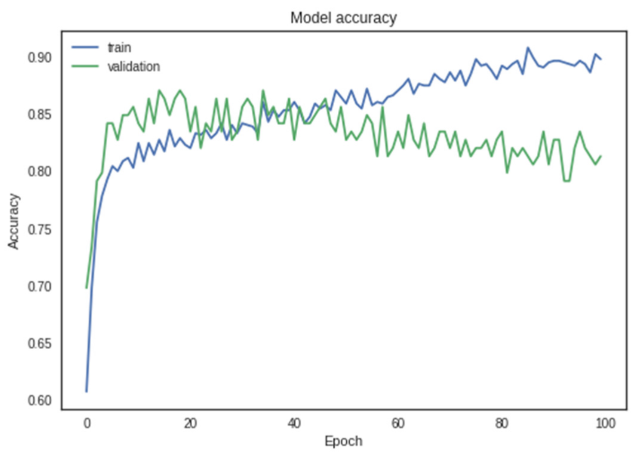


FIGURE 7. Train and validation accuracy.

which the neural network is closely fitted to the training set that it is difficult to generalize and make predictions for new data (Zhang, 2018). Although the dropout technique is used after each layer, preventing overfitting is not always an easy and obvious task. Therefore, the early stopping approach is used in which it stops the training process when the monitored accuracy has stopped improving epoch after another.

Likewise, FIGURE 7 illustrates the model loss for both the train and validation sets in predicting the user's application. In this case, the *sparse_categorical_crossentropy* loss function as presented in the following equation (i.e. *objective function*) is used as the predicted target of the network is treated as an integer number that corresponds to applications names.

$$J\left(\boldsymbol{W}\right) = -\frac{1}{N}\sum\nolimits_{i=1}^{N}\left[y_i \log\left(\hat{y}_i\right) + (1-y_i)\log(1-\hat{y}_i)\right] \quad (8)$$

where,

- w refers to the model parameters, e.g. weights of the neural network
- y is the true label
- $\tilde{y}_i$ is the predicted label

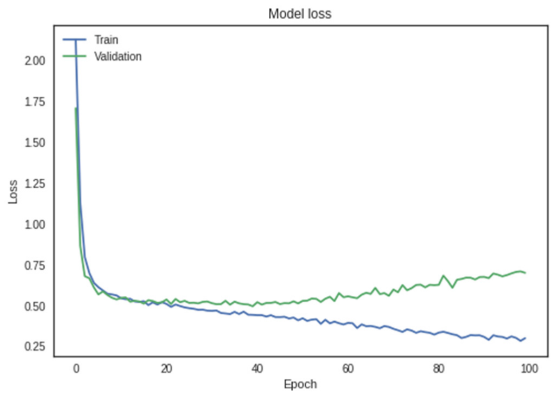


FIGURE 8. The model loss during training and validation.

The overall accuracy of the app's prediction ranges between 60% to 90% for users. From FIGURE 9 **Error! Reference source not found.,** email has the highest prediction rate with an accuracy of around 95%. It looks that Phone calls and SMS have almost the same rates.

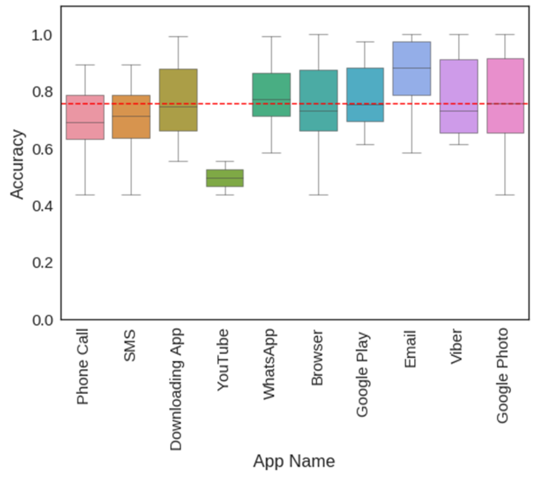


FIGURE 9. Overall accuracy per examined application.

