# ARM: ANN-based Ranking Model for Privacy and Security Analysis in Smartphone Ecosystems

Majid Hatamian*, Jetzabel Serna*
*Chair of Mobile Business & Multilateral Security
Goethe University Frankfurt
Frankfurt am Main, Germany
E-mail: majid.hatamian.h@ieee.org, jetzabel.serna@m-chair.de

***Abstract*—Smartphone ecosystems are considered as a unique source due to the large number of apps which in turn makes an extensive use of personal data. Currently, there is no privacy and security preservation mechanism in smartphone ecosystems to enable users to compare apps in terms of privacy and security protection level, and to alarm them regarding the invasive issues (in terms of privacy and security) of apps before installing them. In this paper, we exploit user comments on app stores as an important source to extract privacy and security invasive (PSI) claims corresponding to apps. Thus, we propose an artificial neural network (ANN)-based ranking model (ARM) in order to classify user comments with privacy and security concerns. Our ranking model is based on three main features namely privacy and security, sentiment, and lifetime analyses as the input of the ranking model along with a novel mathematical formulation in such a way as to maximise the differentiation between comments. The performance results show that ARM is able to classify and predict PSI user comments with accuracy as high as 93.3%. Our findings confirm that due to the functionality of ARM, it has the potential to be widely adopted in smartphone ecosystems.**

***Index Terms*—artificial neural networks; smartphone apps; privacy; security; sentiment.**

## I. Introduction

With the rapid growth of technology in recent years, our life is now significantly surrounded by or even dependent on the use of technological devices, especially smartphones [1]. Accordingly, security and privacy have become a serious concern in the field of information technology in diverse applications such as computer networks, wireless communications, etc [2]–[4]. This is even more serious when it comes to smartphone apps since they provide context-sensitive services to the users. As a result and not surprisingly, the number of mobile apps available on app stores has exploded over the past few years [5]. For instance, the number of available apps in the Google Play Store surpassed 1 million apps in July 2013 and was most recently placed at 2.8 million apps in March 2017 [6]. In addition, app stores allow users to rate and write reviews about the apps they have downloaded. These user comments mostly contain useful information such as claims of privacy and security invasive activities, bugs, suggestions for adding new features, etc. As a result, many apps receive thousands of reviews each day which can be considered as an indirect source that reflect users' experience of and expectation for the apps.

**Motivation** Currently there is no metric on app stores in order to inform users about the privacy and security invasiveness level of apps. As shown in [7], if we would be able to communicate the potential risks of using certain apps to the users, then the users would be most likely to choose the apps with better privacy and security protection, i.e. which apps have been rated negatively regarding the privacy and security issues that they have.

**Our Contribution** In this paper, we introduce an ANN-based ranking model (ARM) as a method for extracting knowledge regarding the privacy and security issues of apps from their comments on app stores. ARM benefits from a mathematical formulation for PSI, sentiment, and lifetime analyses along with a supervised machine learning algorithm which is aimed to classify user comments into different classes. Our research findings clearly reveal that ARM is capable of widely adopting due to its remarkable applicability through an extensive performance evaluation to better preserve the users' privacy.

The rest of this paper is organised as follows. Section II reviews the current approaches for learning from crowds and classification of user comments. In Section III, we explain and elaborate the different parts of ARM. This section also covers our proposed mathematical formulation and the neural network structure. Section IV elucidates the performance evaluation and clarifies to which extent our proposed classification algorithm is efficient. Finally, we discuss the future work and conclude the paper in Section V.