In terms of individual users' prediction performance, FIGURE 10 illustrates the accuracy for all users for all these 12 applications. The average accuracy is nearly 80%, similar to the averaged application predictions. The finding provides evidence that some of these users application usage patterns can be predicted with high confidence in which can result in.

Although the experimental results have shown that the proposed approach has achieved an overall accuracy of around 80%, there are many limitations and investigations that need to be examined. For example, the study has focused on only 12 selected applications in which these are the most used applications by the study sample. However, application usage predication should be utilised to include and predict what the user uses typically without flittering the applications. In which predicting the application usage or what the user will open next can be defined as n class problem. Where n is the number of applications that the smartphone's users have installed and frequently uses in the device. Although

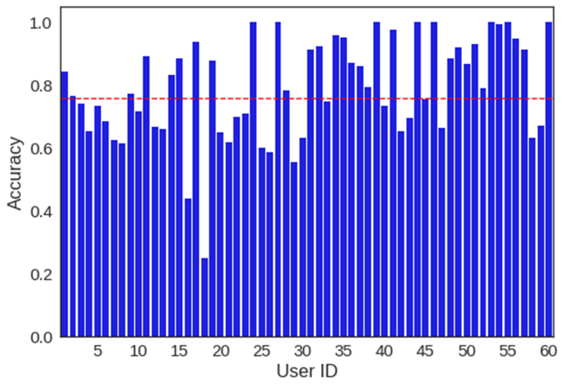


FIGURE 10. Overall achieved accuracy per user.

measuring the frequency of the application usage is not a straightforward task, but it is an aspect that needs to be considered as some of the existing applications (installed) are rarely used such as those comes pre-installed with the device.

Further to that, the evaluation of this study was mainly conducted offline (using an experiential environment). It has not been thoroughly tested in a live device (smartphone in this case) to measure other operational metrics, such as computational overheads, memory consumption, and the time required for the whole pipeline to be completed, starting from acquiring usage patterns to pre-processing, and finally inferencing, where the next app is predicted. In addition, the collected dataset was acquired, including Android-based smartphones. Investigating other devices such as iOS-based devices could reveal how similar/different the users' usage patterns between such operating systems are. Future work could also explore other factors, such as identifying the minimum number of seconds and samples required per individual to train a user-dependent predictable model that can successfully match a given pattern usage sequence with the application that the user will use.

V. CONCLUSION

The increasing number of mobile e applications might cause some trouble to find specific applications promptly. For this reason, this research study presents a novel methodology to predict what is the next mobile app that a user is going to open based on supervised machine learning algorithms. This approach might lead to improving the user experience and thereby make the smartphone system more efficient and user-friendly. Based on these findings, this approach can provide a robust approach. The proposed methodology utilizes a deep learning algorithm (long short-term memory) to accurately predict the probability of a given application to be used by the smartphone user after a sequence of applications usage. The experimental result shows that the forecasting of those applications' usage performance can be correct with an achieved accuracy of approximately 80%. However, this study does not cover privacy, security threats that the proposed model could expose the user to such as those use cases of Smartphones of user data theft and Smartphone based frauds (Srivastava, et el, 2019). Future work will include a collection of a bigger dataset from different types of users. This could allow more understanding of users' behavioral and interactions. Also, more recent and advanced learning algorithms will be evaluated such as attention models which have been proved in other research to achieve high performance in predicting time series-based data.

REFERENCES

[1] A. A. Alzubaidi, "Continuous authentication of smartphone owners based on app access behavior," Ph.D. dissertation, Dept. Comput. Sci., Univ. Colorado Colorado Springs, Springs, CO, USA, 2018.

[2] A. Alzubaidi and J. Kalita, "Authentication of smartphone users using behavioral biometrics," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 3, pp. 1998–2026, 3rd Quart., 2016.

[3] *Android Debug Bridge*. (2018). Accessed: Jan. 25, 2018. [Online]. Available: https://developer.android.com/studio/commandline/adb.html#howadbworks

[4] R. Baeza-Yates, D. Jiang, F. Silvestri, and B. Harrison, "Predicting the next app that you are going to use," in *Proc. 8th ACM Int. Conf. Web Search Data Mining*, Feb. 2015, pp. 285–294.

[5] H. Cao and M. Lin, "Mining smartphone data for app usage prediction and recommendations: A survey," *Pervasive Mobile Comput.*, vol. 37, pp. 1–22, Jun. 2017.

[6] *Christopher Olah*. (2015). *Understanding LSTM Networks*. Accessed: Mar. 31, 2019. [Online]. Available: http://colah.github.io/posts/2015-08-Understanding-LSTMs/

[7] N. Clarke, *Transparent User Authentication: Biometrics, RFID and Behavioural Profiling*. Cham, Switzerland: Springer, 2011.

[8] (2019). *Colab*. Accessed: Jan. 7, 2019. [Online]. Available: https://colab.research.google.com/

[9] M. D. Marsico, M. Nappi, D. Riccio, and H. Wechsler, "Mobile iris challenge evaluation (MICHE)-I, biometric iris dataset and protocols," *Pattern Recognit. Lett.*, vol. 57, pp. 17–23, May 2015.

[10] (2018). *Deloitte Mobile Consumer Survey*. Accessed: Mar. 15, 2019. [Online]. Available: https://www2.deloitte.com/uk/en/pages/technology-media-and-telecommunications/articles/mobile-consumer-survey.html

[11] C. Fang, Y. Wang, D. Mu, and Z. Wu, "Next-app prediction by fusing semantic information with sequential behavior," *IEEE Access*, vol. 6, pp. 73489–73498, 2018.

[12] D. Han, J. Li, L. Yang, and Z. Zeng, "A recommender system to address the cold start problem for app usage prediction," *Int. J. Mach. Learn. Cybern.*, vol. 10, no. 9, pp. 2257–2268, 2018.

[13] K. Huang, C. Zhang, X. Ma, and G. Chen, "Predicting mobile application usage using contextual information," in *Proc. ACM Conf. Ubiquitous Comput. (UbiComp)*, 2012, pp. 1059–1065.

[14] K. Janocha and W. M. Czarnecki, "On loss functions for deep neural networks in classification," 2017, *arXiv:1702.05659*.

[15] J. Brownlee. (2019). *How to Improve Neural Network Stability and Modelling Performance With Data Scaling*. Accessed: Mar. 29, 2019. [Online]. Available: https://machinelearningmastery.com/how-to-improve-neural-network-stability-and-modeling-performance-with-data-scaling/

[16] U. Mahbub, J. Komulainen, D. Ferreira, and R. Chellappa, "Continuous authentication of smartphones based on application usage," 2018, *arXiv:1808.03319*.

[17] W. Meng, D. S. Wong, S. Furnell, and J. Zhou, "Surveying the development of biometric user authentication on mobile phones," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 3, pp. 1268–1293, 3rd Quart., 2015.

[18] A. Parate, M. Böhmer, D. Chu, D. Ganesan, and B. M. Marlin, "Practical prediction and prefetch for faster access to applications on mobile phones," in *Proc. ACM Int. Joint Conf. Pervasive Ubiquitous Comput.*, Sep. 2013, pp. 275–284.

[19] E. P. Torres, "Authentication of legitimate users of smartphones based on app usage sequences," Doctoral dissertation, Univ. Colorado Colorado Springs. Kraemer Family Library, Springs, CO, USA, 2018, pp. 1–54.

[20] R. Inostroza, C. Rusu, S. Roncagliolo, V. Rusu, and C. A. Collazos, "Developing SMASH: A set of SMArtphone's uSability heuristics," *Comput. Standards Interfaces*, vol. 43, pp. 40–52, Jan. 2016.

[21] C. Shin, J.-H. Hong, and A. K. Dey, "Understanding and prediction of mobile application usage for smart phones," in *Proc. ACM Conf. Ubiquitous Comput. (UbiComp)*, 2012, pp. 173–182.