## II. Related Work

So far, various methods for classification of user comments have been proposed. In [8], the authors used a supervised multi-label learning method to identify different types of user comments with security and privacy issues. In [9], a method has been proposed to investigate the most informative user reviews from a large and rapidly increasing pool of user reviews. The authors used a review ranking scheme to prioritise the informative user reviews. Furthermore, a filtering process is utilised to filter out non-informative comments. Although our method performs the same, but we do not focus on the quality of the information, but whether the user comments are PSI or not. In [10], the authors proposed a fully unsupervised algorithm for selecting the most helpful book reviews. Although this approach does not focus on privacy and security sensitiveness of comments, but it can efficiently classify the user comments. It uses a score-based approach to identify the optimal reviews. Reviews are then converted to this score-based representation and the ranking phase is started. However, this approach uses an unsupervised machine learning method, without the possibility to tell the algorithm what to do. As a consequence, it is difficult to judge the quality of clustering results in a definitive way, as opposed to our work which uses a supervised learning approach. In [11], a theoretical analysis of crowdsourced content curation has been proposed. The authors studied crowd-curation mechanisms that rank articles according to a score which is a function of user comments. Although their theoretical approach is not especially investigated for smartphone ecosystems, but it is able to quantify the dynamics of which articles become popular regarding the scores obtained from user comments. We also did a mathematical analysis in our work which enables us to score the user comments according to a keyword-based searching approach. In [12], an in-depth analysis of commenting and comment rating behavior in social web has been proposed. In this work, the authors examined the dependencies of comment ratings with textual content (e.g. videos and their meta data) to collect a comprehensive understanding of the community commenting behavior. They also exploited the applicability of machine learning and data mining to identify the acceptance of comments by the community. Authors in [13] proposed a crowdsourcing ranking method for user comments in smartphone ecosystems. The authors suggested to use risk assessment of an app from its user comments as a crowdsourcing problem in order to provide a ranking model. They used a security labeling system from user comments to automatically rank the risks of app based on these learned labels as features. On the contrary, in addition to PSI analysis, our approach benefits from a sentiment and lifetime analyses of user comments.

## III. ANN-based Ranking Model for Privacy and Security Assessment

ANNs with Back Propagation (BP) learning algorithm are widely adopted in solving various classification problems. The main advantage of a BP algorithm is that the convergence to an optimal solution is guaranteed. Additionally, ANN is a nonlinear and non-paramateric model that is easy to use and understand compared to statistical methods [14]. Initially, we consider a set of $m$ user comments $\mathcal{C} = \{c_1, c_2, \ldots, c_m\}$ consisting of the training set $t_1$ which includes $m$ user comments with their respective privacy and security classes (see Section IV) $R = \{r_1, r_2, \ldots, r_m\}$. Our goal is to measure the privacy and security ranks $[r_{m+1}, r_{m+2}, \ldots, r_{m+n}\}$ associated with each comment in testing set $t_2$, where $n \leq m$. Moreover, each comment $c_i$ has a set of features $F = \{f_1, f_2, \ldots, f_s\}$. We determine each feature as an informative element regarding each comment (see Section III-A). As a result, the set of features related to each comment is defined as follows:

$$\{(F_{c_1}, r_1), \ldots, (F_{c_m}, r_m), (F_{c_{m+1}}, r_{m+1}), \ldots, (F_{c_{m+n}}, r_{m+n})\}, \quad (1)$$

where $(F_{c_1}, r_1)$ represents the feature $F_{c_1}$ associated to the comment $c_1$ and rank $r_1$.

### A. Inputs

*1) Privacy and Security Invasive (PSI) Score:* We define PSI score as the first input of the neural network. The main purpose in measuring the PSI score is to investigate to which level a user comment includes privacy and security sensitive terms. For this reason, we define a privacy catalogue. To derive this catalogue, we combined the insights from the preceding literature review and our own search using WordNet [15] to enhance the listing with e.g. synonyms or new buzzwords. In our algorithm, we assume that a user comment comprises keywords which are happening more than once, are more informative.

Let $\mathcal{PK} = \{pk_1, pk_2, \ldots, pk_t\}$ be the set of keywords which are sensitive to the privacy and security concerns. Consider set $\mathcal{Q} = \{(q_{pk_1}, c_1), (q_{pk_2}, c_2), \ldots, (q_{pk_t}, c_m)\}$ consisting of pairs in which $(q_{pk_1}, c_1)$ is represented as the number of repetitions of keyword $pk_1$ associated with user comment $c_1$. Accordingly, we define PSI score ($PSIS$) as follows:

$$PSIS = \frac{\sum_{j=1}^{t} \text{index}_{pk_j}}{\sum_{j=1}^{t} q_{pk_j}(\geq 2)} + \frac{1}{\sum_{j=1}^{t} q_{pk_j}(=1)}, \quad (2)$$

where $\text{index}_{pk_j}$ indicates the index related to each keyword $pk_j$. The index of each keyword will be obtained as follows:

$$\text{index}_{pk_j} = \begin{cases} 1 & \text{if} \quad q_{pk_j} \neq 0 \\ 0 & \text{otherwise.} \end{cases} \quad (3)$$

According to (3), if keyword $pk_j$ happens in a given user comment ($q_{pk_j} \neq 0$), then its index is 1 ($\text{index}_{pk_j}$=1). This is due to the fact that, we aim to investigate to which extent the number of repetitions of each keyword can influence the overall rank which will be assigned to each user comment. The denominator of the first element of (2) indicates the total number of repetitions of keywords which occurred at least two times. It means in the first element, we highlighted the importance of occurrence of keywords with two ore more repetitions in a given user comment. The more it approaches to zero, the more it highlights the importance of keyword repetitions with two or more than two times happenings. Furthermore, the second element of (2) is aimed to differentiate the occurrence of keywords with one time repetition from the keywords with two or more than two times repetitions (first element). This is why (2) considers both kinds of repetitions (first element for keywords with two or more repetitions, and second element for keywords with one repetition). This will further influence the way by which ARM classifies user comments since we assume there should be a mechanism to differentiate user comments with more repetitive privacy and security sensitive keywords.

*2) Sentiment Score:* We introduce sentiment score ($SS$) as the second input of the neural network which is calculated in the similar mathematical structure as $PSIS$. Having considering this fact, we analyse the impact of sentiment on the overall ranking list for user comments. For example, consider user comments $C_1$:*"This app starts sending you spam notifications."* and $C_2$:*"This app starts sending you* ***lots of*** *spam notifications.* ***I hate it!****"*. Although both comments claim the same privacy and security concern, but the overall impression from these two comments is relatively different. Comment $C_1$ generally claims spamming issues, however, comment $C_2$ strongly emphasises that this app notifies *'lots of'* spams. Additionally, it also asserts the overall user's impression about the app (*'I hate it!'*). We assume that our algorithm should be able to differentiate these two comments.

*3) Lifetime Score:* To evaluate the reliability of user comments, we also consider the lifetime score ($LS$) of each user comment. Therefore, the third input of the neural network is the date on which the comment was published. The reason lies in the fact that, recent comments are more informative. We assume that users who install new apps tend to read new comments because: 1) it is easy to reach them since they are in the first page 2) they are more informative and reliable since they have been published recently and contain new claims. As a particular case, consider an app which had inappropriate privacy and security protection in past. If we do not consider the app's lifetime while running our classification algorithm, we wrongly neglect this fact that maybe the developers of this app have ameliorated the privacy and security protection and resolved the existing issues. Similarly, an app which had very good reputation in past (in terms of privacy and security protection), might have recently fallen through privacy and security issues. Thus, lifetime plays an increasingly important role in the classification procedure. For the sake of convenience, we converted all the dates into days.

## B. Network Structure

The proposed structure of the neural network for ARM is shown by Figure 1.

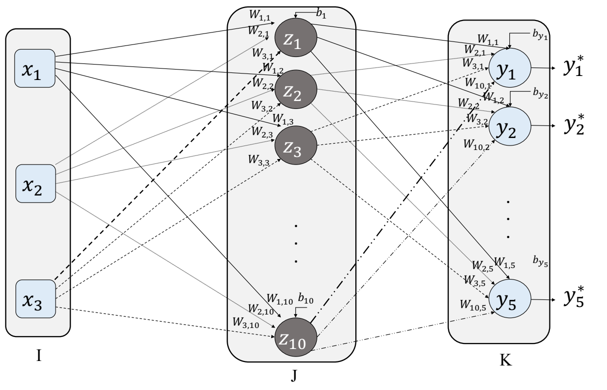


Figure 1: The main structure of the proposed model for ANN.