[22] V. Srinivasan, S. Moghaddam, A. Mukherji, K. K. Rachuri, C. Xu, and E. M. Tapia, "MobileMiner: Mining your frequent patterns on your phone," in *Proc. ACM Int. Joint Conf. Pervasive Ubiquitous Comput.*, Sep. 2014, pp. 389–400.

[23] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, Jan. 2014.

[24] S. R. Srivastava, S. Dube, G. Shrivastaya, and K. Sharma, "Smartphone triggered security challenges-issues, case studies and prevention," in *Cyber Security in Parallel and Distributed Computing: Concepts, Techniques, Applications and Case Studies*, 2019, pp. 187–206.

[25] N. S. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, 2019. Accessed: Mar. 15, 2019. [Online]. Available: https://www.statista.com/statistics/269025/worldwide-mobile-app-revenue-forecast/

[26] Statista. (2019). *Annual Number of Global Mobile App Downloads 2017–2022*. Accessed: Mar. 15, 2019. [Online]. Available: https://www.statista.com/statistics/271644/worldwide-free-and-paid-mobile-app-store-downloads/

[27] (2019). *TensorFlow*. Accessed: Aug. 31, 2019. [Online]. Available: https://www.tensorflow.org/community/roadmap

[28] S. Xu, W. Li, X. Zhang, S. Gao, T. Zhan, Y. Zhao, and T. Sun, "Predicting smartphone app usage with recurrent neural networks," in *Proc. Int. Conf. Wireless Algorithms, Syst., Appl.*, Cham, Switzerland: Springer, Jun. 2018, pp. 532–544.

[29] C. Yu, Y. Liu, D. Yao, L. T. Yang, H. Jin, H. Chen, and Q. Ding, "Modeling user activity patterns for next-place prediction," *IEEE Syst. J.*, vol. 11, no. 2, pp. 1060–1071, Jun. 2017.

[30] M. Zeng and N. Xiao, "Effective combination of DenseNet and BiLSTM for keyword spotting," *IEEE Access*, vol. 7, pp. 10767–10775, 2019.

[31] C. Zhang, O. Vinyals, R. Munos, and S. Bengio, "A study on overfitting in deep reinforcement learning," 2018, *arXiv:1804.06893*.

[32] J. Zhao, Y. Li, J. Wu, and Q. Liao, "Predict what app to use next time only consider time and latest used app context," in *Proc. 2nd Int. Conf. Manage. Eng., Softw. Eng. Service Sci. (ICMSS)*, 2018, pp. 163–166.

[33] S. Zhao, Z. Luo, Z. Jiang, H. Wang, F. Xu, S. Li, J. Yin, and G. Pan, "AppUsage2 Vec: Modeling smartphone app usage for prediction," in *Proc. IEEE 35th Int. Conf. Data Eng. (ICDE)*, Apr. 2019, pp. 167–170.

[34] X. Zou, W. Zhang, S. Li, and G. Pan, "Prophet: What app you wish to use next," in *Proc. ACM Conf. Pervasive Ubiquitous Comput. Adjunct Publication*, 2013, pp. 167–170.

ABDULRAHMAN ALRUBAN received the bachelor's degree in information technology from Solent University, Southampton, U.K., the master's degree in computer systems security from the University of Glamorgan, U.K., and the Ph.D. degree in computer science from the University of Plymouth, U.K.

He has published more than 15 scientific papers as a senior researcher and participant in several scientific conferences and magazines in the field of artificial intelligence, digital forensics, and cybersecurity. As an inventor, he invented a digital forensics system which patented in British and American Patent Offices, in 2017. He received several honors and awards, such as the Golden Medal of the 12th International Invention Fair in the Middle East, in 2020; the Second Place of the GCC Patent Office Award, in 2020; and the First Place Award of the Ministry of Economy, United Arab Emirates Hackathon, in 2020. As a recognition, he was chosen in 2018 to represent outstanding Saudi students in the U.K., and delivered a speech on their behalf at the Saudi Embassy in London. He was also recognition of his scientific distinction, he received the Scientific Creativity Award from Majmaah University, in 2018.