It consists of one hidden layer ($J$) with ten neurons to maintain the computational complexity on a reasonable level [16]. Layers $I$ and $K$ indicate the input and output layers. We show $W_{i,j}$ and $W_{j,k}$ as the weights from node $x_i$ of layer $I$ to node $z_j$ of layer $J$, and from node $z_j$ of layer $J$ to node $y_k$ of layer $K$, respectively. Every node in hidden layer has the bias value $b_j$ which allows us to shift the transfer function curve horizontally along the input axis. This will allow the network to produce arbitrary outputs different from the defaults. Since the modification of neuron's weights alone enables us to change the shape of our transfer function curve, and not its equilibrium/zero crossing point.

Given a set of training data set, $y_k$ is a function of all the nodes in layer $K$ ($y_k = \rho(k)$), and $y_k^*$ is the target output. Since we are using a backpropagation model, and we aim to solve a classification problem (and we are not dealing with a neural network that does regression, where the value to be predicted is numeric, or a time series neural network, or any other types of neural networks), we should minimise the error through iterative updates of weights for all training samples (user comments). In practice, this approach guarantees the network to have a good performance, and it enables us to evaluate the quality of the neural network. Therefore, we use the following cross entropy error function [17] which should be minimised:

$$E = -\frac{1}{N}\sum_{k\in K}[y_k^* \ln y_k + (1-y_k^*)\ln(1-y_k)], \tag{4}$$

where $k$ is the indice of $k$th node in layer $K$ (output), $N$ is the total number of user comments in training set, and $y_k^*$ is the corresponding desired output, respectively. We let the error of the network for a single training user comment be denoted by $E$. We want to calculate $\frac{\delta E}{\delta W_{j,k}^l}$, the rate of change of the error with respect to the given connective weight, so we can minimise it. Now we have:

$$\frac{\delta E}{\delta W_{j,k}} = -\frac{\delta}{\delta W_{j,k}}\frac{1}{N}\sum_{k\in K}[y_k^* \ln y_k + (1-y_k^*)\ln(1-y_k)]. \tag{5}$$

Having considering this fact, the input layer comprises three nodes $x_1$, $x_2$, and $x_3$ which are equivalent to our predefined inputs (Section III-A), including PSI score, sentiment score, and lifetime score. At the hidden nodes, denoted $z_1$ to $z_{10}$, values are computed according to:

$$z_j = f(b_j + \sum_{i=1}^{3}\sum_{j=1}^{10} W_{i,j}.x_i), \tag{6}$$

where $f$ denotes a continuous transfer function (usually a sigmoid-type function such as $\rho(x) = 1/(1+e^{-x})$). These values are in turn fed forward to the output node $y_k$, where the desired network output, $y_k^*$, is computed according to

$$y_k^* = g(b_{y_k} + \sum_{j=1}^{10}\sum_{k=1}^{5} W_{j,k}.z_j), \tag{7}$$

where $W_{j,k}$ denotes the weight assigned to the hidden node output $y_k$, $b_{y_k}$ denotes the bias at the output node, and $g$ denotes a linear transfer function.

As already discussed, $y_k = \rho(k)$, so we substitute $y_k = \rho(k)$ into (5), and apply the chain rule twice, obtaining:

$$\begin{aligned}\frac{\delta E}{\delta W_{j,k}} &= -\frac{1}{N}\sum_{k\in K}\Big(\frac{y_k^*}{\rho(k)} - \frac{(1-y_k^*)}{1-\rho(k)}\Big)\frac{\delta\rho}{\delta W_{j,k}} \\ &= -\frac{1}{N}\sum_{k\in K}\Big(\frac{y_k^*}{\rho(k)} - \frac{(1-y_k^*)}{1-\rho(k)}\Big)\rho'(k)x_j \qquad (8) \\ &= \frac{1}{N}\sum_{k\in K}\frac{\rho'(k)x_j}{\rho(k)(1-\rho(k))}(\rho(k)-y_k^*).\end{aligned}$$

As previously mentioned, $k$ is a function of $\rho(x) = 1/(1+e^{-x})$. As a result, we can conclude that $\rho'(k) = \rho(k)(1-\rho(k))$. So we can rewrite (8) as follows:

$$\frac{\delta E}{\delta W_{j,k}} = \frac{1}{N}\sum_{k\in K} x_j(\rho(k)-y_k^*). \qquad (9)$$

Equation (9) tells us that the rate at which the weight learns is controlled by $\rho(k) - y_k^*$ (by the error in the output). In other words, as long as the value of error is too high, the neurons will learn faster. Algorithm 1 shows the whole learning procedure of ARM.

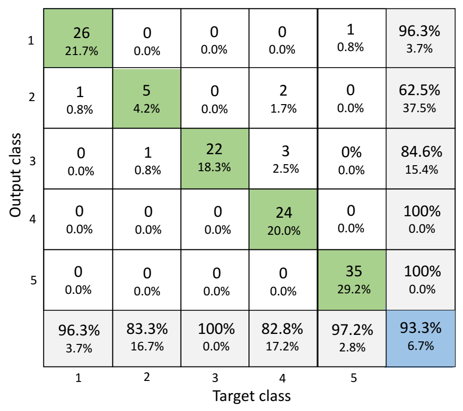
**Algorithm 1** Pseudo-code of all steps in ARM.

```
1.  Procedure ARM
2.  START
3.      Compute $PSIS$ AND $SS$= $\frac{\sum_{j=1}^{t}\text{index}_{pk_j}}{\sum_{j=1}^{t} q_{pk_j}(\geq 2)} + \frac{1}{\sum_{j=1}^{t} q_{pk_j}(=1)}$
4.      Get $LS$
5.      START Learning
6.          for each training sample
7.              Compute $z_j$, $y_k^*$, $E$
10.                 Backpropagate the error
11.                 Compute $\frac{\delta E}{\delta W_{j,k}}$
12.         END for
13.     END Learning
14. END Procedure
```

## IV. Performance Evaluation

For the evaluation of our algorithm, we used the data set in [8]. This data set has been collected from Google Play Store since the user comments are publicly available. A crawler has been written in python for crawling the user comments. It is worth mentioning that the data set was annotated manually by mobile app professionals. The proposed algorithm has been evaluated by MATLAB R2015B [18]. We use early stopping method to estimate the best situation in which the proposed algorithm is adequately trained. For this reason, we divide the whole data in three data sets, including training, validation and testing sets. In our scenario, we used 120 comments for training, 30 comments for validation (we exploit validation set to estimate how well our classifier has been trained), and 90 comments for testing. The training process uses training data set and must be executed epoch by epoch (the maximum number of epochs is set to 1000 - we also define an epoch as a measure for the number of times that all of the training samples are used to update the weights.) in order to calculate the error of the network in each epoch for the validation set. The network for the epoch with the minimum validation error is selected for the testing process.

As it can be seen in Figure 2, we used confusion matrix to demonstrate the performance of ARM. Basically, confusion matrix contains information about actual and predicted classifications. In Figure 2, the horizontal axis shows the target class and the vertical axis shows the output class. We have defined five classes. User comments with the most privacy and security invasive issues are classified in class 5. Similarly, the comments with the least privacy and security invasive issues are classified in class 1. The results show that the proposed classifier is able to classify the user comments with the accuracy of 96.3%, 83.3%, 100%, 82.8%, and 97.2% into classes 1, 2, 3, 4, and 5, respectively. Accordingly, the overall accuracy of the proposed classifier is as high as 93.3%.



| Output class \ Target class | 1 | 2 | 3 | 4 | 5 | |
|---|---|---|---|---|---|---|
| 1 | 26 21.7% | 0 0.0% | 0 0.0% | 0 0.0% | 1 0.8% | 96.3% 3.7% |
| 2 | 1 0.8% | 5 4.2% | 0 0.0% | 2 1.7% | 0 0.0% | 62.5% 37.5% |
| 3 | 0 0.0% | 1 0.8% | 22 18.3% | 3 2.5% | 0% 0.0% | 84.6% 15.4% |
| 4 | 0 0.0% | 0 0.0% | 0 0.0% | 24 20.0% | 0 0.0% | 100% 0.0% |
| 5 | 0 0.0% | 0 0.0% | 0 0.0% | 0 0.0% | 35 29.2% | 100% 0.0% |
| | 96.3% 3.7% | 83.3% 16.7% | 100% 0.0% | 82.8% 17.2% | 97.2% 2.8% | 93.3% 6.7% |

Figure 2: Confusion matrix for performance evaluation of ARM.

Figure 3 shows the receiver operating characteristic (ROC) curve. We used two metrics to evaluate the functionality of the proposed classifier, including sensitivity (true positive rate) and specificity (true negative rate). Sensitivity is defined as a metric which indicates the ratio of positives which are correctly labeled and identified as positive. On the other hand, specificity represents the ratio of negatives which are correctly labeled and identified as negative. If we show the values of true positives, false positives, true negatives, and false negatives by $TP$ , $FP$ , $TN$ and $FN$, then we have:

$$\text{Sensitivity} = \frac{TP}{TP+FN}. \qquad (10)$$

$$\text{Specificity} = \frac{TN}{TN+FP}, \qquad (11)$$

where $TP$ , $FP$ , $TN$ and $FN$ show ARM correctly classifies the user comment as PSI if the comment is, ARM incorrectly classifies the user comment as PSI if the comment is not, ARM correctly classifies the user comment as non-PSI if the comment is not, and ARM incorrectly classifies the user comment as non-PSI if the comment is, respectively. As it can be seen in Figure 3, Y-axis shows the true positive rate and X-axis shows the false positive rate. This curve shows a detailed overview regarding the trade-off between sensitivity and specificity. The most important notion concerning ROC curve is that the point $(0, 1)$ represents perfect classification, where the true positive rate is at its maximum level, and false positive rate is at its minimum value. In fact, the more the points are accumulated closely near this point, the more the classifier is accurate.

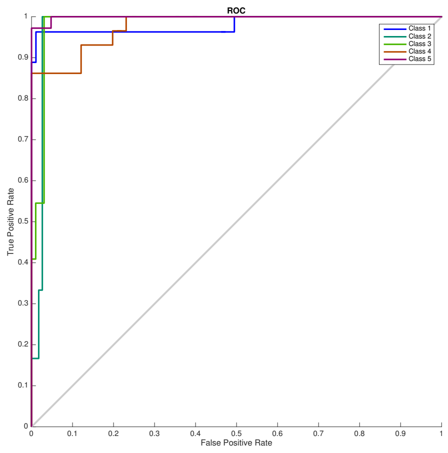


Figure 3: True positive and false positive ratios of ARM in form of ROC.

Figure 4 shows the network performance with regards to the cross entropy error function. As it can be clearly seen, the error reduces after more epochs of training, but might start to increase on the validation data set as the network starts overfitting the training data. With respect to the default settings, the training stops after six consecutive increments in validation error, and the best performance is taken from the epoch with the lowest validation error. Figure 4 shows that the best validation performance is 0.022756 at epoch 19. This figure does not show any major problem with the training. It is worth to mention that the distributions and variances of the validation and test curves are very similar which confirms that the functionality of ARM during the validation and test phases is almost the same that is always desirable in every classification problem.

Table I shows some examples regarding the strength of ARM in distinguishing different types of comments with different sentiments and PSI concerns (due to space limitation, we could not provide a long list of comments). Comment #1 is the basic comment. Based on it, there are more complicated comments with more descriptive sentiment and PSI terms. Both comments #1 and #2 are classified in Class 1 (the class with the least privacy and security concerns). This is due to the fact that, both comments report the same fact, although comment #2 comprises a sentimental term ("*annoying*"). Importantly, comments #3 and #4 are categorised in

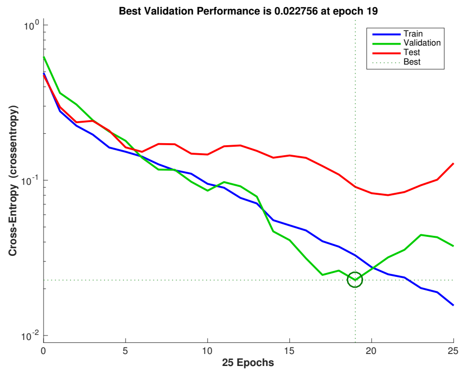


Figure 4: ANN performance regarding the cross entropy error function.

Class 2 due to existing of more sentimental and PSI keywords. The most interesting thing about this table is indicated in comments #5, #6, and #7, where ARM was able to properly recognise the discrepancy among these three comments in terms of sentiments and PSI concerns. This is one of the important features of ARM which reveals the importance of using both sentiment and PSI analyses together by exploiting a machine learning technique in order to better discriminate user comments.

Table I: Discrimination of user comments by ARM

| # | Sample comment | Class |
|---|---|---|
| 1 | *I'd privacy issues with this app* | 1 |
| 2 | *I'd annoying privacy issues with this app* | 1 |
| 3 | *I'd the worst annoying privacy issues with this suspicious app* | 2 |
| 4 | *I'd the worst annoying privacy issues with this suspicious app. Don't trust it!* | 2 |
| 5 | *I'd the worst annoying privacy protection issues with this suspicious app. Don't trust it! it sends you many unwanted ads!* | 3 |
| 6 | *I'd the worst annoying privacy protection issues with this spammy and suspicious app. Don't trust it! it sends you many unwanted ads, it also shares your personal information with other parties.* | 4 |
| 7 | *I'd the worst annoying privacy protection issues with this spammy and suspicious app. Don't trust it! it sends you many unwanted ads, it also shares your personal information with other parties. I don't know why my profile photo is shared with other apps even I haven't been informed, they don't respect our privacy!* | 5 |

Table II shows the comparison between different versions of ARM and the proposed method in [8] called CDCE. First, we evaluated the accuracy of ARM while neglecting the sentiment analysis. This means we only considered PSI analysis of user comments. ARM-P indicates the method using only PSI analysis expansion without any extra-feature. Second, we measured the accuracy of ARM while considering both PSI and lifetime analyses (showed by ARM-PL in Table II). Finally, we examined the accuracy of ARM while considering PSI, sentiment and lifetime analyses which is shown by ARM in Table II. By comparing ARM-P and ARM-PL, the general improvement from using both lifetime and PSI analyses is obvious. This improvement makes our results much more "smoother" among similar comments in feature level, and improves the performance of the ranking model. Similarly, the difference between ARM-PL and ARM shows that using all the three features together guarantees a more precise classification. This is especially tangible when we compare ARM with CDCE that shows the best performance among others.

Table II: Comparative results on accuracy

| Method | Accuracy |
|---|---|
| ARM-P | 70.1% |
| ARM-PL | 84.6% |
| **ARM** | 93.3% |
| CDCE | 72.6% |

### A. *Discussion*

Our findings confirm a better performance when we use sentiment and lifetime analyses. This leads to the conclusion that sentiment and lifetime of user comments should be considered as the important elements of the classification task. Moreover, the scope of this paper comprises any smartphone ecosystem. It is important to note that one critical limitation against our work is the limited samples that we used for training and testing phases. Furthermore, using not annotated user comments, could have increased the scope and depth of analyses.

## V. Conclusions and Future Work

In this paper, we introduced ARM as an ANN-based ranking model for privacy and security assessment in smartphone ecosystems. We proposed a mathematical analysis to investigate the importance of PSI score of user comments, as well as sentiment analysis based on keyword searching. We further identified the lifetime of user comments as an important metric while assigning rank to the comments. Additionally, the evaluation results revealed that ARM is capable of predicting and classifying the user comments during the testing phase with the overall accuracy of 93.3%.

The mathematical analysis in this work can serve as the basis for a number of interesting future research directions. We previously showed that an appropriate mathematical analysis of user comments can be exploited as different features to train and test a machine learning algorithm which in turn positively influences the classifier's performance. In addition, we believe using a multi-labeling learning can potentially lead to a more efficient privacy and security quantification, meaning that one comment could be assigned to more than one privacy and security class.

## VI. Acknowledgement

This research work has received funding from the H2020 Marie Skłodowska-Curie EU project "Privacy&Us" under the grant agreement No 675730.

## References

[1] M. Hatamian and J. Serna, "Beacon alarming: Informed decision-making supporter and privacy risk analyser in smartphone applications," in *Proceedings of the* 35th *IEEE International Conference on Consumer Electronics (ICCE)*, Las Vegas, USA, 468–471, 2017.

[2] A. Naghizadeh, B. Razeghi, E. Meamari, M. Hatamian and R. E. Atani, "C-trust: A trust management system to improve fairness on circular P2P networks," Peer-to-Peer Networking and Applications, vol. 9, no. 6, 1128–1144, 2016.

[3] A. Naghizadeh, B. Razeghi, I. Radmanesh, M. Hatamian, R. E. Atani and Z. N. Norudi, "Counter attack to free-riders: Filling a security hole in BitTorrent protocol," To appear in *Proceedings of the* 12th *International Conference on Networking, Sensing and Control (ICNSC 2015)*, Taipei, Taiwan, 128–133, 2015.

[4] S. Berenjian, M. Shajari, N. Farshid, and M. Hatamian, "Intelligent automated intrusion response system based on fuzzy decision making and risk assessment," in *Proceedings of the* 8th *IEEE International Conference on Intelligent Systems (IS)*, Sofia, Bulgaria, 709–714, 2016.

[5] M. Hatamian, J. Serna, K. Rannenberg and B. Igler, "FAIR: Fuzzy alarming index rule for privacy analysis in smartphone apps," in *Proceedings of the* 14th *International Conference on Trust, Privacy & Security in Digital Business (TrustBus 2017)*, Lyon, France, 3–18, 2017.

[6] "Number of apps available in leading app stores as of March 2017," accessed June 11, 2017, https://www.statista.com/statistics/276623/number-of-apps-available-in-leading-app-stores/

[7] G. Bal and K. Rannenberg, "User control mechanisms for privacy protection should go hand in hand with privacy-consequence information: The case of smartphone apps," in *Proceedings of W3C Workshop on Privacy and User-Centric Controls*, Berlin, Germany, pp. 1–5, 2014.

[8] L. Cen, L. Si, N. Li and H. Jin, "User comment analysis for android apps and CSPI detection with comment expansion," in *Proceedings of the* 1st *International Workshop on Privacy-Preserving IR (PIR)*, Gold Coast, Australia, pp. 25–30, 2014.

[9] N. Chen, J. Lin, S. C. H. Hoi, X. X. Nanyang and B. Z. Nanyang, "Ar-miner: Mining informative reviews for developers from mobile app marketplace," in *Proceedings of the* 36th *International Conference on Software Engineering*, Hyderabad, India, pp. 767–778, 2014.

[10] O. Tsur and A. Rappoport, "A fully unsupervised algorithm for selecting the most helpful book reviews," in *Proceedings of the* 3rd *International AAAI Conference on Weblogs and Social Media*, California, USA, pp. 154–161, 2014.

[11] G. Askalidis and G. Stoddard, "A theoretical analysis of crowdsourced content curation," in *Proceedings of the* 3rd *Workshop on Social Computing and User Generated Content*, Pennsylvania, USA, 2013.

[12] S. Siersdorfer, S. Chelaru, J. S. Pedro, I. S. Altingovde and W. Nejdl, "Analyzing and mining comments and comment ratings on the social web," Journal ACM Transactions on the Web, vol. 8, no. 3, Article no. 17, 2014.

[13] L. Cen, D. Kong, H. Jin and L. Si, "Mobile app security risk assessment: A crowdsourcing ranking approach from user comments," in *Proceedings of SIAM International Conference on Data Mining*, Vancouver, Canada, pp. 658–666, 2015.

[14] R. Rojas, "Neural networks," *Springer-Verlag, Berlin*, 1996.

[15] Princeton University, "WordNet,", http://wordnetweb.princeton.edu/perl/webwn," 2010.

[16] D. Stathakis, "How many hidden layers and nodes?," International Journal of Remote Sensing, vol. 30, no. 8, pp. 2133–2147, 2009.

[17] P. Boer, D. P. Kroese, S. Mannor and R. Y. Rubinstein, "A tutorial on the cross-entropy method," Annals of Operations Research, vol. 134, no. 1, pp. 19–67, 2005.

[18] MATLAB R2015B, The MathWorks, Inc., Natick, Massachusetts, USA.